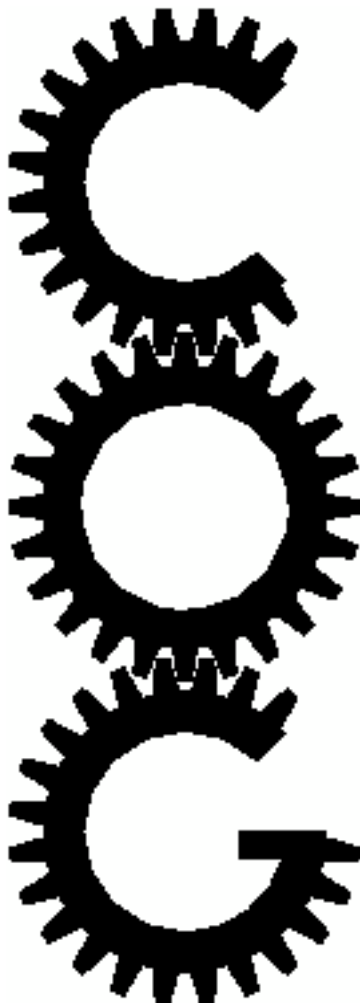




User's Manual

Sixth Edition

A Multiparticle
Monte Carlo
Transport Code

October 30, 2024



Lawrence Livermore
National Laboratory

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor Lawrence Livermore National Security, LLC nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

This work was produced at the Lawrence Livermore National Laboratory (LLNL) under contract no. DE-AC-52-07NA27344 (Contract 44) between the U.S. Department of Energy (DOE) and Lawrence Livermore National Security, LLC (LLNS) for the operation of LLNL. The rights of the Federal Government are reserved under Contract 44.

This report has been reproduced directly from the best available copy.

Available electronically at <http://www.doc.gov/bridge>

Available for a processing fee to U.S. Department of Energy
And its contractors in paper from:

U.S. Department of Energy

Office of Scientific and Technical Information

P.O. Box 62

Oak Ridge, TN 37831-0062

Telephone: (865) 576-8401

Facsimile: (865) 576-5728

E-mail: reports@adonis.osti.gov

Available for the sale to the public from:

U.S. Department of Commerce

National Technical Information Service

5285 Port Royal Road

Springfield, VA 22161

Telephone: (800) 553-6847

Facsimile: (703) 605-6900

E-mail: orders@ntis.fedworld.gov

Online ordering: <http://www.ntis.gov/ordering.htm>

OR

Lawrence Livermore National Laboratory

Technical Information Department's Digital Library

<http://www.llnl.gov/tid/Library.html>

Authorship

This manual is the sixth edition of the original COG manual written by Thomas Wilcox and Edward Lent in 1989. It was extensively revised through the years by Edward Lent, Richard Buck and Chuck Lee to correspond to new version of COG developed for UNIX-based workstations. Many new capabilities have been added to this version of COG and a few little-used capabilities have been deleted. The code is nearly completely backwards compatible with previous COG input files. The few exceptions are noted in the manual.

The authors of this manual update thank the many COG users who have offered useful suggestions for the improvement of code and manual. These contributors include our “power users”:

Shauntay Coleman
Jacob Glesmann
Jim Hall
David Heinrichs
Tony Nelson
Aaron Tamashiro
William Zywiec

Lastly, we ask our users to visit <https://cog.llnl.gov> and inform the authors at cog@llnl.gov of any bugs or suggestions for future development improvements.

Sincerely,

Ed Lent and Chuck Lee
Nuclear Criticality Safety Division
Lawrence Livermore National Laboratory
7000 East Avenue, L-198, Livermore, CA, 94550, USA

Dedication

This manual is dedicated to Tom Wilcox, Ed Lent, Richard Buck the principal authors of the COG transport code. Tom's experience and insight in shielding and criticality calculations provided the basis for choosing the most useful methods and best algorithms for incorporation into COG. Ed's knowledge and expertise in atomic and nuclear cross sections produced the scattering models and the cross section databases that form the physics foundation of COG.

Acknowledgments

We give special thanks to Drs. Wini Parker, James Hall, and Ken Sale for their dedicated efforts in scrutinizing early drafts of this manual, and for their many useful criticisms that have made this manual (and the COG code) a far better work.

We are especially indebted to Richard Neifert, the Test Program and L-Division Leader, for his energetic support and direction of the COG development effort. COG was made possible by the sponsorship and funding of the Department of Energy, through the LLNL Nuclear Test / Experimental Science Directorate. COG was developed to meet the needs of Test Program in computing the types and placement of shielding required for underground test diagnostics. COG played a significant role in the history of the Program's success in making difficult diagnostic measurements.

With the end of nuclear testing in the 1992 and subsequent demise of L-Division, management of the COG code transferred to Nuclear Criticality Safety Division and further development was made possible through the sponsorship and funding from Drs. Jerry McKamy and Angela Chambers, Managers of the Department of Energy Nuclear Criticality Safety Program.

Table of Contents

1	COG Introduction	16
1.1	Summary of Features	16
1.2	Overview of Monte Carlo Particle Transport	18
1.2.1	Neutron Transport and Cross Sections	19
1.2.2	Overview of Photon Transport	19
1.2.3	Photonuclear Reactions	20
1.2.4	Overview of Electron Transport	20
1.2.5	Overview of AlphaTransport	21
1.2.6	Overview of DeuteronTransport	21
1.2.7	Overview of Proton Transport	21
1.2.8	Overview of Activation Calculations	22
1.3	Why and When to Use COG	22
1.4	How to Use the COG Manual	23
1.5	Running COG	24
1.5.1	COG Job Execution in the Background	24
1.5.2	COG Background Job Control	24
1.5.3	COG Job Status/Termination: Control-C	24
1.5.4	COG Batch Runs	25
1.5.5	COG Parallel Runs	25
1.5.6	RESTART — Running COG Longer	25
1.6	COG Output Files Summary	28
1.7	Overview of Problem Specification	29
1.7.1	The Structure of COG Problem Input	29
1.7.2	COG Data Blocks and Keywords	30
1.7.3	Overview of COG Data Blocks	32
1.7.4	Input Notation Shortcuts	37
1.7.5	Manual Notation Conventions	38
1.7.6	Summary of COG Data Blocks	41
1.7.7	Catching Input Errors	42
1.7.8	Sample COG Problem	43
1.8	References	47
2	BASIC Data Block	49
2.1	Particle Types to be Followed	49
2.2	COG Units of Distance, Energy, and Time	50
2.3	BASIC Data Block Switches and Physics Models	51
2.4	Maximum Running Time – TMAX	51
2.5	Time Interval Between Restart Dumps – TDUMP	51
2.6	Particle Interval Between Restart Dumps – NDUMP	52
2.7	NOFINALRESTART	52
2.8	Starting Random Numbers – RN	52
2.9	Electron-Positron Annihilation Events – ANNIH–GAMMAS	53
2.10	Using Crit Detector Variance Reduction (Hybrid Mode) – CRITDETVR	54

2.11	PHOTONUCLEAR Including Photonuclear Reactions –	54
2.12	Including Delayed Neutrons in Fission Reactions – DELAYEDNEUTRONS	55
2.13	Delayed Fission Gammas	55
2.14	Number of Secondary Particles Exiting a Collision– NEUTRONPRODUCTION and PHOTONPRODUCTION	56
2.15	Fission-Produced Neutrons and Photons – NOFISSION and MAXFISSION	57
2.16	Probability Of Initiation – POI	58
2.17	Using Unresolved Resonance Region Probability Tables – URRPT	58
2.18	Using Fission Reaction Event Yield Algorithm – FREYA	58
2.19	Nuclear Resonance Fluorescence –NRF	58
2.20	Limiting the fraction of (n,X γ) events processed –FRACNXG	59
2.21	Writing Production Data Files– EPFILE and PHPFILE	59
2.22	Proton Physics Options	59
2.23	BASIC Data Block Example	60
2.24	References	62
3	The SURFACE Data Block	63
3.1	SURFACES and GEOMETRY—Concepts	63
3.2	Introduction to GEOMETRY—Approaches	65
3.3	Numbering and Naming Surfaces	67
3.4	Positive and Negative Sides of Surfaces	68
3.5	Finite Limits to Surfaces	70
3.6	Translation and Rotation (TR) of Surfaces	71
3.6.1	“SameAs” Surface Feature	76
3.7	First Order Planar Surfaces	78
3.7.1	Plane	78
3.7.2	Tetrahedron	81
3.7.3	Pentahedron	82
3.7.4	Box	83
3.7.5	RPP: Rectangular Parallelepiped	83
3.7.6	Hexahedron	84
3.7.7	Right Prism	86
3.7.8	Right Pyramid	87
3.8	The Whole Family of Second-Order Surfaces	89
3.8.1	Sphere	89
3.8.2	Ellipsoid	90
3.8.3	Right Circular Cylinder	91
3.8.4	Right Elliptical Cylinder	93
3.8.5	Hyperboloid of One Sheet	94
3.8.6	Hyperbolic Cylinder	95
3.8.7	Parabolic Cylinder	97
3.8.8	Right Circular Cone	98
3.8.9	Elliptical Cone	100
3.8.10	Hyperboloid of Two Sheets	101
3.8.11	Elliptic Paraboloid	102

3.8.12	Hyperbolic Paraboloid-----	103
3.9	Special Figures of Rotation-----	104
3.9.1	Torus-----	104
3.9.2	Curve Made of Straight-Line Segments Rotated About the X'-axis -----	105
3.10	Special Surfaces-----	107
3.10.1	General Analytic Surfaces -----	107
3.10.2	Topographical Surface -----	109
4	The GEOMETRY Data Block-----	110
4.1	Sector Definitions -----	110
4.1.1	The OR (Union) Operator -----	113
4.1.2	The NOT (Exclusion) Operator-----	115
4.2	Void and Infinite-Absorber Sectors-----	118
4.3	FILL Sectors-----	118
4.4	Specifying Boundary Conditions -----	120
4.4.1	Boundary Vacuum-----	120
4.4.2	Boundary Reflecting -----	121
4.4.3	Boundary Periodic-----	123
4.5	Geometry UNITS-----	124
4.5.1	The UNIT Definition -----	124
4.5.2	The USE UNIT Statement -----	126
4.5.3	The UNIT FILL -----	131
4.6	General Lattice Feature-----	131
4.6.1	Rectangular Lattice-----	131
4.6.2	X-Oriented Triangular Lattice -----	133
4.6.3	Y-Oriented Triangular Lattice -----	135
4.7	PICTURES of the Geometry -----	138
4.7.1	Introduction to PICTURES -----	138
4.7.2	PICTURE — Cross-Section -----	139
4.7.3	General Cross-Section PICTURE-----	140
4.7.4	Patterns for Cross-Section Pictures-----	143
4.7.5	PICTURE — Perspective-----	144
4.7.6	PICTURES and Error Checking-----	148
4.8	VOLUME Calculations -----	150
4.9	The SWEEP Statement -----	152
5	The MIX Data Block -----	154
5.1	Specifying Problem Materials -----	154
5.2	Special Material Numbers 0 and -1 (VOID and Infinite Absorber -----	158
5.3	Thermal Treatment -----	158
5.4	Data Libraries-----	158
6	Cross Section Libraries-----	164
6.1	Neutron Libraries (NLIB)-----	164
6.1.1	Secondary Neutron Libraries (NLIB2 and NLIB3)-----	165
6.2	S(α , β) Libraries (SABLIB and SABLIB2)-----	166
6.2.1	Using S(α , β) data in a COG run-----	167

6.3	Deuteron Libraries (DLIB)-----	167
6.4	Alpha Libraries (ALIB)-----	167
6.5	Photon Libraries (PLIB)-----	167
6.6	Photonuclear Libraries (PNLIB)-----	168
6.7	Probability Table Libraries (PTLIB)-----	168
6.8	Delayed Fission Gamma Libraries (DGLIB)-----	168
6.9	Dosimetry Libraries (DOSLIB)-----	168
6.10	Compounds and Mixtures-----	169
7	The I/O Data Block -----	172
8	The ASSIGN Data Blocks-----	175
8.1	Assigning Attributes to Sectors -----	175
8.1.1	ASSIGNing Physical Properties to Sectors-----	175
8.1.2	ASSIGN-M — Assign Material to a Sector-----	175
8.1.3	ASSIGN-MD—Assign Material and Density to a Sector-----	176
8.1.4	ASSIGN-D — Assign Density to a Sector -----	177
8.1.5	ASSIGN-ML — Assign Materials to a List of Sectors -----	177
8.1.6	ASSIGNing REGIONS and COLORS to Sectors -----	178
8.1.7	ASSIGN-R — Assign Region Number to a Sector -----	178
8.1.8	ASSIGN-RL — Assign Region Number to a List of Sectors -----	179
8.1.9	ASSIGN (General)— Assign Material Number, Region Number, and Density Factor to a Sector-----	179
8.1.10	ASSIGN-MC— Assign Plotting Colors to Materials -----	180
8.1.11	Summary of ASSIGN Data Blocks-----	181
8.1.12	Multiple Use of ASSIGN Blocks-----	181
8.1.13	Examples of ASSIGN Data Blocks -----	182
9	The SOURCE Data Block-----	184
9.1	Outline of General Method -----	184
9.1.1	Format of the SOURCE Data Block -----	185
9.1.2	Special Free-Form Inputs -----	186
9.2	Miscellaneous Parameters-----	188
9.2.1	NPART – Number of Source Particles -----	188
9.2.2	WRITESOURCE – Produce a COG SOURCE File -----	188
9.2.3	READSOURCE – Read a Previously-made SOURCE File-----	189
9.2.4	RETRACE Source Particles -----	189
9.2.5	CORRELATED Source Particles -----	192
9.3	Source INCREMENTS-----	194
9.4	Source IMPORTANCES-----	197
9.5	Source POSITION Dependence-----	201
9.5.1	Source POSITION Graphics-----	201
9.5.2	POINT Source -----	203
9.5.3	Surface Sources -----	205
9.5.4	Volume Sources -----	211
9.6	Source ENERGY Dependence-----	216
9.6.1	Source ENERGY Graphics -----	216

9.6.2	LINE Energy Source	218
9.6.3	CASCADE Line Energy Source	220
9.6.4	RADSRC Energy Source	221
9.6.5	GAUSSIAN Energy Source	223
9.6.6	WATT Fission Neutron Energy Source	224
9.6.7	MAXWELLIan Energy Source	225
9.6.8	ONE/E Energy Source	226
9.6.9	Point-Wise Energy DISTRIBUTION	227
9.6.10	BIN Energy Source	229
9.6.11	Spontaneous Fission Energy Source (SFS)	231
9.6.12	FissSrc: A New Source Energy Option	231
9.7	Source TIME Dependence	233
9.7.1	Source TIME Graphics	234
9.7.2	DELTA Function Time Source	235
9.7.3	PULSE Time Source	236
9.7.4	GAUSSIAN Time Source	237
9.7.5	Point-Wise Time DISTRIBUTION	238
9.7.6	BIN Time Source	239
9.7.7	STEADY State Time Source	240
9.8	Source ANGLE Dependence	241
9.8.1	Source ANGLE Graphics	244
9.8.2	FIXED Direction Angle Source	246
9.8.3	ISOTROPIC Angle Source	247
9.8.4	Point-Wise DISTRIBUTION in Polar Angle μ	248
9.8.5	BIN Angle Source in Polar Angle μ	249
9.8.6	GENERAL Angle Bins in μ and Φ	250
9.8.7	COG Standard Double-Angle Bin Structure	251
9.9	SOURCE-PATH Option	254
9.10	USRSOR – User-Defined Source Option	259
9.11	CENSUS Source File Option	261
9.12	Source Examples	263
9.13	References	273
10	The DETECTOR Data Block	274
10.1	Introduction to COG Detectors	274
10.2	Outline of the Detector Data Block	276
10.2.1	Detector Output Units	277
10.2.2	Detector Results	277
10.3	COG Detector Types	278
10.3.1	Reaction Detector	278
10.3.2	Boundary-Crossing Detector	280
10.3.3	Point Detector	282
10.3.4	Volume Point Detector	287
10.3.5	Pulse Detector	288
10.3.6	Imaging Detector	292

10.3.7	USRDET — User-Defined Detector Subroutine	295
10.3.8	Lucky Particle Detection Mode	297
10.4	Masks That Limit What a Detector Can See	299
10.4.1	Energy, MASK-E	299
10.4.2	Time, MASK-T	299
10.4.3	Single-Angle, MASK-A	300
10.4.4	Double-Angle, MASK-A*	300
10.4.5	Isotope, MASK-ISO	301
10.4.6	Reaction, MASK-REA	302
10.4.7	Collision, MASK-C	302
10.4.8	Region, MASK-REG (Point Detectors only)	302
10.4.9	Half-Life, MASK-HL (Activation problems only)	303
10.5	Detector Response Functions (DRFs)	304
10.5.1	Energy, DRF-E	304
10.5.2	Secondary Energy Detector Response Function, DRF2-E	307
10.5.3	Time, DRF-T	308
10.5.4	Single-Angle, DRF-A	309
10.5.5	Double-Angle, DRF-A*	310
10.6	BINs — Obtaining Differential Detector Results	312
10.6.1	ENERGY Bins	312
10.6.2	TIME Bins	313
10.6.3	ANGLE Bins - Single Angle	313
10.6.4	Multiple Bins	314
10.6.5	Output of Differential Detector Results	316
10.7	LISTing Properties of Scoring Particles	317
10.8	Tagged List-Mode Detector	317
10.9	Plotting Detector Differential Results	318
10.9.1	Changing Detector Plot Scales/Labels	319
10.10	ANALYSIS Data Block	323
10.10.1	Plots of Collision Sites	323
10.11	Calculated Detector IMPORTANCE	325
10.12	References	331
11	The Monte Carlo Random Walk	332
11.1	Analog vs. Biased Monte Carlo	332
11.1.1	Random Walk IMPORTANCE	335
11.1.2	Random Walk Figure-of-Merit (fom)	336
11.2	Random Walk Modification Techniques:	337
	The WALK Data Block	337
11.2.1	WALK-XX	338
	How to Specify WALK Parameters	338
11.2.2	WALK-AGE	342
	Particle Termination Based On Age	342
11.2.3	WALK-BC	343
	Splitting/Russian Roulette at a Boundary	343

11.2.4	WALK-COLLISION	345
	Splitting/Russian Roulette at a Collision	345
11.2.5	WALK-ENERGY	348
	Particle Termination Based On Energy	348
11.2.6	WALK-FC	349
	Forced Collisions	349
11.2.7	WALK-FPP	351
	Fast Photon Physics	351
11.2.8	WALK-ISOTOPE	352
	Forced Collision with an Isotope	352
11.2.9	WALK-PRODUCTION	353
	Secondary Particle Control	353
11.2.10	WALK-PS	354
	Path Stretching/Exponential Transform	354
11.2.11	WALK-REACTION—Forced Reaction	358
11.2.12	WALK-SDB—Scattered Direction Bias	359
11.2.13	WALK-SURVIVAL	365
	Survival of a Particle in a Collision	365
11.2.14	WALK-WT	367
	Splitting/Russian Roulette Based on Weight	367
11.3	Random Walk Biasing Examples	368
11.3.1	Example 1— Splitting at a Boundary	368
11.3.2	Example 2—Path Stretching	378
11.3.3	Example 3—Direction Biasing	391
11.4	References	398
12	The CRITICALITY Data Block	399
12.1	CRITICALITY Results	402
12.2	Alpha Calculations	405
12.2.1	Alpha Eigenvalue	405
12.2.2	Alpha Dynamic	405
12.2.3	Alpha New	406
12.3	Criticality ENDL Limitation	407
12.4	Criticality Restart—the DUMP Data Block	407
13	The ACTIVATION Data Block	410
14	Parallel Processing	414
14.1	Setting Up A Parallel Processing Run	415
14.2	The PARALLEL Data Block (optional)	416
14.2.1	Control Modes for Shielding Calculations	418
14.2.2	Control Modes for Criticality Calculations	419
14.3	Running COG in PARALLEL Mode under MPI	420
14.3.1	When Trouble Occurs...	420
14.4	Running COG in PARALLEL Mode under HPC (LLNL LC Only)	421
14.4.1	When Trouble Occurs...	424
14.5	Very Long COG Run with Restart	424

14.5.1	Criticality Run with Restart	424
14.5.2	Source Run with Restart	426
14.6	References	427
15	Understanding Your COG Run	428
15.1	COG OUTPUT Files	428
15.1.1	Graphics Output – The ps File	428
15.2	Graphics Output	430
15.2.1	Text Output – The out File	430
15.2.2	Detector Output The det File	434
15.2.3	Cross Section Files	434
15.2.4	LIST of Scoring Particles – The lst File	435
15.3	COG Statistical Considerations: Elementary Statistics	435
15.4	The Problem of Under Sampling	437
15.5	Precision and Accuracy – a Reality Check	441
15.6	Estimated Errors in COG	443
16	Electron Transport: Using the EGS5 Transport Code Kernel	444
16.1	Specifying Electron Transport	444
16.1.1	The DNEAR Geometry Option	445
16.2	EGS Data Block	446
16.2.1	Bremsstrahlung Angular Distribution	447
16.2.2	Known Limitations	447
16.3	Example Problem	448
16.4	References	450
17	Proton Transport	451
17.1	Hadronic Collisions	451
17.2	Multiple Small-angle Coulomb Scattering	452
17.3	Continuous Slowing Down Approximation (CSDA)	452
17.4	Energy Straggling	452
17.5	Transport Step Size Control	453
17.6	Transport in a Magnetic Field	453
17.7	Setting Up a Proton Transport Problem	454
17.8	Low-Energy Cutoffs	456
17.9	Detectors	456
17.10	Known Limitations	456
17.11	References	457
17.12	BFIELD Data Block	458
17.12.1	Magnetic Field Sectors	458
17.12.2	BFIELD Limitations	459
17.13	Example Problem	460
18	Alpha Transport - ALPHATRANS	462
18.1	Alpha Transport Example 1	463
18.2	Example 1 COG input deck:	464
19	Deuteron Transport - DEUTTRANS	468
19.1	Deuteron Transport Example 1	468

19.2	Example 1 COG input deck -----	469
20	COG Example Problem 1 -----	471
20.1	Problem 1 Description and Input -----	471
20.2	Problem 1 Results -----	476
20.2.1	The ps file for Problem 1 -----	476
20.2.2	The out file for Problem 1 -----	484
21	COG Example Problem 2 -----	501
21.1	Problem 2 Description and Input -----	501
21.2	Problem 2 Results -----	503
21.2.1	The ps file for Problem 2 -----	503
21.2.2	The out file for Problem 2 -----	508
22	COG COGLEX11.2 Dictionary Listing -----	515
23	COG Reaction Number Listing -----	546
24	Index -----	548

1 COG Introduction

COG is a high-resolution code for the Monte Carlo simulation of coupled particle transport in arbitrary 3-D geometry. COG will transport neutrons, protons, deuterons, alpha particles with energies up to hundreds of GeV, and photons with energy ranges limited by the available cross section sets and physics models. Electrons can be transported via the EGS5¹ electron transport kernel, electrons can also be transported. The COG code is a significant upgrade from earlier Monte Carlo transport codes and has been written specifically to make it more versatile, accurate, and easy to use. COG has provisions for calculating deep penetration (shielding) problems, criticality problems, and neutron activation problems while retains all of the standard capabilities found in other Monte Carlo transport codes. COG uses high-resolution pointwise cross-section databases and makes no compromises in the transport physics, so that the results of a COG run are limited only by the accuracy of the databases used. COG runs primarily on Linux Operating System workstations with MPICH software installed – currently, Red Hat 7 & 8, Windows 10 (Windows Subsystem for Linux –WSL), Ubuntu 16, 18 & 20, OpenSUSE Leap 15.2, Fedora 32, Apple Power Mac with Intel CPU (with MacPorts installed) workstations, and LLNL LC supercomputer CTS-1 cluster with TOSS 3 are supported.

1.1 Summary of Features

Some of the main features in COG include:

- COG problem specification is user-friendly. User inputs are free-form and extensively checked by the code. Pictures are made of the problem geometry and sources so that the user can easily identify and correct input errors.
- The cross section data sets are the key physics data for any Monte Carlo transport code. COG was designed to be as accurate as the underlying cross sections would allow, and uses commonly available pointwise data sets. See the Appendix for a detailed listing of transported particles and available databases.
- COG geometry is described by specifying surfaces that bound each volume in the problem; for example, planes, cylinders, spheres, and tori. Analytical surfaces up to fourth order may be used. In addition, COG supports *pseudosurfaces* which describe items like boxes, finite cylinders, and topographic surfaces. COG has a larger list of available surfaces and figures compared to other Monte Carlo transport codes allowing for the world to be described more adequately and efficiently. For problems involving multiple use of sub-assemblies, the sub-assembly can be defined once as a COG UNIT and used many times. To check the validity of a geometry setup, COG can calculate volumes of parts and draw cross-sectional or perspective pictures of the user's geometry.
- COG particle sources are very general and flexible which allows input from a previously-generated source or a user-defined source.

- A Monte Carlo transport calculation often requires the simulation of millions of particle paths to achieve a reliable result, a calculational process that can take hours to days of computer time. A catalog of Monte Carlo biasing techniques is available in COG to modify the random walk procedure to provide better answers using much less computer time. These biasing techniques include:

- **Splitting and Russian roulette, at collision sites and boundaries.**

- **Path stretching.**

- **Survival weighting.**

- **Scattered direction bias.**

- **Reaction bias.**

- **Isotope bias.**

- **Forced collisions.**

- **Particle weight control.**

- **Secondary particle production control.**

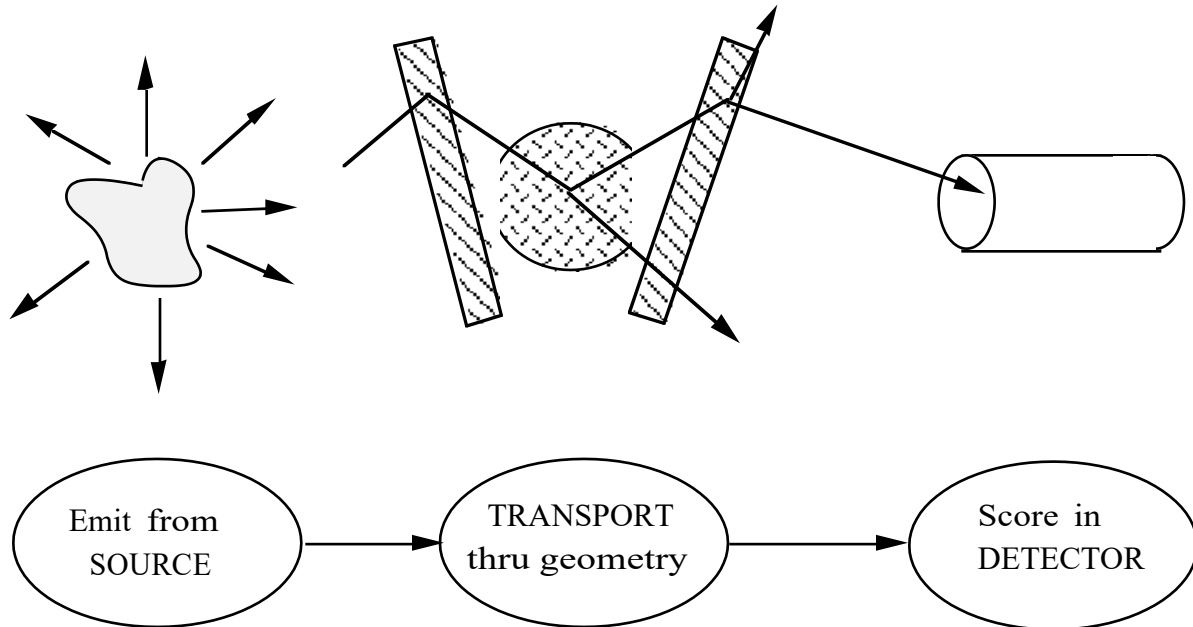
- COG output includes a summary of random-walk events tabulated for user-specified REGIONS. For each reaction in a region, COG lists the number of particles scoring and the energy deposited. Optional ANALYSIS plots show the location of scattering events in the problem's phase space.

- A COG run produces results in the form of DETECTOR responses in user-designated scoring regions. Detector types include REACTION, BOUNDARY-CROSSING, POINT, PULSE, and IMAGE detectors. MASKS may be specified to limit detector scoring to some part of ENERGY, TIME, REGION, COLLISION, or ANGLE space. Detectors may also be masked by the HALF-LIFE of the scoring particle or the ISOTOPE and/or REACTION of the last event prior to scoring. Each detector may have a RESPONSE specified as a function of energy, angle, or time. Differential BINS may be specified in time, energy, polar angle, and azimuthal angle, or any combination of these. The relative IMPORTANCE of events causing detector scores may also be listed.

- In the criticality mode COG makes calculations of certain kinetics; the effective delayed neutron fraction, β_{eff} , and the decay time constant, α .

1.2 Overview of Monte Carlo Particle Transport

A Monte Carlo transport code simulates the actual trajectories of particles as they move through the materials that make up the problem's geometry. In a typical problem, thousands to millions of particle histories are computed to determine the “average” particle transport to any given location. It is helpful to think of COG as consisting of three major computational phases: Source, Transport, and Detection.



Particles are "born" at a particle SOURCE, with an initial time, energy and direction obtained by sampling a COG Source distribution. In the Transport phase, COG tracks each particle through the materials of the geometry and computes its interaction with the atoms of each material.

Neutrons and photons are neutral particles which travel undeflected in linear trajectories except for occasional point-like collisions in the near proximity of atomic electrons or nuclei. These collisions can either be elastic, scattering the incident particle in another direction, or inelastic, perhaps creating new secondary particles and depositing energy locally. A sampling procedure tells COG the average distance the particle travels through a material before colliding with an atom. For each collision event, COG samples the cross-section database to determine if the particle is to be elastically or inelastically scattered. This process is repeated until the particle is absorbed, escapes from the problem, or is terminated because its energy or age has fallen outside of predetermined limits. Particles which enter Detector regions are counted or "scored". COG output largely consists of detector scores. A primary

source particle may create secondary particles in collisions; these are also tracked and scored.

Protons and electrons are charged particles which interact with materials via the long-range Coulomb force; thus these particles continually undergo deflection from a linear trajectory as they traverse a material. Transport of these particles is accomplished by the condensed history method, which models the cumulative effect of many small Coulomb scatterings as the particle is advanced a macroscopic step.

1.2.1 Neutron Transport and Cross Sections

All the reaction physics used in transporting neutrons comes from the COG databases, which are based on well-known nuclear reaction data libraries.

New ENDF/B-VIII.0 nuclear data libraries are available for continuous energy cross-sections (ENDFB8R0 and ENDFB8R0.ACE), probability tables for the unresolved resonance region (PT.ENDFB8R0.ACE), and thermal scattering laws (T.ENDFB8R0 and T.ENDFB8R0.ACE). Note that the ACE libraries were originally processed using Los Alamos National Laboratory's NJOY; whereas, the library ENDFB8R0 was processed using Red Cullen's PREPRO2018. LLNL utility codes were then used to create the library files named above for direct use in COG. (See Appendix for a complete listing of available databases.)

For these libraries, the cross sections have been Doppler broadened to represent room-temperature cross sections.

Two types of thermal neutron treatment are available: the hot gas model (the default), or the $S(\alpha,\beta)^5$ model, which attempts to account for molecular binding effects. The $S(\alpha,\beta)$ model is only employed for incident energies below a few eV.

Results obtained from transport calculations using these data will only be as good as the cross-sections allow. The successful user will acquire a feeling for the cross-section information and its accuracy and the consequent accuracy of their calculated results. To assist the user in this process, COG can plot the total cross section of materials and isotopes in the problem (see **The I/O Data Block**). There are ancillary routines external to COG which will list desired parts of any cross-section library.

1.2.2 Overview of Photon Transport

Photon reaction data are obtained from the Lawrence Livermore National Laboratory's Evaluated Photon Data Library (EPDL).⁶ The database includes cross-section information for the reactions: Rayleigh, Photoelectric, Compton, and Pair-Production.

EPDL89 is EPDL, 1989 version. EPIC2017 is EPDL, 2017 version

EPIC2023 is EPDL, 2023 version

EPDL97 is EPDL, 1997 version. (**This is the default photon library.**)

For the Photoelectric reaction, relaxation data for the emission of secondary electrons and photons are included.

The default gamma cross sections are the LLNL Evaluated Photon Data Library, EPDL97.

Note: that the photon library does **not** include photonuclear reactions.

The default photon transport option is **all physics enabled**. Alternatively, the user can specify the Fast Photon Physics Option, which speeds up transport of photons by replacing COG's normal physics models with faster approximate-physics versions.

1.2.3 Photonuclear Reactions

When the **PHOTONUCLEAR** option is invoked COG forces a photonuclear reaction to occur in concert with every Monte Carlo photon reaction. The photonuclear event is weighted by the ratio of the total photonuclear cross section to the total photon cross section. A particular photonuclear event is selected from the possible photonuclear events and any secondaries produced are placed on the secondary bank to be followed as part of the normal random walk.

Note: COGPNUC does not contain data for all isotopes available on the photon and/or neutron data libraries.

1.2.4 Overview of Electron Transport

COG performs coupled transport of electrons, positrons, and photons, using the EGS5¹ electron transport kernel. Whenever an electron-producing photon reaction occurs, COG checks whether the reaction occurred in a region enabled by the user for electron transport. If not (the standard case), then the electron energy is immediately deposited. If enabled, then the electrons are placed on the EGS kernel stack for transport. If secondary photons due to bremsstrahlung or positron annihilation occur, these photons are placed on the COG stack for COG to follow.

Electron transport is accomplished by advancing the particle in a straight-line macroscopic step, which encompasses numerous microscopic scattering events. At the end of the step, the electron is deflected through a scattering angle chosen from the Moliere multiple small-angle scattering distribution. Energy loss is modeled by allowing discrete bremsstrahlung and knock-on events above a specified energy threshold, and using the continuous slowing-down approximation (CSDA) for lower-energy events. Stopping powers and collision data used in a run are obtained from a library file prepared by running the PEGS database generator for the problem materials. See the EGS5 Manual for further information.

1.2.5 Overview of Alpha Transport

Alpha transport is handled as a two-step process.

1) CSDA. Using data and coding borrowed from the AlfaMC code¹², a Continuous Slowing Down Approximation for alphas has been implemented in COG. Following the guide of the AlfaMC code: the NIST/ASTAR¹³ stopping-power database is used; Gaussian, Vavilov, or Landau distributed energy straggling is performed; and a simple Fermi small-angle multiple scattering model is adopted. Since for the energy range under consideration (~1 to 20 MeV), the mean free path for nuclear interactions (e.g., (α, n), (α, γ), etc.) is very much greater than the range of the alpha, the CSDA approach alone is used to track the alpha, i.e., nuclear interactions are ignored in this step.

2) Nuclear interactions. A fraction of the CSDA steps is sampled for nuclear reactions – i.e., for each step if a random number is less than the fraction, then nuclear reactions are included, and any (appropriately weighted) secondaries are produced and followed.

1.2.6 Overview of Deuteron Transport

Deuteron transport is modeled after alpha transport.

1) CSDA. Following the guide of the AlfaMC code: the NIST/PSTAR¹³ stopping-power database is used - dE/dx for protons at energy E = dE/dx for deuterons at energy $2E$; Gaussian, Vavilov, or Landau distributed energy straggling is performed; and a simple Fermi small-angle multiple scattering model is adopted. Since for the energy range under consideration (~1 to 20 MeV), the mean free path for nuclear interactions (e.g., (d, n), (d, γ), etc.) is very much greater than the range of the deuteron, the CSDA approach alone is used to track the deuteron, i.e., nuclear interactions are ignored in this step.

2) Nuclear interactions. A fraction of the CSDA steps is sampled for nuclear reactions – i.e., for each step if a random number is less than the fraction, then nuclear reactions are included, and any (appropriately weighted) secondaries are produced and followed.

1.2.7 Overview of Proton Transport

COG uses a condensed-history method to transport protons. At the end of each linear macroscopic step, the particle is scattered through an angle chosen from the Moliere distribution in Gaussian approximation (Lynch and Dahl). Energy loss is computed via the CSDA approximation, using the stopping power curves of Anderson- Ziegler for $E_p < 100$ MeV, and using the Bethe-Bloch stopping power formula above this limit. In addition, the Landau straggling distribution is sampled to account for the statistical variation of energy loss about the mean. The proton may undergo an inelastic nuclear scattering event, which is modeled using the Letaw fit to experimental cross section measurements. This event is treated as an absorption. Protons may be tracked through a vacuum magnetic field. Currently, only quadrupole fields inside a cylindrical magnet are supported.

1.2.8 Overview of Activation Calculations

COG can tally the photon flux due to neutron activation of materials, by making use of the Activation Library.⁸ This library provides data for neutron-activated isotopes and one- and two-step decay schemes. When an activation event occurs during neutron transport, COG samples the decay data to determine a half-life for photon emission. For each of a set of specified tally times, COG computes the flux at all photon detector locations due to the photons released by the radioactive decay. A Half-Life Mask can be specified to limit the detector response to a range of emission half-lives.

1.3 Why and When to Use COG

For complex geometries, the transport of radiation can be estimated either by experiment or by a Monte Carlo simulation of the transport process. In nearly all cases, calculating the radiation transport is the most reliable and cost- and time-effective method. Distributed or multiple radiation sources, convoluted geometries, multiple scattering contributions, combined neutron and gamma ray fields, complicated time history of background signals, criticality calculations, etc., are all problems almost uniquely amenable to a Monte Carlo transport calculation and often nearly impossible to accurately estimate by other means. Many insight-generating computational experiments are possible in Monte Carlo simulation which are not realizable in the laboratory. The new user may wish to review the Monte Carlo radiation transport method.^{9,10}

Hand or spreadsheet calculations and other techniques such as simple point-kernel estimation, discrete-ordinates codes in one or two dimensions, or extrapolation from experimental results are all viable techniques to solve a significant number of radiation transport problems. If an acceptable answer with any simpler method is obtainable than by employing any general Monte Carlo transport code, then do so. Setting up and running COG, or any general Monte Carlo transport code, takes an appreciable investment in both human and computer time.

COG is not a black box. Just specifying a correct SOURCE, GEOMETRY, and DETECTOR will not necessarily yield the correct answer. Some knowledge of the physics of particle transport is essential to understanding whether the results are “reasonable”. Monte Carlo particle transport is a marvelous, but sometimes tricky, calculational technique. Occasionally, an answer with a very small standard deviation, of three orders of magnitude smaller than the experimental result is often due to inadequately sampling or modeling the energy and geometrical space and/or inappropriate use of biasing techniques. To minimize these possible problems, experimental measurements and simple calculations should always be employed to check that the “answers” calculated are “reasonable”.

COG calculations, set up and run correctly, can give the best answer possible within the constraints of the cross sections available. COG can provide a benchmark

calculation for a new untried system (too expensive or too hazardous of a configuration to evaluate experimentally). COG is often used to calculate a benchmark result in order to evaluate the performance of transport codes which use more approximate physics models. Once set up, a particular COG job can be quickly and simply iterated to find an optimum solution.

1.4 How to Use the COG Manual

The first-time user will profit from reading the introductory parts of these Manual sections: BASIC, SURFACE, GEOMETRY, MIX, DETECTOR, and WALK. This will give you a perspective of problem specifications without getting you too involved in detailed descriptions. Then you can read more closely those sections that are of interest to you. The example problems in this chapter and in the **Sample Problem Section** should give a good overview of COG input requirements.

Each major section has an introduction that briefly discusses what the section covers. The rest of the section consists, for the most part, of the detailed specifications required to activate code features. The experienced user may only need to refer to this material. To make this referencing easier, we have provided an extensive index and examples of COG input specifications.

If it is necessary to bias the random walk in a problem, then go to the various WALK sections and view the WALK examples.

1.5 Running COG

To run COG interactively on a UNIX workstation (once you have COG installed and set up correctly), log in and type:

COG input-file

where *input-file* is the user-specified name of the ASCII text file containing the COG problem specification.

1.5.1 COG Job Execution in the Background

Some COG jobs can run for many hours or even days. To run a COG problem in the **background**, without tying up your input terminal, use the command line:

COG input-file >& *ttyfile* &

This command line puts the COG calculation into the background mode and sends terminal output and error messages to the file *ttyfile*. The background mode allows you to use your terminal in a normal manner while the problem is running. Several COG jobs can be run simultaneously in the background. UNIX assigns a process ID number (**pid**) to each background job and reports it to the terminal. This allows you to track the progress of your background COG jobs and terminate them if necessary.

1.5.2 COG Background Job Control

UNIX Job Control exists for background and detached COG jobs. Typing

kill -USR1 *pid*

sends a signal to the background COG job with process ID number *pid*, which will cause COG to write a status line into a file named *inputfile.status*. This file may be examined with a UNIX editor (e.g., **vi**, **more**, **cat**) to see how many particles COG has run up to that point.

Typing

kill -USR2 *pid*

signals the specified COG job to write a final status line into the file and terminate in a restartable fashion.

1.5.3 COG Job Status/Termination: Control-C

To check the status of a COG job running in the **foreground**, or to terminate the run after the current batch or particle has finished, type **Control-C**. The current job status will be listed on the screen. One of three options will be prompted:

c: continue to resume the calculation;

s: stop to end the COG job nicely with the usual scoring tallies for the particles run, saving the results of the calculation and leaving behind a restart file;

k: kill to stop COG immediately without producing scoring tallies or saving anything.

1.5.4 COG Batch Runs

A series of COG jobs may be run back-to-back in the **background** mode by means of the **batchcog** utility, which takes as input a file that lists the names of the COG input files to be run, one name per line. Each job is run, in turn, until it is complete; then the next job is started.

To start **batchcog**, type:

```
batchcog COG batchfile >& logfile &
```

where

batchfile is the file that contains the list of COG input files;

logfile is the name of a file to receive the tty output.

All the input files should reside in the same directory as *batchfile*.

To check on **batchcog** status, type:

```
ps -ef|grep COG
```

This command will report the input file that COG currently has open.

1.5.5 COG Parallel Runs

COG will execute concurrently on a set of networked workstations or on a MPP (Massively Parallel Processor), using the MPI¹⁷ (Message Passing Interface) library from MPICH software. Parallel processing is the fastest way to solve long COG problems. See the **Parallel Processing** Section.

1.5.6 RESTART — Running COG Longer

Many COG jobs must be run for many hours or even days to generate results with acceptably small statistical errors. To maximize ease of use and flexibility, a **RESTART** option has been implemented for shielding-type calculations. (Criticality-type job restarts are also available—see the **CRITICALITY** section.) When the calculation ends, either normally (by running the desired number of particles) or by user interrupt (using the **control-c** option), a **RESTART** file, named *inputfile.rst*, is written to disk. COG also periodically creates a **RESTART** file after every hour of CPU time (by default), so at most only 60 minutes of calculation time is lost if the machine "crashes" during a long calculation. When the *inputfile.rst* file is available, any job can be restarted. Restarts are accomplished by adding the line

(as a one word Data Block) to the input file and, possibly, changing **NPARTICLES** (in the **SOURCE** Data Block). NPARTICLES is the **total** number of particles the problem will run for the entire problem (including particles run prior to the restart).

- 26 -

RESTART N *restart_dump_no* (i.e. xxxxxxxxxxxx)

where *restart_dump_no* is the source particle number of the restart dump you wish to use. See the **Parallel Processing** Section.

Note: This is all well and good if you are running in serial mode but running (shielding jobs) in parallel mode is quite different. Things work as advertised, but with caveats. The slaves report back to the master approximately every 10 minutes (this is the default and can be reset). This is the only chance to make restart files in parallel mode. In the **TDUMP** mode, assuming a default dump time of 60 minutes and a slave report time of 10 minutes, the first restart file is 60 to 70 minutes after the job start, the second restart file is 60 to 70 minutes after the previous restart file, and so on. In the **NDUMP *mpartdump*** mode, again assuming a slave report time of 10 minutes, the actual number of particles per dump depends on the relative interplay of *mpartdump* and the number of particles processed by the slaves in 10 minutes and will not generally be a multiple of *mpartdump*. For example, a job with an **NDUMP 50000** may have restart files at 66774 particles, 133131 particles, and 166522 particles.

1.6 COG Output Files Summary

COG provides a wealth of output intended to ensure the problem is properly set up and to help the user gauge the validity of the answers. Each COG run produces several output files, which reside in the same directory as the COG input file. Output files are named by adding suffixes to a root name, which is the name of the input file. Where output file names are discussed in this manual, the root name is indicated by

A full explanation of the outfiles is given in the section **Understanding Your COG Run: The COG OUTPUT Files**.

The following files can be created by a typical COG run:

....**out**: COG principal output text file containing a listing of the input-file and calculated results.

....**ps**: COG PostScript-format graphics file containing geometry PICTUREs, cross-section plots, ANALYSIS results, and DETECTOR results. COG graphics are generated via the PGPLOT Graphics Subroutine Library¹¹

....**src**: COG source file containing the source parameters and starting coordinates of all particles run. This allows the user to run CORRELATED problems. It is only produced if an input flag is set.

....**det**: COG text file containing DETECTOR results (COG answers), in a form suitable for post-processing.

....**rst**: COG restart file containing restart parameters for a shielding job (see **RESTART**).

....**rdnnnn**: COG restart file containing restart parameters for a Criticality job (see **Criticality Restart**).

....**status**: COG status files created when COG receives a USR1 or USR2 signal while running in the background mode (see **RESTART** and **COG Background Job Control**).

....**list**: COG list of particles created when using the COG LIST option to track a particular result in a detector. (see **LISTing Properties of Scoring Particles**).

....**outfile**: COG list of time, energy, and score of individual detector events. (see **TAGing of Scoring Particles in the Detector section**).

....**cnvrt**: COG ‘convert’ file – modified input file. (see **COGcnvrt Option in the Geometry section**).

....**detid.census**: COG census file containing the parameters of all particles that scored in detector detid. (see **CENSUS File Option in the Detector section**.) This file may be used as a SOURCE file in a subsequent COG run (see **CENSUS File Option in the Source section**).

1.7 Overview of Problem Specification

1.7.1 The Structure of COG Problem Input

In every transport problem, particles are born at a SOURCE, with specified energies and directions. They are then transported through the GEOMETRY, which consists of the defined three-dimensional space of the problem, and interact with the atoms of the problem MATERIALS. At every collision, particles will be absorbed or scatter (change direction and perhaps energy) and possibly create secondary particles. Some particles may eventually reach DETECTORS, where they will be counted or scored. These detector results are the COG "answers".

Input to COG consists of preparing an ASCII input file that specifies the properties of SOURCE, GEOMETRY, MATERIALS, and DETECTORS, and turns on biasing techniques for the problem at hand. To each of these problem parts there corresponds a COG "Data Block" of user-specified data. Each Data Block begins with an identifying block name. Blocks that are usually specified for every problem are:

TITLE: Problem name;

BASIC: Specifies particles to be followed, and other general information;

SOURCE: Describes the source;

SURFACES: Describes the surfaces that bound the three-dimensional volumes of the problem (e.g., spheres, cylinders, planes);

GEOMETRY: Describes the SECTORS (volumes) of the problem, in terms of the bounding surfaces;

MIX: Specifies the problem's physical materials (e.g., carbon, air, aluminum);

DETECTOR: Specifies the type (e.g., BOUNDARY, REACTION, POINT, PULSE) and location of the detectors.

The detailed definitions of these and other Data Blocks are given in the following section.

1.7.2 COG Data Blocks and Keywords

COG ASCII input consists of blocks of input information (Data Blocks) headed by the Data Block's name. Each Data Block name is followed by one or more input lines containing keywords and ASCII strings or numerical data. This data may extend over many input lines. COG recognizes the end of a data set by finding the next keyword or Data Block name.

Example: *COG SOURCE Data Block*

```
SOURCE
  NPARTICLES = 1000
  FILE = reactor-problem.sor
  RETRACE 101 102 115
```

Here, SOURCE is a Data Block name, NPARTICLES, FILE, and RETRACE are keywords, reactor-problem.sor is a filename, and 1000, 101, 102, 115 are data fields.

The order of the keywords in a given Data Block usually does not matter. The order of the data fields after the keyword is significant and must be input in the prescribed sequence(s).

Not all Data Blocks are needed in any one problem. Only the first line (the TITLE Data Block) and the last line (the single word END) must have to be in a specified order.

Data Block names and keywords may be typed in upper or lower case. In this manual, these are shown in upper case for ease of identification. Many keywords have shorter equivalents (aliases) for convenience.

Restrictions:

- Data Block names and keywords are reserved words and should **only** be used within the problem to denote the start of a new Data Block or keyword. (There are no word restrictions inside comment fields).
- File names may contain up to 32 characters; titles may contain up to 80 characters.

Two-word "keywords" are written with a dash between them, so the code can recognize them as a single unit.

Example of a keyword which consists of two words:

```
BOUNDARY-CROSSING
```

Keyword lines

Most keywords are followed by number fields in a prescribed order. All types of numbers (integers, scientific, or floating point) are equivalent, but integers are recommended for ID numbers. Data may extend to 512 characters per line and can be continued from line to line. Data fields may be separated by tabs, spaces, commas, equal signs (=), or end-of-line separators. A string containing spaces or other separators must be enclosed in quotes to prevent the parser from breaking it into pieces. Keyword entries are terminated by the next keyword or Data Block name.

***Example:** Specification of surface #20, a BOX, and its Translation/Rotation parameters.*

```
20 BOX 0.2 10.0 10.0 TR 4.1 0. 0.
```

Comments

Comments may be inserted anywhere in the input file, after the first line. A comment field consists of any text that begins with the character \$ and continues to the end of the line. The \$ sign terminates all further COG processing of that line.

Example:

```
$ This whole line is a comment
MAT = 1 CO 8.85 $ The rest of this line is a comment
```

Comments in the input file are an excellent way to write yourself notes explaining where certain information came from, what you were doing or trying to calculate, and the approximations that were made. When you go back to modify and rerun a problem that was initially run months before, such comments can be very valuable. Also note that all lines after the END statement are ignored by COG, so users may “store” remarks about the job or temporarily unneeded input lines by placing them *after* the END statement.

1.7.3 Overview of COG Data Blocks

Seven main COG Data Blocks are *required* in all problems. These provide the necessary information to describe a problem. Supplying these Data Blocks will produce some basic results—energy deposition and reactions by type in specified geometric regions as well as how many particles crossed boundaries.

Required COG Data Blocks:

TITLE

By definition, the first line of a COG input file is the problem TITLE Data Block. It should be less than 80 characters (including spaces) long. The keyword TITLE is not used to start this line. Comments cannot be placed before this line.

Example of a TITLE input line:

```
COG INPUT FOR PROBLEM #9  JANUARY 3, 1994
$ A SET OF COMMENT LINES FOLLOWING THE TITLE CAN BE
$ USED TO FURTHER IDENTIFY THE PROBLEM AND ITS
$ PARAMETERS
```

BASIC

This Data Block describes the units (length, time, energy) used in the problem, the particle types to be followed, and “special” features. Physics switches are also found here. The units specified in this block are used in all other Data Blocks. The only *required* input in the BASIC Data Block is the *type of particle(s)* in the problem. The default units are: *cm*, *seconds*, and *MeV*.

Example of a BASIC Data Block:

```
BASIC
PHOTON CM
```

SURFACES

The SURFACES Data Block specifies all surfaces used in the problem geometry. Surfaces form the boundaries between the physical objects or *things* you are describing. Each surface is given an identifying number, *surf-ID#*, for later reference in the GEOMETRY Data Block. Surfaces may also be used to define any volumes of interest and do not have to correspond to physical boundaries.

Example of a SURFACE Data Block:

```

SURFACES
  20 PLANE  0. 0. 0.    2. 0. 0.
  21 PLANE  1. 0. 0.    2. 0. 0.
  22 P Y 3

```

GEOMETRY

The GEOMETRY Data Block specifies the SECTORS or volumes that compose the problem geometry through which the particles are to be tracked. In other codes, these volumes are often termed “zones”. The sector is the smallest unit of geometry that you can specify. For clarity, or for statistical purposes, these sectors may be much smaller than the physical size of any one geometric component. Each sector is defined by listing the *surf-ID*#s of the surfaces which bound it. The GEOMETRY keyword PICTURE is used to create cross-sectional (CS) and perspective (P) pictures of the geometry, and to check the geometry for errors. Geometry error checking is turned off during normal execution of the problem.

Example of a GEOMETRY Data Block:

```

GEOMETRY
  SECTOR 10 Sec1 +20 -21 OR -22 +5

```

MIX

The MIX Data Block defines the different materials, such as steel, lead, water, sand, or air, used in the problem. In the MIX Data Block, the user specifies the exact composition and density (gm/cc) of each material used in the problem and assigns a material number (*mat-ID*#) to each one.

Example of a MIX Data Block:

```

MIX  MAT = 1 SI 1.23

```

Alternatively, the user may specify the amount of constituents by weight fraction, atomic fraction, or atoms/barn-cm.

ASSIGN

By default, SECTOR **n** is assigned MATERIAL number **n** and REGION number **n**. The various ASSIGN Data Blocks allow you to override these defaults and attach a specified MATERIAL number and/or a REGION number to each SECTOR. A REGION is a set of one or more sectors that are to be grouped for analysis purposes. By using the ASSIGN statement, the user can make a group of SECTORS contain the

same material but differ in REGION number. It is also possible to specify a relative density factor df to be applied to the material within the sector.

Example of an ASSIGN Data Block that assigns SECTOR 1 to REGION 2 and SECTOR 3 to REGION 7.

```
ASSIGN-R
1 2      3 7
```

SOURCE

This Data Block specifies the source particle distribution in location, angle, energy, particle type, and time. The source may be set up to emit realistically (e.g., isotropically) in order to fully simulate the physical problem being modeled, or it may be constrained to emit into a small solid angle about a preferred direction, in order to more rapidly arrive at a valid answer. (But it should be noted that any such variation on the simple analog Monte Carlo technique may lead to errors if used improperly. See WALK sections for more information on Biasing.) *Criticality problems have their own source definition and do not use the SOURCE Data Block.*

Example of a SOURCE Data Block. These lines specify the SOURCE position dependence to be a sphere of radius 0.1 centered at (1., 0., 0.).

```
SOURCE
  DEFINE POSITION = 1
  SPHERE 1. 0. 0. 0.1
```

END

This word on a single line signals the end of problem input. COG reads no more input lines after encountering this END flag. All data or information after the word END will be disregarded. Input lines previously used or which may be used in subsequent runs can conveniently be stored after the END card. Comments can also be placed after END.

OPTIONAL COG DATA BLOCKS:

DETECTOR

This Data Block specifies the types and locations of the DETECTOR(s) in the problem. A detector consists of a specified geometrical space with an associated detector response to scoring particles. The output of a COG run is largely the score for each detector.

ALPHATRANSPORT

This Data Block is used to provide parameters needed to do alpha transport.

DEUTTRANSPORT

This Data Block is used to provide parameters needed to do deuteron transport.

ANALYSIS

This Data Block is used to generate a pictorial breakdown of where events are occurring in the problem's phase space. This can be very helpful in understanding the relative importances of various source-detector paths to the detector results. Analysis pictures are described under the **DETECTOR Data Block** section.

WALK

The WALK Data Blocks are used to modify (bias) the basic analog Monte Carlo random walk to make particle transport more efficient. Biasing can be done by spatial region, particle type, and energy.

I/O

The I/O (Input/Output) Data Block allows the user to request the output of additional problem information, beyond what is normally provided by COG. This includes text files and plots of the cross sections for the isotopes and materials in the problem.

RESTART

This Data Block allows the user to restart a problem that was terminated early, or to continue a run for a longer time to get more precise results.

CRITICALITY

This Data Block specifies the starting neutron source for a criticality calculation.

DUMP

The DUMP Data Block allows you to take RESTARTable DUMPS of your Criticality problem while COG is running.

PARALLEL

This Data Block enables concurrent processing of a COG job on a network of Unix workstations, or an MPP.

EGS

The EGS Block specifies parameters needed to transport electrons, using the EGS5 kernel.

ACTIVATION

This Data Block specifies a set of times at which photons produced by neutron activation are to be scored.

BFIELD

The BFIELD Block is used to specify vacuum magnetic field sectors through which protons will be transported.

1.7.4 Input Notation Shortcuts

COG offers some notational shortcuts to speed input of repeated or interpolated lists of numbers.

To *repeat* a number or a group of numbers use this form:

n [m1 m2 m3]

to generate **n** instances of the numbers **m1 m2 m3** enclosed in the brackets.

For example, 3 [1.1 2.2] will produce the equivalent of entering:

1.1 2.2 1.1 2.2 1.1 2.2

To *linearly interpolate* values between given end values, use this form:

n [m1 i m2]

to generate **n** linearly interpolated values, starting at **m1** and ending at **m2**.

For example, 5 [1. i 5.] will produce the equivalent of entering:

1. 2. 3. 4. 5.

To *logarithmically interpolate* values between given end values, use this form:

n [m1 L m2]

to generate **n** logarithmically interpolated values, starting at m1 and ending at m2.

For example, 5 [1. L 5.] will produce the equivalent of entering:

1.00 1.49 2.24 3.34 5.00

Note: Delimiting spaces are not needed around the brackets. The letter L can be in upper or lower case but is less likely to be confused with the number one when capitalized.

1.7.5 Manual Notation Conventions

In the listing of COG input line forms that follows, the manual makes use of certain notation conventions. Their syntax is presented here.

- Words shown in the manual as uppercase letters are Data Block names and keywords. (Note: capitalization is optional for the user—the code converts everything [except titles and filenames] to lowercase before parsing the input.) COG expects the data after the keywords to occur in a prescribed order. Data entry for a given keyword is concluded by the next keyword.

Example of the different acceptable keyword forms:

```
SURFACES
surfaces
```

- Words shown in the manual as lowercase letters indicate number fields that the user must supply. Dashes between words are used to indicate that only one number is to be typed (i.e., inner-radius).

Example: a manual specification of the form:

```
surf-ID# PLANE x1 y1 z1 x2 y2 z2
would be satisfied by this user line:
10 PLANE 1. 2. 3. 1. 3. 3.
```

- **Optional** inputs appear inside curled brackets { } or parenthesis (). Do **not** include the curled brackets or parenthesis in your input.
- **Required** inputs (all else) must be provided. (Square brackets, [], in the PULSE detector definition and shorthand BIN definitions are **required**.)

Example: The manual specifies an input line of the form:

```
surf-ID# SPHERE r {TR...}
```

where:

SPHERE (or **sphere**) is a keyword for a spherical surface;

surf-ID# requires an ID number (an integer);

radius **r** requires a number;

but the Translate/Rotate field **{TR...}** is optional.

- When items appear in the manual as a bracketed stack, the user can choose any **one** of the choices offered inside the stack.

Example of a bracketed stack:

**[SPHERE
SPH
S]**

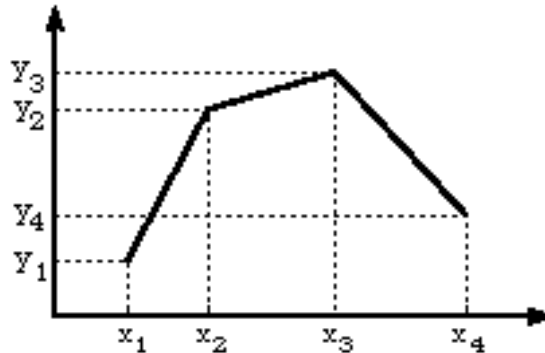
(The square brackets in the bracketed stack, [], are never typed.)

A user may choose SPHERE for easier reading. After some experience, SPH may be preferable; and for a lot of input, the letter S may be sufficient. All three definitions can be used in the same input file.

Examples of some definitions showing required and optional inputs:

```
CIRCLE X1 Y1 Z1 X2 Y2 Z2 R {X3 Y3 Z3 PHIMIN PHIMAX}
ss-disk x1 y1 z1 x2 y2 z2 r0 {rin}
      {x3 y3 z3 phimin phimax}
```

- In various places, the user may want to specify a curve, such as a source spectrum. This may be done by providing enough (x,y) points so that a straight-line approximation between the points adequately represents the desired spectrum.



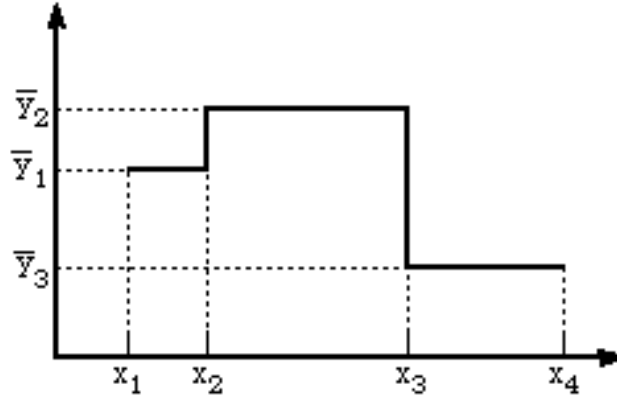
The points are given as x y pairs i.e.

$$x_1 y_1 \quad x_2 y_2 \quad x_3 y_3 \quad x_4 y_4$$

The approximation assumes that y is zero for x outside the range of values given. The x values may be given in either increasing or decreasing order. This type of input is represented throughout the manual as:

$$(x,y)_i$$

Alternatively, the user may want to specify a set of BINS for the input of data or the output of calculated results.



Here the $x, \bar{y}$ points specify bin edge and amplitude (a histogram):

$$x_1 \bar{y}_1 \quad x_2 \bar{y}_2 \quad x_3 \bar{y}_3 \quad x_4$$

Again, y is assumed to be zero for x outside the range of values given. The values of x may be either increasing or decreasing.

Note: One more x value (x_4) than y value **must** be given. This input type is represented in the manual by

$$(x, \bar{y})_i$$

- To avoid wearisome repetition in the manual of entire option fields, we often indicate these input fields by the ellipsis (...). The full field definitions are given at the start of each section. For example, the translate/rotate (TR) option for surfaces is defined as:

$$\text{TR } x_0 \ y_0 \ z_0 \ (x_1 \ y_1 \ z_1 \ x_2 \ y_2 \ z_2)$$

However, after it has been discussed in the text the TR option is often represented more simply by:

$$(\text{TR } \dots)$$

1.7.6 Summary of COG Data Blocks

Data Block	What it does
TITLE – required	One-line description of problem
BASIC – required	Defines length, energy and time units, type of particles being transported and physics options
SURFACES – required	Defines surfaces used in geometrical description of problem
GEOMETRY – required	Defines the geometry (bounded volumes) of the problem
MIX – required	Specifies materials, densities, and cross-section libraries to be used
ASSIGN – optional	Assigns material, region, and density to a sector
ASSIGN-M – optional	Assigns a Material to a sector ASSIGN –
ML – optional	Assigns a Material to a List of sectors
ASSIGN-R – optional	Assigns a Region number to a sector
ASSIGN-RL – optional	Assigns a Region number to a List of sectors
ASSIGN-D – optional	Assigns Density to a sector
ASSIGN-MD – optional	Assigns Material and Density to a sector
ASSIGN-MC – optional	Assigns Materials to plotting Colors
SOURCE – required for all non-criticality calculations	Defines energy, location, time, angular distribution, and particle type of source
DETECTOR – optional	Defines the type of Detector(s) for scoring results.
WALK – optional	Modifies the default analog random walk to improve run efficiency (biasing)
CRITICALITY – required for nuclear criticality problems	Defines criticality parameters and initial neutron source
DUMP – optional	Creates restartable dumps for criticality jobs
ALPHATRANSPORT –optional	Provides parameters for alpha transport
DEUTTRANSPORT –optional	Provides parameters for deuteron transport
ANALYSIS – optional	Plots collision densities in selected phase-space regions
I/O – optional	Prints and plots cross sections
RESTART – optional	Restarts a COG problem to run longer
PARALLEL – optional	Allows concurrent processing on a network of workstations or on an MPP
EGS – optional	Specifies parameters for electron transport
ACTIVATION – optional	Specifies detector times at which photons from neutron activations are scored
BFIELD – optional	Specifies sectors containing vacuum magnetic fields, for proton transport
END – required	Terminates reading of input file

1.7.7 Catching Input Errors

COG flags errors it detects during the input phase. If fatal errors are found, COG stops at this point. Output errors are listed in the**out** file. To speed code execution, automatic geometry error checking is ***not*** done during the normal random-walk process. COG only checks for geometry errors during the production of optional PICTURES and SWEEPS and VOLUME calculations, which are specified by the user in the GEOMETRY Data Block. The user should always debug a new problem by making pictures to catch errors in defining the geometry. These cross-section or perspective pictures also enable the user to visually check the geometry specifications. The cross-section pictures in particular are very helpful in verifying that the SURFACES and SECTORS specified by the input file are correct. While generating these pictures, COG checks that all points in the view fall in one and only one defined sector. But the picture-making process can only check the sectors contained in the specified view. It is therefore recommended that several cross-sectional views be drawn through all important or complex geometry.

After the geometry has been verified, the pictures can be turned off to save time. This is done by prefacing the picture lines with a \$ sign (this turns them into comments). They can be quickly reactivated (by deleting the \$) if the geometry is changed and needs to be rechecked. The SWEEP option in the GEOMETRY Data Block will sweep a line between specified endpoints and print out the locations of surface and sector intersections of the line. This can be useful in verifying exact placement of problem surfaces.

1.7.8 Sample COG Problem

Suppose we have this problem to model:

Example: A spherical volume source "sp1" emits gammas isotropically. At three centimeters from the sphere center is a lead shield "sh1". Six centimeters from the source is a solid-state silicon detector "det1".

Our geometry input would look like this:

```

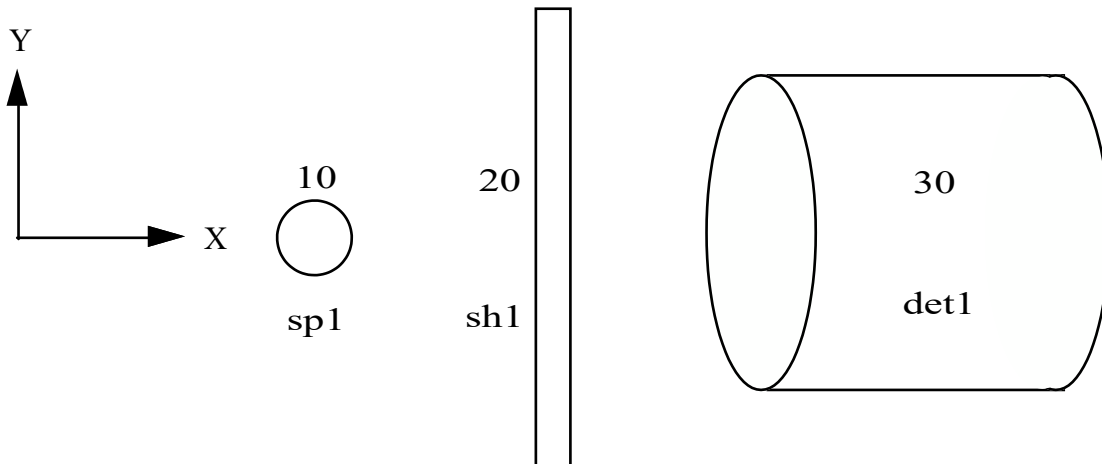
SURFACES
10 SPHERE 0.5  TR 1. 0. 0.  $ SPHERICAL VOLUME SOURCE
                             $ OF RADIUS 0.5, TRANSLATED
                             $ SO CENTER IS AT (1,0,0)

20 BOX 0.5 6. 6. TR 4.25 0. 0.
                             $ BOX WITH SIDES 0.5,6,6
                             $ TRANSLATED SO SIDE
                             $ FACING SOURCE IS AT X=4.0

30 CYLINDER 2.  6. 10.      $ CYLINDER WITH RADIUS 2.0,
                             $ EXTENDING FROM X=6.0 TO X=10.

GEOMETRY
SECTOR 1 SP1 -10            $ SOURCE IS VOLUME INSIDE
                             $ SURFACE 10,  A SPHERE
SECTOR 2 SH1 -20            $ SHIELD IS VOLUME INSIDE
                             $ PSUEDOSURFACE 20, A BOX
SECTOR 3 DET1 -30           $ DETECTOR IS VOLUME INSIDE SURFACE
                             $ 30, A FINITE CYLINDER

```



At this point we have specified all the volumes of interest to us in this problem (unspecified volumes are assumed by COG to contain vacuum). Now we need to specify the materials in the problem.

Let our source be ^{60}Co , the shield be lead, and the detector silicon. We will model the 1.17 MeV and the 1.33 MeV lines of ^{60}Co .

Our MIX input is:

```
MIX
MAT=1 CO 8.85          $ ELEMENT AND DENSITY
MAT=2 PB 11.4
MAT=3 SI 2.33
```

Now we need to ASSIGN materials to SECTORS.

```
ASSIGN-ML
1 1 / 2 2 / 3 3 $ ASSIGN MATERIAL 1 (CO) TO SECTOR 1,
                  $ MATERIAL 2 (PB) TO SECTOR 2, ETC.
                  $ BY DEFAULT, MATERIAL # = SECTOR #,
                  $ SO THIS STATEMENT NOT REALLY NEEDED
```

Specify the SOURCE location, and energy-, angle-, and time-dependences

```
SOURCE
NPARTICLES = 10000      $ WANT TO RUN 10,000 GAMMAS
DEFINE POSITION = 1      $ POSITION SPECIFICATION #1
SPHERE 1. 0. 0. 0.5     $ SPHERICAL VOLUME SOURCE, CENTER
                        $ AT (1,0,0), R = 0.5
DEFINE ENERGY = 1      $ ENERGY SPECIFICATION #1
PHOTON
LINE 1.17 1. 1.33 1.    $ EMIT LINE SPECTRUM AT 1.17
                        $ AND 1.33 MEV, WITH
                        $ UNIT INTENSITIES
DEFINE ANGLE = 1        $ ANGLE SPECIFICATION #1
ISOTROPIC               $ EMIT ISOTROPICALLY
INCREMENT 1 P=1 E=1 A=1 $ SOURCE INCREMENT HAS SOURCE
                        $ STRENGTH=1, AND THESE
                        $ SPECIFIED POSITION, ENERGY,
                        $ AND ANGLE DEFINITIONS
```

Specify the DETECTOR

```

DETECTOR
NUMBER = 1
TITLE = "REACTION DETECTOR FOR CO60 PROBLEM"
REACTION 3 50.26      $ DETECTOR IS REGION 3,
                        $ DETECTOR VOLUME = 50.26 CM3

```

The complete COG input file for this problem is:

```

COG PROBLEM #1          $ PROBLEM TITLE LINE

BASIC                   $ BASIC DATA BLOCK
  PHOTON                $ TRACK PHOTONS ONLY
  CM                    $ DIMENSIONS ARE CENTIMETERS

SURFACES
10 SPHERE 0.5  TR 1. 0. 0. $ SPHERE CENTERED AT
                        $ (1,0,0)
20 BOX 0.5 6. 6. TR 4.25 0. 0. $ BOX WITH SIDES 0.5,6,6
                        $ TRANSLATED SO SIDE
                        $ FACING SOURCE IS AT X=4.0
30 CYLINDER 2. 6. 10. $ CYLINDER WITH RADIUS 2.0,
                        $ EXTENDING FROM X =6.0 TO X=10.

GEOMETRY
SECTOR 1 SP1 -10 $ SOURCE IS VOLUME INSIDE
                $ SURFACE 10, A SPHERE
SECTOR 2 SH1 -20 $ SHIELD IS VOLUME INSIDE PSUEDOSURFACE
                $ 20, A BOX
SECTOR 3 DET1 -30 $ DETECTOR IS VOLUME INSIDE SURFACE
                $ 30, A FINITE CYLINDER

MIX
  MAT=1 CO 8.85      $ ELEMENT AND DENSITY
  MAT=2 PB 11.4
  MAT=3 SI 2.33

ASSIGN-ML
  1 1 / 2 2 / 3 3    $ ASSIGN MATERIAL 1 (CO) TO
                    $ SECTOR 1, MATERIAL 2 (PB) TO
                    $ SECTOR 2, ETC.

SOURCE
NPARTICLES = 10000    $ WANT TO RUN 10,000 GAMMAS
DEFINE POSITION = 1     $ POSITION SPECIFICATION #1
  SPHERE 1. 0. 0. 0.5 $ SPHERICAL VOLUME SOURCE,
CENTER                $ AT (1,0,0), RADIUS = 0.5
DEFINE ENERGY = 1     $ ENERGY SPECIFICATION #1

```

```
PHOTON
LINE 1.17 1. 1.33 1. $ EMIT LINE SPECTRUM AT 1.17
                        $ AND 1.33 MEV, WITH
                        $ UNIT INTENSITIES
DEFINE ANGLE = 1      $ ANGLE SPECIFICATION #1
      ISOTROPIC        $ EMIT ISOTROPICALLY
INCREMENT 1 P=1 E=1 A=1 $ SOURCE INCREMENT HAS SOURCE
                        $ STRENGTH=1, AND THESE
                        $ SPECIFIED POSITION, ENERGY,
                        $ AND ANGLE DEFINITIONS

DETECTOR
NUMBER = 1

TITLE = "REACTION DETECTOR FOR CO60 PROBLEM"
REACTION 3 50.26      $ DETECTOR IS REGION 3,
                        $ DETECTOR VOLUME = 50.26 CM3
END                   $ END OF COG INPUT
```

1.8 References

- 1.0 H. Hirayama, Y. Namito, A.F. Bielajew, S.J. Wilderman and W. R. Nelson, The EGS5 Code System, SLAC-R-730 (2005) and KEK Report 2005-8 (2005), Stanford University, (January, 2016). Internet Address: <https://rcwww.kek.jp/research/egs/egs5.html>
- 2.0 R. J. Howerton, R. E. Dye, and S. T. Perkins, Evaluated Nuclear Data Library, Lawrence Livermore National Laboratory, Livermore, CA, UCRL-50400, Vol. 4, Rev. 1, (1981).
- 3.0 The Cross Section Evaluation Working Group, Data Formats and Procedures for the Evaluated Nuclear Data File ENDF-6, Report BNL-NCS-44945 (ENDF-102) edited by V.McLane, et al., National Nuclear Data Center, Brookhaven National Laboratory, U.S.A, (1995).
- 4.0 Dermott Cullen, POINT2021: A Temperature Dependent ENDF/B-VIII.0, Release 7 Cross Section Library, Lawrence Livermore National Laboratory, UCRL-ID-127776, Rev. 1, (November, 2000). <https://www-nds.iaea.org/point/>
- 5.0 R. E. MacFarlane, New thermal neutron scattering files for ENDF/B-VI release 2", report LA-12639-MS (ENDF-356), Los Alamos National Laboratory, March 1994.
- 6.0 Dermott E. Cullen, John H. Hubbell, Lynn Kissel, EPDL97: the Evaluated Photon Data Library, Lawrence Livermore National Laboratory, UCRL-50400, Vol. 6, Rev. 5, (September, 1997).
- 7.0 IAEA Photonuclear Data Library, Recommended Data, Internet Address: <https://www.iaea.org/resources/databases/iaea-photonuclear-data-library>
- 8.0 M. A. Gardner and R. J. Howerton, ACTL: Evaluated Neutron Activation Cross Section Library—Evaluation Techniques and Reaction Index, Lawrence Livermore National Laboratory, Livermore, CA, UCRL-50400, vol 18, (1978).
- 9.0 L. L. Carter and E. D. Cashwell, Particle–Transport Simulation with the Monte Carlo Method, Los Alamos National Laboratory, Los Alamos, NM, TID-26607, (1975).

- 10.0 I. Lux and L. Koblinger, Monte Carlo Particle Transport Methods: Neutron and Photon Calculations, (CRC Press, Boston, MA, 1990).
- 11.0 PGPLOT is a copyrighted Graphics Subroutine Library developed by the California Institute of Technology, 1200 East California Blvd., Pasadena, CA. Internet Address: <https://sites.astro.caltech.edu/~tjp/pgplot/>
- 12.0 L. Peralta and A. Louro, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Volume 737, 11 February 2014, Pages 163–169
- 13.0 Stopping-power and range tables for helium ions, <https://physics.nist.gov/PhysRefData/Star/Text/ASTAR.html>
- 14.0 W. B. Wilson, M. Bozian, and R. T. Perry, LA-UR-88-1976
- 15.0 E. M. Gunnerson and G. James, *On the efficiency of the reaction $H^3(d,n)He^4$ in titanium tritide bombarded with deuterons*, Nuclear Instruments and Methods **8** (1960), Pages 173-184
- 16.0 A. Milocco and A. Trkov. Modelling of the Production of Source Neutrons from Low-Voltage Accelerated Deuterons on Titanium-Tritium Targets. Science and Technology of Nuclear Installations (Hindawi Publ. Corp.), v. 2008, Article ID 340282, pp 1–7
- 17.0 W. Gropp, E. Lusk, and A. Skjellum, Using MPI: Portable Parallel Programming with the Message-Passing, Interface, MIT Press, Cambridge, Massachusetts, 1994. (2nd ed. 1999). Internet Address: <https://www.mpich.org>

2 BASIC Data Block

The BASIC Data Block tells the code what units of measurement are used in the problem, what particles are being followed, and turns on certain code switches and physics options as desired. The form of the BASIC Data Block is:

BASIC *param1 param2 ...*

Where:

BASIC is the keyword indicating the start of the BASIC Data Block;

param1 param2 ... are the various Basic Data Block parameters, as described below. The only required BASIC parameter is the **particle-type**.

Problem units will default to centimeters, MeV and seconds unless otherwise specified in the BASIC Data Block.

Examples of BASIC data input:

```
BASIC PHOTON CM MEV
basic photon mm keV
```

2.1 Particle Types to be Followed

particle-type(s) to be followed is the only information that the user **must** supply in the BASIC Data Block. All other required parameters have default values inserted by the code (see **COG Units of Distance, Energy, and Time**).

Acceptable *particle-type* names are (capitalization not required):

```
NEUTRON
DELAYEDNEUTRONS
PHOTON or GAMMA or GAMMA-RAY or X-RAY
DELAYEDPHOTONS or DELAYEDGAMMAS
PHOTONUCLEAR
ELECTRON
PROTON
DEUTERON
ALPHA
```

A problem in which **only** neutrons are to be transported would include the single keyword NEUTRON. If gammas produced from neutron reactions are also to be followed, both NEUTRON and GAMMA keywords **must** be specified.

If a reaction during transport creates particles of a type not mentioned in the BASIC Data Block, the energy of these secondaries is immediately deposited. E.g., in the absence of the ELECTRON type, all electron and positron particle energy is treated as if it were deposited instantly at the scattering site.

2.2 COG Units of Distance, Energy, and Time

In most Monte Carlo codes, only one system of units is allowed. To avoid making the user spend much time converting his inputs to “standard” units, COG allows the user to specify desired units of distance, energy, and time to be used throughout the COG problem.

Default COG units are: CENTIMETERS, MEV, and SECONDS. Units used are listed in the COG printout. To use a different unit of length, one of the following may be added to the BASIC Data Block inputs:

ANGSTROM or AU or A
MILLIMICRON
MICRON
MILLIMETER or MM
CENTIMETER or CM
METER or M
KILOMETER or KM
MIL or MILS
INCH or INCHES or IN
FOOT or FEET or FT
YARD or YD
MILE-STATUTE
MILE-INTERNATIONAL-NAUTICAL

Note that this definition of units applies *throughout* the problem—thus, if the unit of length is set to be a meter, the unit of area in a detector definition must be given in square meters.

Note: Calculated particle fluences are *always* reported in units of particles/cm², regardless of BASIC Data Block settings.

The unit of energy may be set to any of the following units by specifying the appropriate name in the BASIC Data Block:

GEV
MEV
KEV
EV
JOULE
ERG
JERK

Note: Calculated particle energies are always reported in units of MeV, regardless of BASIC Data Block settings.

Likewise, the units of time may be set to any of the following units by specifying the appropriate name in the BASIC Data Block:

NANOSECOND or NS
 SHAKE or SH
 MICROSECOND
 MILLISECOND or MS
 SECOND or SEC or S
 MINUTE or MIN
 HOUR or HR or H
 DAY or D
 MONTH or MO
 YEAR or YR

This time unit definition only affects the particle lifetime units or time masks and *does not affect other time definitions*, e.g., the problem maximum running time limit set by the **TMAX** option is always in minutes.

Note: Calculated particle times are always reported in units of seconds, regardless of BASIC Data Block settings.

Internal to COG, units used are time in seconds, energy in MeV, distance in user-specified units.

2.3 BASIC Data Block Switches and Physics Models

A number of options are allowed in the BASIC Data Block to set code switches and select physics models.

2.4 Maximum Running Time – TMAX

To set a maximum time (CPU time) the problem will transport particles, the TMAX option may be added to the BASIC Data Block :

TMAX = *max-time* (in minutes)

where *max-time* is the maximum running time in minutes.

TMAX does not affect the running time of parallel jobs.

Note: TMAX's units (minutes) are not affected by the BASIC time-unit setting. The BASIC time-unit setting applies only to particles.

Default: TMAX = 1.0E+8 minutes (~190 years) , which effectively turns this option off.

2.5 Time Interval Between Restart Dumps – TDUMP

TDUMP sets the time interval (CPU minutes) between successive RESTART dumps made by COG, as it runs your problem. See the preceding section **RESTART** –

Running COG Longer for a description of this option. Default is 60 minutes. A final restart dump is made at the end of the problem, unless the **NOFINALRESTART** option is used.

$$\text{TDUMP} = \text{dump-interval-time} \quad (\text{in CPU minutes})$$

Note: TDUMP's units (minutes) are not affected by the BASIC time-unit setting. The BASIC time-unit setting applies only to particles.

Default: TDUMP = 60 minutes .

2.6 Particle Interval Between Restart Dumps – NDUMP

As an alternative to TDUMP, NDUMP allows the making of a family of RESTART files based on the number of source particles run.

$$\text{NDUMP} = \text{npartdump} \quad (\text{in an integer number})$$

will produce a RESTART file every *npartdump* source particles. These restart files have names of form: *input.rsnnnnnnnnnnnnnn*, where the 11 x-s are the zero-filled source particle number. See the preceding section **RESTART – Running COG Longer** for a description of this option. There is no default value for *npartdump* – it must be specified. .

2.7 NOFINALRESTART

As noted above, COG will make a final restart dump at the end of the problem run, unless the option NOFINALRESTART is specified in the **BASIC** Data Block.

NOFINALRESTART

2.8 Starting Random Numbers – RN

Monte Carlo codes use pseudo-random-number sequences to determine what will happen to a particle in the course of the random walk. The initial state of the pseudo-random-number generator is determined by the starting seeds (integer numbers). The exact same sequence of random numbers is replicated by generating another sequence using the same starting seeds. Therefore, if two identical COG problems are run with the same starting seeds, both will produce identical results. This is not normally desirable, so COG starts each problem with seeds derived from the internal clock present on all computers. For nearly all purposes, the initial seeds are randomly chosen. These starting seeds are printed in the COG output file.

COG now uses the lagged-Fibonacci sequence Random Number Generator devised by Marsaglia, Zaman, and Tsang.¹ Each call to the RNG returns one real*4 floating-point number in the range [0,1]. The lagged-Fibonacci pseudo-random-number sequence is determined by two starting seeds. An independent random-number sequence is generated for every unique pair of starting seeds. Acceptable values for

first seed are 1 to 31328. Acceptable values for second seed are 1 to 30081. The sequence length (the number of values generated before the sequence starts to repeat) is 2^{144} or $\sim 2 \times 10^{43}$. The state of the RNG is specified by an array of ~ 100 real*4 generator values.

When COG is run in Parallel Mode, each slave COG process is given its own pair of starting seeds. The uniqueness of the sequence generated from each pair of seeds, and the great length of each sequence before it repeats, guarantee that each slave calculation is statistically independent of all others. This allows the results from all slaves to be validly summed, and a valid error estimate to be computed.

On some occasions, for debugging a user job or particle RETRACEing, it is desirable to exactly duplicate a previous COG run. To do this, you must specify the values of the two starting seeds within the BASIC Data Block as follows:

RN = *rn1 rn2 (seqno)*

where:

rn1 and ***rn2*** are the two starting seeds you wish to use to start the sequence.

rn1 is an integer in the range [1,31328].

rn2 is an integer in the range [1,30081].

seqno is the ordinal number in the sequence of the first random number to be used by COG.

Default: New random number seeds are used to start each run. The default value of ***seqno*** is 1.

2.9 Electron-Positron Annihilation Events – ANNIH–GAMMAS

When an electron-positron pair-production event occurs, and the user has not specified electron transport for the current region, then COG immediately annihilates the pair to create gammas. In the interest of speed, normally just one gamma is produced and followed (statistically, we get the same result as if two were followed, because this one gamma is given twice the usual weight). If the user has a PULSE type detector, then he may be simulating coincidence detection of the annihilation gamma pair, and COG creates and follows two gammas. In general, the user may specify the number of gammas, *ngammas*, to be created and followed by using this line in the BASIC Data Block:

ANNIH–GAMMAS *ngammas*

Where ***ngammas*** may be 0, 1, or 2.

Example of requiring the code to follow both pair-production gammas.

BASIC ANNIH–GAMMAS 2

Note: This specification will not override the COG choices for the case of PULSE-type detector ($ngammas = 2$) or EGS transport ($ngammas = 0$).

Default: $ngammas = 1$ (if electron transport not enabled for this region).

$ngammas = 0$ (if electron transport is enabled for this region).

2.10 Using Crit Detector Variance Reduction (Hybrid Mode) – CRITDETVR

The Crit Detector Variance Reduction mode was introduced to allow VR methods to be employed in a criticality eigenvalue problem without biasing in any way the outcome of the K calculation. COG interleaves crit batches with shielding cycles in such a way that each shielding cycle transports the source neutrons generated by the preceding criticality batch.

There is no “feedback” from the shielding cycle to the eigenvalue calculation, so the computed K is an unbiased result. Each shielding cycle treats the last computed crit source as a sample of the problem’s “settled source” and can employ any of the WALK-XX random-walk VR methods available to enhance scoring statistics at the problem’s detectors.

Simply include the following statement in the BASIC Data Block:

CRITDETVR

To modify the random walk during shielding cycles, add a WALK-XX block to your input file (see the section on Random Walk Modification Techniques).

2.11 PHOTONUCLEAR Including Photonuclear Reactions

Photonuclear reactions are not contained in the available photon libraries (EPDL89, EPDL97, EPIC2017, and EPIC2023), so generally are not included in a coupled neutron-photon problem. To allow photonuclear reactions in the calculation, include the following statement in the

BASIC Data Block:

PHOTONUCLEAR

This causes COG to load photonuclear reaction data from the IAEA Photonuclear Data Library and proceed as described in the **Overview of Photon Transport**, above.

2.12 Including Delayed Neutrons in Fission Reactions – DELAYEDNEUTRONS

The fission process produces most of its neutrons instantaneously, but a few delayed neutrons are released up to minutes after the initial fission event. Normally, only the instantaneous (prompt) neutrons need be followed. However, in problems involving steady-state reactor operation, the delayed neutrons should also be included. To include these delayed neutrons in the calculation, include the following statement in the BASIC Data Block:

DELAYEDNEUTRONS

Note: To get a realistic simulation of the delayed neutrons, the user must specify the ENDFB6 library. Alternative COG neutron libraries (e.g., the ENDL database) do not contain the data required to specify the delayed neutron spectra and emission times. If these libraries are used, the only effect of specifying delayed neutrons is to increase the nubar parameter for the prompt fission neutrons.

2.13 Delayed Fission Gammas

The DFG option requires 2 entries in the COG input file.

1) In the **Mix Block**:

DGLIB = *libname*

where *libname* is one of the DFG libraries currently available.

2) In the **Basic Block**:

DELAYEDPHOTONS *t1 t2*

or

DELAYEDGAMMAS *t1 t2*

where *t1* and *t2* are times in seconds. Only delayed fission gammas born between *t1* and *t2* are tracked and scored.

In the detector output, in addition to the total photon output, we now have both a prompt and delayed (between *t1* and *t2*) component. Also included is the delayed fission gamma rate at *t2*.

Example: Include

DGLIB = DFG.ENDFB7R1

in the **Mix Block** and

DELAYEDPHOTONS 60. 120.

in the **Basic Block**. Then the detector, assuming a flux to dose response function, would yield the total dose, the prompt dose, and the 60 second (1 minute) accumulated dose at 60 seconds (1 minute). Also included is the delayed fission gamma rate (dose/sec) at 120 seconds (2 minutes).

Below is a sample output.

particle	response	std. dev.	fsd	fom
photon	8.2213E-16	8.7501E-18	1.0643E-02	1.0406E+03
prompt	7.8214E-16	8.6806E-18		
delayed	3.9982E-17	4.9637E-19	(6.0E+01 to 1.20E+02 sec)	
dfg rate	4.6751E-19	5.8112E-21	(at 1.20E+02 sec)	

2.14 Number of Secondary Particles Exiting a Collision— NEUTRONPRODUCTION and PHOTONPRODUCTION

The number of photons and neutrons exiting a particle collision is termed the multiplicity of the reaction. In the default mode, COG produces as many secondary neutrons as the multiplicity of the reaction calls for (e.g., (n,2n) produces two exiting neutrons). If the result of a neutron collision is a fission event, COG emits on average “nubar” secondary neutrons, whose energies are sampled from the fission spectrum of the isotope struck. For example, if nubar is 3.5, COG will emit either 3 or 4 prompt neutrons.

For photons, the default is that COG will track just a single photon exiting a collision regardless of the multiplicity (its weight is adjusted to yield unbiased scores). This single-photon procedure is the fastest-running option, and usually gives good results.

Optionally, the user can specify to COG the number of exiting secondaries to follow, for neutron and/or photon production reactions. Including this line in the BASIC Data Block:

NEUTRONPRODUCTION $\left[\begin{array}{c} SINGLE \\ MULTIPLE \end{array} \right]$

.

will cause COG to follow just one neutron (**SINGLE**) or the number of neutrons specified by the reaction multiplicity (**MULTIPLE**).

Neutron Default: MULTIPLE

Including this line in the BASIC Data Block:

PHOTONPRODUCTION $\left[\begin{array}{c} SINGLE \\ MULTIPLE \end{array} \right]$

.

will cause COG to follow just one photon (**SINGLE**) or the number of photons specified by the reaction multiplicity (**MULTIPLE**).

Photon Default: SINGLE

When **SINGLE** is chosen, fewer secondaries will be produced, and COG will run faster to complete a fixed number of source particles. Whether this feature decreases or increases the running time required to achieve a satisfactory answer depends on the problem.

2.15 Fission-Produced Neutrons and Photons – NOFISSION and MAXFISSION

If the entry:

NOFISSION

is included in the BASIC Data Block, no neutrons or photons will be produced in a fission event. This option is useful for a shielding calculation involving a critical or supercritical system, if one wishes to avoid an exponential increase in the number of particles to be tracked. If a COG shielding job has a neutron source and a near-critical mass of fissile material, very long neutron multiplication chains may be produced. COG may take so much time tracking these secondary neutrons that the job never completes in any reasonable time. You can set an upper limit on the number of secondary neutrons you are willing to track from any source neutron, by including this line in the BASIC Data Block:

MAXFISSION *MaxFissNo*

where *MaxFissNo* is the maximum number of fission events per source neutron to be tracked from any source neutron. If this limit is exceeded, tracking of the source particle and its progeny stops, but all events up to this point are scored in a normal way. A warning will be printed.

2.16 Probability Of Initiation – POI

Including

POI

in the **BASIC** Data Block enables a probability of initiation calculation. POI is calculated as the ratio of number of source neutrons resulting in an infinite chain (e.g., having more than *MaxFissNo* fission neutrons) to total number of neutrons.

2.17 Using Unresolved Resonance Region Probability Tables – URRPT

Including

URRPT

in the **BASIC** Data Block enables the use of Unresolved Resonance Region Probability Tables to do self-shielding calculations.

2.18 Using Fission Reaction Event Yield Algorithm – FREYA

FREYA provides additional fission physics including incident neutron-dependent and multiplicity-dependent spectra with angular correlations based on detailed sampling of the light and heavy fission fragments with pre-equilibrium neutron emission. Simply include the following statement in the BASICData Block:

FREYA**2.19 Nuclear Resonance Fluorescence –NRF**

NRF reactions are the result of nuclear absorption and emission of photons. When a near resonance energy photon strikes the nucleus, the nucleus becomes excited and subsequently decays to its ground state, releasing one or more discrete energy photons. The data needed to do nuclear resonance fluorescence calculations in COG are contained in the COG library COGNRF and consists of 8 elements, 15 isotopes, and 22 lines (7Li, 89Y, 140Ce, 142Ce, 142Nd, 144Nd, 152Sm, 152Sm, 154Sm, 154Gd, 156Gd, 156Gd, 156Gd, 156Gd, 158Gd, 158Gd, 160Gd, 235U, 238U, 238U, and 239Pu). The data provided, by Dr. James Hall of LLNL, are cross sections, branching ratios, and emission energies.

The NRF option is invoked by including

NRF

in the **BASIC** block. With the NRF option on, at each photon event COG determines the quantity ProbNRF. ProbNRF is zero unless the event occurs in one of the NRF isotopes at an energy near the resonance energy, then ProbNRF is the ratio of the NRF cross section to the sum of the usual photon cross section plus the NRF cross section. Once a photon event occurs, the fraction ProbNRF has an NRF event, while the fraction $1 - \text{ProbNRF}$ has a usual photon event.

2.20 Limiting the fraction of (n,X γ) events processed – FRACNXG

Include

FRACNXG *FracNXG*

processed. *FracNXG* defaults to 0.1. In the **BASIC** data Block to set *FracNXG*, the fraction of (n,X γ) events which will be

2.21 Writing Production Data Files– EPFILE and PHPFILE

COG can write files containing electron or photon production data for each problem region. These files could then be used as input to other codes.

To write electron production data from reactions in each COG region into one or more files, include this line in the BASIC Data Block:

EPFILE

To write photon production data arising from neutron reactions in each COG region into one or more files, include this line in the BASIC Data Block:

PHPFILE

2.22 Proton Physics Options

See Proton Physics Models for descriptions of these options that are available in the section number for Proton Transport.

HADRONIC PROTON MODEL *modelname representation*
[ON/OFF]

MSCAT PROTON MODEL *modelname*

DEDX PROTON [ON/OFF]

ESTRAGGLE PROTON [ON/OFF]

ESTEP PROTON *fraceval*

DNEAR [ON/OFF]

BBOUNDCHECK [ON/OFF]

2.23 BASIC Data Block Example

Example of a BASIC Data Block:

```
BASIC NEUTRON INCH NANOSECOND
```

Default: COG units are: centimeters, MeV, seconds , TMAX = 1.0E+8 (minutes)

Required: One of NEUTRON, PHOTON, ELECTRON, PROTON, DEUTERON, ALPHA

For coupled problems, must specify all particle types to be transported (e.g., NEUTRON PHOTON ELECTRON).

Optional:

TMAX *max-time* (in minutes)
TDUMP *dump-interval* (in minutes)
NDUMP *dump-interval* (in # of source particles)
RN = *rn1 rn2 seqno*
ANNIH-GAMMAS *number*
CRITDETVR
PHOTONUCLEAR
DELAYEDNEUTRONS
DELAYEDPHOTONS
DELAYEDGAMMAS
NEUTRONPRODUCTION *single / multiple*
PHOTONPRODUCTION *single / multiple*
NOFISSION
MAXFISSION *MaxFissNo*
POI
URRPT
FREYA
NRF
FRACNX
EPFILE
PHPFILE
HADRONIC PROTON MODEL
MSCAT PROTON
DEDX PROTON
ESTRAGGLE PROTON
ESTEP PROTON
DNEAR
BBOUNDCHECK

2.24 References

- 1.0 George Marsaglia, Arif Zaman, and W. W. Tsang, Toward a Universal Random Number Generator, Florida State University Supercomputer Computations Research Institute publication, FSU-CSRI-87-50.

3 The SURFACE Data Block

3.1 SURFACES and GEOMETRY—Concepts

Particle transport codes require a description of the physical spaces, or volumes, through which particles are to be transported. In code parlance, this is the problem's geometry. These physical spaces and boundaries are defined by the information in the SURFACE and GEOMETRY Data Blocks. To new Monte Carlo users, the specification of a problem's geometry can be a major and daunting effort. COG was written to make this geometry description task as simple and straightforward as possible. But for problems involving complicated physical geometry, setting up geometry even in COG will be no simple undertaking. A large part of learning the COG geometry specifications is learning what geometrical features are important to describe in detail and what features can be simplified.

Incorrectly calculated results can occur because of an oversimplified geometry. Perhaps some significant features were omitted from the problem, while others were inappropriately approximated as, for example, specifying *iron* rather than the actual *stainless steel*. In general, one must take pains to accurately model those spatial features which will contribute significantly to the DETECTOR results. To identify these important regions, it is useful to visualize the possible paths radiation may take from source to detector. If possible, look at the structure being modeled; sometimes viewing in a 3-D perspective is the only way to spot significant details.

At the other extreme, too much time can be spent putting in unnecessary detail. This wastes the time of the user putting in all this detail, and extends the running time of the calculation as the code tracks particles through a complicated geometry. Experience in running transport problems will help determine when items can be lumped together or fine detail omitted. It may be worthwhile to run a number of simple problems to evaluate the effects of problem detail on your results.

Generally speaking, two regions should be modeled in reasonable detail: the region around the source and the region around the detector. An elementary calculation shows why this is true. In a single scattering event the probability of a particle getting to a detector zone is given by $\Omega_1 P_1 \Omega_2$ where Ω_1 is the probability (solid angle) of getting to point 1, P_1 is the probability of scattering at point 1, and Ω_2 is the probability that a particle which scatters at point 1 arrives at the detector zone (at point 2). The product $\Omega_1 \Omega_2$ is a maximum for scattering near the source and near the detector (either Ω_1 or Ω_2 large) and a minimum when $\Omega_1 = \Omega_2$. Another general observation is that the volumes directly in line between the source and the detector zone should be reasonably completely described. Often the direct (unscattered) flux is the most important contribution to a background signal.

In shielding calculations, secondary sources should be identified. These are regions where relatively unattenuated flux can scatter near the detector. These scattering regions can become secondary sources, generating scattered particles which contribute heavily to the detector result. Often what needs to be modeled in reasonable detail is not the complete geometry between source and detector, but only those regions between the secondary sources and the detector.

Once the locations of any secondary sources near the detector have been identified, do a simplified calculation of their respective intensity. A spreadsheet which allows a quick and systematic solid angle calculation and first order (single scattering) transmission, can be a great aid in doing these first order calculations. Distributed sources can be approximated by breaking them down into a number of point sources. These spreadsheet calculations should at least be accurate enough to show the order of magnitude of the likely backgrounds. They are also invaluable as a reality check to see if the Monte Carlo calculation is giving reasonable answers.

Through this kind of basic problem analysis, one can avoid unnecessarily complex geometry specifications which contribute little to the accuracy of the problem solution and may cause the problem to run so long that a “good” answer is never obtained. The code can spend the vast majority of its time transporting particles into areas that do not contribute to the detector results and too little time investigating areas that do contribute significantly.

Distributed sources, multiple scattering, combined neutron and gamma-ray fields, time history of background signals, differential angular and energy distributions at a target, criticality calculations, etc., are all problems almost uniquely amenable to a Monte Carlo calculation and often almost impossible to solve accurately by other methods of analysis.

3.2 Introduction to GEOMETRY—Approaches

Setting up the GEOMETRY description and its associated SURFACE descriptions for a complex three-dimensional object is greatly facilitated by a systematic approach. The more complex the problem, the more a systematic approach pays off; but even simple problems can be set up more quickly and with fewer errors through a systematic approach. Following is one system that works for many users—feel free to modify it.

Obtain or make a sketch or drawing (by hand or with a computer drawing program) of the problem's geometry. Mechanical assembly drawings, if available, often make an excellent start. Assembly drawings may have to be supplemented by other drawings to find all of the needed dimensions. It may even require taking a sketch pad and a tape measure to the assembly and making a dimensioned sketch. If this is a simple case, a cut through the 3-D geometry may only be needed. For a moderately intricate assembly, you may need a number of sectional views or a number of separate sketches or drawings. These drawings do not necessarily have to be to scale but should show the correct relationships for each surface of interest. For a difficult geometry, a physical model is often the only way of correctly visualizing all of the possible interactions.

Analyze the geometry together with the locations and intensities of primary and secondary sources to determine what “problem” is trying to be solved. Once the problem is properly outlined, it is possible to see what geometry needs to be described and what needs to be calculated. This is an important (and sometimes overlooked) step and should be done with reasonable care if a “good” Monte Carlo calculation is to be obtained. Only when the GEOMETRY is decided upon can the necessary

SURFACES and MATERIALS be chosen to describe it.

Four different Data Blocks have to be filled out to adequately describe the physical geometry of the problem: SURFACES, GEOMETRY, MIXTURE and ASSIGN. The details of how to use these Data Blocks are explained in their respective sections. These Data Blocks are most easily completed by referring to the sketch or drawings of the geometry, perhaps supplemented by other information.

- **Step 1:** Obtain a dimensioned sketch(s) or drawing(s) of the problem's geometry. On the sketch or drawing choose a coordinate system. This same coordinate system is used throughout the COG problem. If not using centimeters, MeV and seconds units, specify other units in the BASIC Data Block.
- **Step 2:** Select a volume and its boundary surfaces, and assign a different *surf-ID#* to each surface. Label each surface in the sketch, and draw an arrow in the direction of the positive outward surface normal (see **Positive and Negative Sides of Surfaces**,

below) for each surface. Make sure each *surf-ID#* is used only once. Fill in the data for each surface in the SURFACE Data Block.

- **Step 3:** Define a SECTOR (zonal volume) by its relationship to its boundary surfaces. Assign each sector a number and write this *sect-ID#* on the sketch. Make sure each *sect-ID#* can not be confused with the *surf-ID#*. Fill in the data for each sector in the GEOMETRY Data Block.
- **Step 4:** In the MIX Data Block, specify the material which will fill this sector (if it is not already specified). Assign a material number *mat-ID#* to the material, and specify the name and specific gravity of each component of the material. Keep a list of which sectors are assigned to each material.
- **Step 5:** Cycle through steps 2 to 4 until the GEOMETRY is defined.
- **Step 6:** Use the ASSIGN-M or ASSIGN-ML Data Block to assign the material numbers to the sectors.

3.3 Numbering and Naming Surfaces

The format for the SURFACES Data Block is:

SURFACES

surf-ID# SURFACE-NAME xyz-location etc. TR (translate-rotate) ...

Where:

SURFACES is a keyword identifying the SURFACES Data Block. It is used only once per problem.

surf-ID# is the user-assigned integer which identifies this surface to COG. It must be unique in the problem, and lie in the range 1 to 9,999,999.

SURFACE-NAME is the COG surface name (e.g., CYL, SPHERE, PLANE) indicating the surface type. The list of approved mnemonic names and synonyms of the various surfaces are given in the description of each surface.

xyz-location etc. are the numerical parameters which specify the surface by giving its location, orientation, and extent. These are listed for each surface in the next section.

TR(translate-rotate) are the optional Translate-Rotate commands that can be used to translate and rotate the surface from its initially defined position into its desired location in the problem.

Example:

```
44 CYL 3.85 -1. 2.
```

indicates that *surf-ID#* 44 is a finite CYLINDER (CYL) of radius 3.85 units whose axis lies along the x-axis (by default). The cylinder is terminated by planes at $x = -1$. and $x = 2$.

```
45 CYL 3.85 -1. 2. TR 1. 2. 3. 1. 3. 3. 0. 2. 3.
```

indicates an identically-defined cylinder which has been translated (TR) to the point 1. 2. 3. and rotated so that the cylinder axis is parallel to the y-axis, the cylinder axis now passes through the points (1,2,3) and (1,3,3). The terminating planes are now at $y = 1$. and $y = 4$.

Note: You may define a surface you do not subsequently use. However, two different surfaces with the same *surf-ID#* causes a fatal error. Choose *surf-ID#*s that have some meaning to you for ease of use. Different types of surfaces, planes, cylinders, etc., could be assigned their own sequence of numbers (e.g., planes 1-20, cylinders 40-60 ...). If length units other than centimeters are used to define the surfaces, the units must be specified in the BASIC Data Block.

3.4 Positive and Negative Sides of Surfaces

SURFACES are used in COG to bound volumes (which COG terms SECTORS, and some other codes call “zones”). SECTORS are specified in COG by sector equations, which list each surface that bounds the sector, and indicate which side of that surface is the inside of the sector.

For example, the sector specification:

```
SECTOR 10 shield -10 +20 -30
```

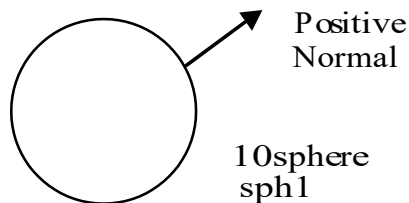
states that the sector whose *sect-ID#* = 10 (which the user labels 'shield') is bounded by the surfaces whose *surf-ID#*s are 10, 20 and 30. The inside of the sector is stated to be on the negative side of surfaces 10 and 30, and on the positive side of surface 20.

The positive side of a surface lies in the direction of the positive surface normal. The direction of the positive surface normal is determined by the code for all of the surface types except PLANE, and is shown in the manual for each surface. For closed surfaces, the positive normal points outward.

Default: Unsigned *surf-ID#*s are assumed to have a positive sign.

For example: to describe a sector whose sect-ID# = 3, named “sph1”, as the volume inside a sphere (surf-ID#=10), all that is needed is:

```
SECTOR 3 sph1 -10
```



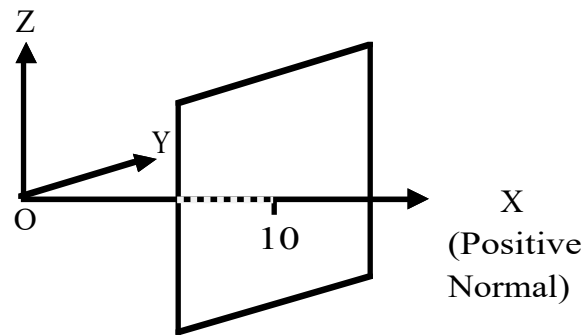
To describe a sector whose sect-ID# = 4, named “out_sph1”, as the volume outside the sphere (surf-ID#=10), all that is needed is: (+ sign optional)

```
SECTOR 4 out_sph1 +10
```

PLANE— Surfaces of Order 1:

For planes specified as normal to a coordinate axis, the positive normal points along that positive coordinate axis.

For the general plane case, the user can specify the direction of the positive normal (see PLANE definition). Normally one would want to choose this direction in some consistent way. For example, one could choose it such that the object bounded by the plane lies on the negative side of the plane, or alternatively, that the positive normal points along a positive coordinate direction.

**CYLINDER, CONE, SPHERE, PARABOLOID, etc. - Surfaces of Order 2:**

For most simple second order surfaces – such as the sphere, cylinder, or cone – the *negative* side of the surface is *inside* the volume enclosed by the surface and the *positive* side is *outside*.

BOX, etc. — Pseudosurfaces:

COG allows the use of several pseudosurfaces. These are simple surfaces pre-defined to form a commonly-used closed volume such as a BOX (a pseudosurface composed of six planes). COG defines the negative side as inside the volume enclosed by the surface and the positive side as outside.

TORUS, etc. — Surfaces of higher order:

In some rare instances of higher-order surfaces, the user may find it difficult to distinguish inside points from outside points (e.g., in a degenerate torus or hyperbolic paraboloid). You should check the definitions of such higher order figures (in this section) to ensure that your definition and the definition the code uses are the same.

NOTE: In the following section describing the surfaces, figures are shown with the *negative* surface side as *shaded*.

3.5 Finite Limits to Surfaces

Many basic mathematically-defined surfaces are unbounded in extent (PLANE, CYLINDER, CONE, etc.). To describe a finite volume, these surfaces must be limited by other surfaces. COG includes *bounding planes* for several surface types.

Example: *If the surface specified by surf-ID# = 17 is a right circular CYLINDER of radius 7.51, you would express the input definition as:*

```
17 CYLINDER 7.51
```

This defines a cylinder with its axis along the x-axis, of unbounded extent. (If no axis is specified, cylinders in COG are generated along the x-axis). If you want a finite cylinder bounded by planes normal to its axis at $x = 5.3$ and $x = 12.8$, you can add this additional information thusly:

```
17 CYLINDER 7.51 5.3 12.8
```

All points *within* the finite cylinder are on the *negative* side of the surface; all points *outside* are on the *positive* side.

In the detailed surface descriptions which follow, surfaces which the user may terminate with planes have this entry:

```
{xb1 xb2}
```

where **xb1** and **xb2** are the two x-axis end points. The planes for the ends are drawn through the points **xb1** and **xb2**, perpendicular to the x-axis. The order that the two x-values are given is not important—the surface exists at all points between the two.

3.6 Translation and Rotation (TR) of Surfaces

All surfaces are initially "born" in the surface–definition coordinate system: x', y', z' . To place a surface in an orientation or location other than its initial one, the user must add to the surface specification a TR (translation/ rotation) specification of the form:

TR $x_0 y_0 z_0$ ($x_1 y_1 z_1$ $x_2 y_2 z_2$)

- **Each COG** problem has *only one* set of coordinates that is used throughout the entire description of the problem. This is the x, y, z coordinate system.
- Each surface is initially defined in the surface–definition coordinate system x', y', z' —independent of all other surfaces and the problem coordinates. For simplicity, the user may let the x', y', z' coordinate system and the x, y, z coordinate system coincide.
- Each surface with an axis of symmetry is born with this axis coincident with the x' axis. (Exception: cylinders and planes may be born along any coordinate axis).
- The TR command "moves" (i.e., maps) the surface from the initial x', y', z' coordinate system into its desired location in the x, y, z coordinate system. If the surface definition (e.g., for cylinders) includes truncating end planes (specified by values of $\{x'b_1 x'b_2\}$), these end planes are mapped along with the surface.

This TR mapping is determined in the following way:

- The first point, $x_0 y_0 z_0$, is the origin of the x', y', z' system, given in terms of its desired COG x, y, z values. This is the *translation* part of the specification.
- The second point, $x_1 y_1 z_1$, is any point *on the positive x' -axis*, other than the origin, given in terms of its desired x, y, z values.
- The third point, $x_2 y_2 z_2$, is any point *on the positive y' -axis*, other than the origin, given in terms of its desired x, y, z values.

Because both systems are right-handed coordinate systems, it is unnecessary to specify a point on the z' -axis.

For surfaces in which the x', y', z' and x, y, z systems are coincident, all the TR specifications may be omitted. Obviously the easiest problem description is when x', y', z' coordinates and the x, y, z coordinates are the same. Indeed an attempt should be made to align any major symmetries in the problem along one of the coordinate axes.

In the following definitions of surface specifications, the full statement of the surface TR:

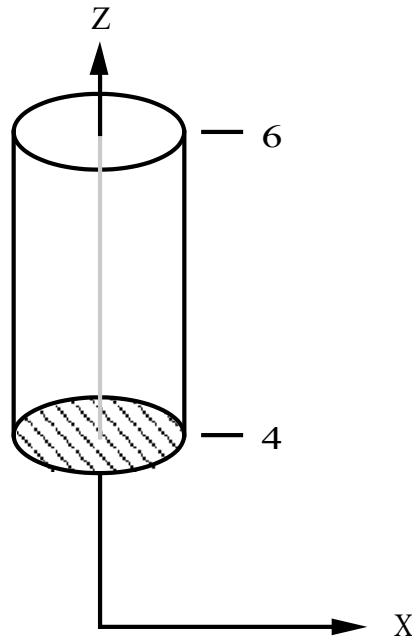
TR x_o y_o z_o ($x1$ $y1$ $z1$ $x2$ $y2$ $z2$)

will be abbreviated for convenience as:

(TR)

Note: The effect of rotating a PLANE can be to reverse the positive normal direction. This needs to be considered when sector equations using the plane are defined.

Example of a finite z-axis cylinder of radius 0.5, whose center is at $z = 5$ and whose ends are at $z = 4$ and $z = 6$. In the surface-definition coordinate system, the initial axis of this cylinder is along the x' -axis and the end planes are normal to the x' -axis. We will use a TR specification to move it to the desired position shown here.



We start with this definition:

```
17 CYLINDER 0.5 -1. 1.
```

which describes a finite cylinder of radius = 0.5 along the x' -axis, with its center at the origin and its ends at $x' = -1$ and $x' = +1$.

For the TR specification, we write:

(x_0, y_0, z_0) = desired location of the cylinder origin in the x, y, z system = $(0, 0, 5)$

(x_1, y_1, z_1) = desired location of a point on $+x'$ axis in the x, y, z system = $(0, 0, 6)$

(x_2, y_2, z_2) = desired location of a point on $+y'$ axis in the x, y, z system = $(0, 1, 5)$

This specification aligns the cylinder axis with the z -axis.

The final surface specification is:

```
17 CYLINDER 0.5 -1. 1. TR 0. 0. 5. 0. 0. 6. 0. 1. 5.
```

Example of the same cylinder in the same location, but defined in an alternative way.

```
17 CYLINDER 0.5 4 6 TR 0 0 0 0 0 1 0 1 0
```

This demonstrates a common use of the TR command to rotate an object about the origin (0, 0, 0) into a preferred orientation.

Common rotation commands are:

TR 0 0 0 0 1 0 0 0 1 rotation from x-axis aligned to y-axis aligned

TR 0 0 0 0 0 1 0 1 0 rotation from x-axis aligned to z-axis aligned

These rotations are combined with translations by **adding** the translation increments to both points.

For example,

- to only rotate a surface from x-axis aligned to y-axis aligned, use:

```
TR 0 0 0 0 1 0 0 0 1
```

- to both rotate a surface from x-axis aligned to y-axis aligned, and translate it to point (1. 3. 5.) use:

```
TR 1. 3. 5. 1. 4. 5. 1. 3. 6.
```

Example: Two TR commands are used to rotate two x-planes to truncate an x-cylinder and form a bounded volume. The volume enclosed by the truncated cylinder is given by the sector definition:

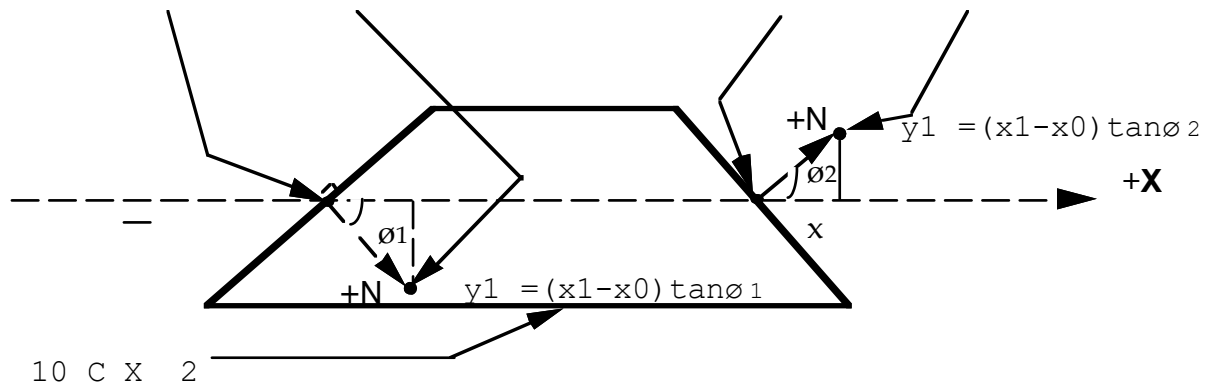
```
SECTOR 1 EXAMPLE +10 +1 -2
```

```
1 PLANE 0 0 0 1 0 0
```

```
TR 0 0 0 1 -1.321 0
(x0 y0 z0) (x1 y1 z1)
```

```
2 PLANE 0 0 0 1 0 0
```

```
TR 10 0 0 11 1 0
(x0 y0 z0) (x1 y1 z1)
```



Example

Here we assemble several volumes via the **TR** command, which Translates/Rotates surfaces from the surface-definition coordinate system x',y',z' into the problem coordinate system x,y,z . The figure shows a section in the x - y plane.

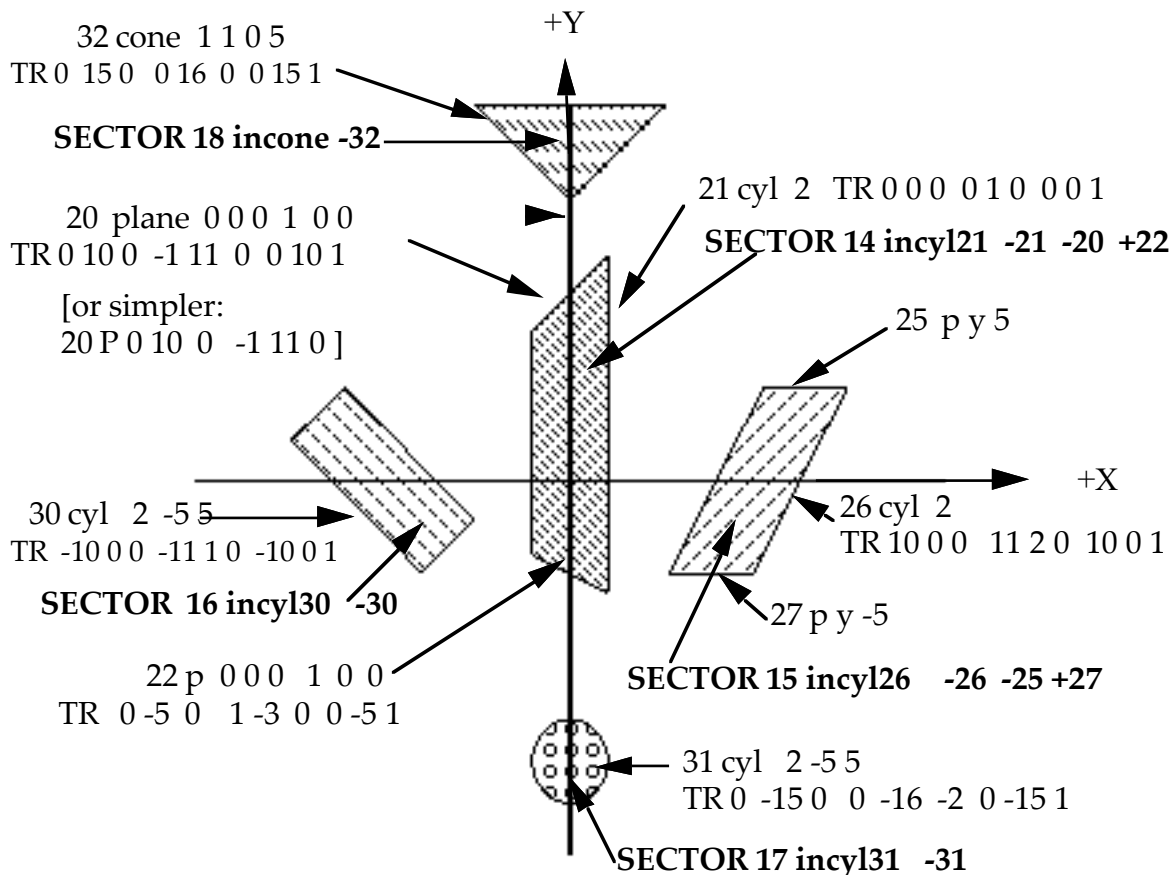
Sector **14** consists of a cylinder (21) terminated in $+y$ by plane 20 and in $-y$ by plane 22. These planes are placed into the geometry by defining them as x' -planes and then **TR**ing them to the desired location.

Sector **15** consists of a cylinder (26) terminated in $+y$ by plane 25 and in $-y$ by plane 27. The cylinder is translated to $x = 10$ and rotated by the **TR** command.

Sector **16** is defined as an x' -cylinder (30) which extends from -5 to 5 (center at zero). It is translated by a **TR** command to $x = -10$ and rotated by -45° .

Sector **17** is defined as an x' -cylinder (31) which extends from -5 to 5 . It is translated to $y = -15$ and rotated into the y - z plane.

Sector **18** describes a 45° cone confined between planes at $x' = 0$ and $x' = 5$. The cone is translated to $(0, 15, 0)$ and rotated into alignment with the $+Y$ axis.



Examples of using the TR command in the SURFACE Data Block and alternate ways of defining surface positions.

Surface Definition	Explanation
1 plane 0. 0. 0. 1. 0. 0. TR 6. 0. 0.	an x-plane (normal to the x-axis) at x=6
1 plane 6. 0. 0. 7. 0. 0.	an x-plane at x=6
1 PLANE X 6	an x-plane at x=6
1 P X 6	an x-plane at x=6
2 plane 0. 5. 0. 0. 6. 0.	a y-plane at y=5
2 P Y 5	a y-plane at y=5
3 P Z 7	a z-plane at z=7
3 plane 0. 0. 7. 0. 0. 8.	a z-plane at z=7
10 CYL Y 2.5 2.1 6.2	a y-axis cylinder of r=2.5, extending from y=2.1 to y=6.2
1 CYL 3.3 2.1 6.2	an x-axis cylinder of r=3.3, extending from x=2.1 to x=6.2 (x-axis = default axis)
2 CYL 4.9 TR 0. 0. 0. 0. 1. 0. 0. 0. 1.	a y-axis cylinder of r=4.9
2 C Y 4.9	a y-axis cylinder of r=4.9
3 C Z 2.9	a z-axis cylinder of r=2.9
4 CYL 1.9 TR 0. 10. 10. 0. 11. 11. 1. 10. 10.	a cylinder of r=1.9 with axis passing through point (0,10,10) and oriented at 45° between the y- and z-axes
5 CYL 2.1 TR 10. 0. 0. 10.866 0.5 0. 10. 0. 1.	a cylinder of r=2.1 with axis passing through point (10,0,0) and oriented at 30° between the x- and y-axes
2 CYL 2.2 TR 10. 0. 0. 10.866 0. 0.5 10. 1. 0.	a cylinder of r=2.2 with axis passing through point (10,0,0) and oriented at 30° between the x- and z-axes
6 SPHERE 5.0 TR 3. 4. 5.	a sphere of r=5.0 with a center at (3, 4, 5)

3.6.1 “SameAs” Surface Feature

This feature allows the user to specify a surface simply by stating that it is identical to a previously-specified surface, but with a different TR Translation/Rotation spec. This permits the user to easily specify a set of complex surfaces (e.g., surfaces of revolution), all identical in size and shape, without having to enter the basic surface spec more than once.

The “sameas” feature is used to specify a surface this way:

surf-ID# SAMEAS ***reference-surf-ID#*** (TR)

where:

surf-ID# is the user-defined surface ID# (integer) for the surface;

reference-surf-ID# is the ***surf-ID#*** of a previously-defined surface (not a sameas surface) which provides the data for this surface definition;

(TR ...) is the Translation/Rotation specification to be applied to this surface.

Example: Set up a set of surfaces of revolution, based on the description given for surface 102.

```

SURFACES
  100 BOX 30. 20. 20.
  102 REV 6 0. 0. 0.1 0.1 0.2 0.2 0.3 0.21 0.4 0.5 0.5
0.
  103 SAMEAS 102 TR 1. 0. 0.
  104 SAMEAS 102 TR 2. 0. 0.
  ...
GEOMETRY
  SECTOR 102 REV1 -102
  SECTOR 103 REV2 -103
  SECTOR 104 REV3 -104
  ...

```

3.7 First Order Planar Surfaces

3.7.1 Plane

A plane can be specified by any one of the following four methods.

Coordinate plane (Plane normal to a coordinate axis)

To specify a plane normal to a coordinate axis, the surface definition is:

$$\text{surf-ID\#} \quad \begin{bmatrix} \text{PLANE} \\ \text{PLA} \\ \text{P} \end{bmatrix} \begin{bmatrix} \text{X} \\ \text{Y} \\ \text{Z} \end{bmatrix} \quad d \quad (\text{TR } \dots)$$

where:

surf-ID# is the user-defined surface ID# (integer) for the surface;

PLANE (or **PLA** or **P**) is the keyword identifying the surface type;

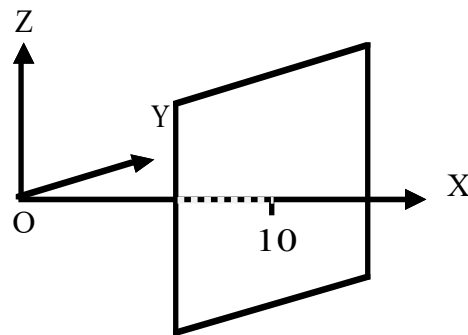
X (or **Y** or **Z**) specifies the coordinate axis normal to this plane;

d = distance from the origin to the surface, along the coordinate axis.

Note: The positive plane normal is in the direction of increasing coordinate value.

Example:

```
2 PLANE X 10
```



specifies that surface 2, a plane, is normal to the x-axis; its x-axis intercept is at x=10. The positive surface normal points in the +x direction.

General plane, 2-point definition: To specify a general plane using the two-point method, the surface definition is:

$$surf-ID\# \begin{bmatrix} \text{PLANE} \\ \text{PLA} \\ \text{P} \end{bmatrix} x'_1 y'_1 z'_1 \quad x'_2 y'_2 z'_2 \quad (TR \dots)$$

where:

surf-ID# is the user-defined surface ID# for the surface;

PLANE (or **PLA** or **P**) is the keyword identifying the surface type;

$x'_1 y'_1 z'_1$ specifies a point on the plane;

$x'_2 y'_2 z'_2$ specifies a point located on the plane positive normal through the first point.

For this general plane case, the user specifies the direction of the positive normal. Normally one would want to choose this direction in some consistent way. For example, one could choose it such that the object bounded by the plane lies on the negative side of the plane, or alternatively, that the positive normal points along a positive coordinate direction.

For example, the specification:

3 PLANE 0. 5. 0. 0. 6. 0.

says that this is a plane with *surf-ID#* 3, which passes through the point (0,5,0). (Length units are specified in the BASIC Data Block). The second x y z triplet (0,6,0) tells COG the plane is oriented in the y-direction (plane normal passes from point (0,5,0) through point (0,6,0)).

General PLANE, 4-point definition: To specify a general plane using the four-point method, the surface definition is:

$$surf-ID\# \begin{bmatrix} \text{PLANE} \\ \text{PLA} \\ \text{P} \end{bmatrix} x'_1 y'_1 z'_1 \quad x'_2 y'_2 z'_2 \quad x'_3 y'_3 z'_3 \quad x'_4 y'_4 z'_4 \quad (TR \dots)$$

where:

surf-ID# is the user-defined surface ID# for the surface;

PLANE (or **PLA** or **P**) is the keyword identifying the surface type;

$x'_1 y'_1 z'_1, x'_2 y'_2 z'_2, x'_3 y'_3 z'_3$ specifies three points on the plane (but not in a line);

$x'_4 y'_4 z'_4$ specifies a point off the plane, located anywhere on the positive-normal side.

General PLANE, analytic definition: To specify a general plane using the analytic method, the surface definition is:

$$surf-ID\# \begin{bmatrix} \text{PLANE} \\ \text{PLA} \\ \text{P} \end{bmatrix} \cdot c_0 \ c_1 \ c_2 \ c_3 \ (\text{TR} \dots)$$

where:

surf-ID# is the user-defined surface ID# for the surface;

PLANE (or **PLA** or **P**) is the keyword identifying the surface type;

c_i values are the coefficients of the analytic representation of the plane:

$$c_0 + c_1 x' + c_2 y' + c_3 z' = 0$$

Examples of the PLANE command used in the SURFACE Data Block

<u>EXAMPLE</u>	<u>Explanation</u>
1 PLANE 0. 0. 0. 1. 0. 0	an x-plane at the origin
2 plane 6. 0. 0. 7. 0. 0.	an x-plane at x = 6 units
or	
3 PLANE x 6	an x-plane at x = 6 units
4 plane 0. 5. 0. 0. 6. 0.	a y-plane at y = 5 units
or	
5 plane y 5.	a y-plane at y = 5 units
4 PLANE 0. 0. 8. 0. 0. 9.	a z-plane at z = 8 units
6 plane 0. 0. 0. 0. 0. 1.	a z-plane at z = 7 units
TR 0. 0. 7.	
or	
7 plane 0. 0. 7. 0. 0. 8.	a z-plane at z = 7 units
or	
8 plane z 7.	a z-plane at z = 7 units
9 PLANE 0. 1. 2. 0. 2. 3.	a plane rotated 45° with respect to the y and z axes

3.7.2 Tetrahedron

A tetrahedron, a four-sided polyhedron, is specified by:

surf-ID# $\left[\begin{array}{c} \text{TETRAHEDRON} \\ \text{TET} \\ \text{P4} \end{array} \right] \quad \begin{array}{l} x'_1 y'_1 z'_1 \quad x'_2 y'_2 z'_2 \quad x'_3 y'_3 z'_3 \\ x'_4 y'_4 z'_4 \quad \{x'_{b1} \ x'_{b2}\} \text{ (TR)} \end{array}$

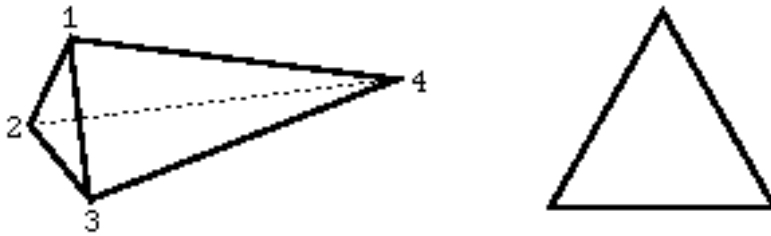
where:

surf-ID# is the user-defined surface ID# for the surface;

TETRAHEDRON(or **TET** or **P4**) is the keyword identifying the surface type;

$x'_1 y'_1 z'_1 \ x'_2 y'_2 z'_2 \ x'_3 y'_3 z'_3 \ x'_4 y'_4 z'_4$ specify the four points falling at the corners of the figure;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a *TETRAHEDRON* with points at $(-0.01 \ 2.5 \ 0.) \ (-0.01 \ -2.5 \ -2.5) \ (-0.01 \ -2.5 \ 2.5) \ (2.5 \ .0 \ .0)$:

```
10 TET      -0.01  2.5  0.   -0.01 -2.5 -2.5   -0.01 -2.5  2.5
              2.5  0.  0.
```

3.7.3 Pentahedron

A PENTAHEDRON, a five-sided polyhedron, is specified by:

$$\text{surf-ID\#} \left[\begin{array}{c} \text{PENTAHEDRON} \\ \text{PEN} \\ \text{P5} \end{array} \right] x'_1 y'_1 z'_1 \dots x'_5 y'_5 z'_5 \{x'_6 y'_6 z'_6\} \\ \{x'_{b1} \ x'_{b2}\} (TR \dots)$$

where:

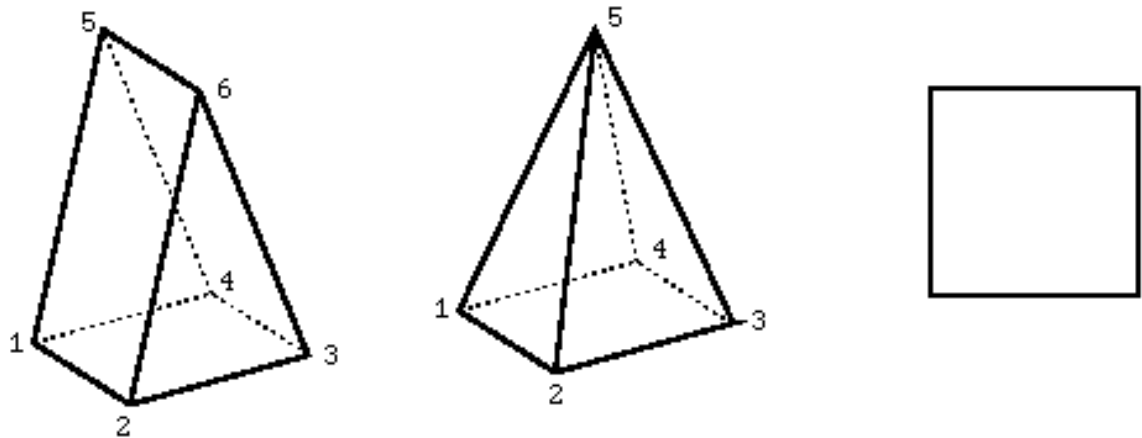
surf-ID# is the user-defined surface ID# for the surface;

PENTAHEDRON(or **PEN** or **P5**) is the keyword identifying the surface type;

$x'_1 y'_1 z'_1 \dots$ specify the five or six points falling at the corners of the figure.

If there are five corners, the figure is a pyramid. If there are six corners, the figure is a wedge. Note that the order of specifying the points must follow the order indicated below.

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a PENTAHEDRON with points at (-5 -5 0) (5 -5 0) (5 5 0) (-5 5 0) (0 0 10):

12 PEN -5 -5 0 5 -5 0 5 5 0 -5 5 0 0 0 10

3.7.4 Box

A BOX is a special case of a six-sided polyhedron and is specified by:

surf-ID# **[BOX
B]** a_x a_y a_z (TR)

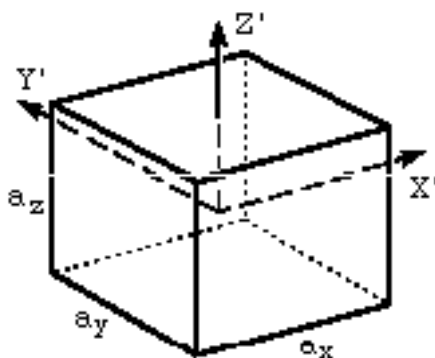
where:

surf-ID# is the user-defined surface ID# for the surface;

BOX (or **B**) is the keyword identifying the surface type;

a_x a_y a_z specify the length of each side.

The center of the box is at the origin and the sides are parallel to the coordinate planes.



Example of a rectangular BOX, x side = 2, y side=3, z side = 2

32 BOX 2 3 2

3.7.5 RPP: Rectangular Parallelepiped

surf-ID# **RPP** $x1$ $x2$ $y1$ $y2$ $z1$ $z2$

where:

surf-ID # is the surface number, **RPP** is the key word, $x1$ and $x2$ are the x limits, $y1$ and $y2$ are the y limits, and $z1$ and $z2$ are the z limits defining the rectangular parallelepiped.

Example1:... surface box

1 box 7. 8. 9.

and... surface rpp

1 rpp -3.5 3.5 -4. 4. -4.5 4.5

Example2:... surface box

1 box 7. 8. 9. tr 3.5 4. 4.5

and... surface rpp

1 rpp 0 7. 0. 8. 0. 9.

3.7.6 Hexahedron

A hexahedron is a six-sided polyhedron specified by:

$$surf-ID\# \left[\begin{array}{c} \text{HEXAHEDRON} \\ \text{HEX} \\ \text{P6} \end{array} \right] x'_1 y'_1 z'_1 \dots x'_8 y'_8 z'_8 \{x'_{b1} \ x'_{b2}\} (TR \dots)$$

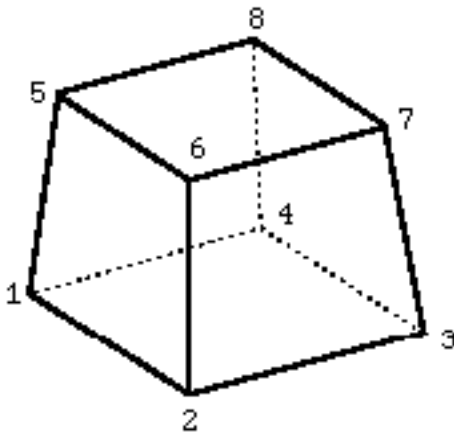
where:

surf-ID# is the user-defined surface ID# for the surface;

HEXAHEDRON (or **HEX** or **P6**) is the keyword identifying the surface type;

$x'_1 y'_1 z'_1 \dots$ specify the eight points falling at the corners of the figure and should be specified in the order indicated on the figure below.

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example: HEXAHEDRON

```
27 HEXAHEDRON 0. 0. 0. 5 0. 0. 5. 5. 0. 0. 5. 0. 1. 1. 1.
               5 4 1 5. 4.4 5. 1. 4. 5.
```

This input is listed in the COG outputfile as:

SURFACE NUMBER	27	HEXAHEDRON
POINT AT A CORNER	(0.00000E-01,	0.00000E-01, 0.00000E-01)
POINT AT A CORNER	(5.00000E+00,	0.00000E-01, 0.00000E-01)
POINT AT A CORNER	(5.00000E+00,	5.00000E+00, 0.00000E-01)
POINT AT A CORNER	(0.00000E-01,	5.00000E+00, 0.00000E-01)
POINT AT A CORNER	(1.00000E+00,	1.00000E+00, 1.00000E+00)
POINT AT A CORNER	(5.00000E+00,	4.00000E+00, 1.00000E+00)
POINT AT A CORNER	(5.00000E+00,	4.40000E+00, 5.00000E+00)
POINT AT A CORNER	(1.00000E+00,	4.00000E+00, 5.00000E+00)

3.7.7 Right Prism

A general right prism with its axis parallel to the x' -axis is specified by:

surf-ID# **PRISM** *number-points* $y'_1 z'_1$ $y'_2 z'_2$... $y'_n z'_n$

where:

$\{x'_{b1} \ x'_{b2}\}$ (*TR*)

surf-ID# is the user-defined surface ID# for the surface;

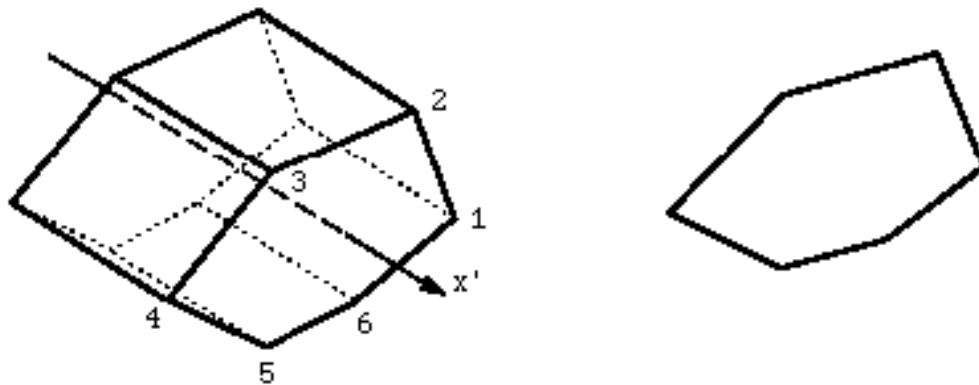
PRISM (or **PRI**) is the keyword identifying the surface type;

number-points is the number of corner points (and the number of prism edges) on the $y'z'$ face (see figure);

$y'_1 z'_1$... $y'_n z'_n$ specify the corners of the prism on the $y'z'$ face.

Note: The corner points should be given in the order shown in the figure. The defined prism must be non-reentrant—i.e., any straight line drawn so it starts within the figure and passes out through a side cannot again enter the figure.

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a *PRISM* of six points located at $(y' z') = (4 \ 0)$, $(3 \ 4)$, $(-2 \ 4)$, $(-4 \ 2)$, $(-3 \ -1)$, $(0 \ -2)$. Via a *TR* command, the prism is translated to a new origin $(-3 \ -6 \ 0)$, and rotated to point along the z -axis.

```
15 PRISM 6  4 0  3 4 -2 4 -4 2 -3 -1  0 -2
          TR -3 -6 0 -3 -6 1 -3 -5 0
```

3.7.8 Right Pyramid

A general right pyramid with its apex at $x' = 0$ and its axis along the x' -axis is specified by:

surf-ID# **PYRAMID**
PYR x'_p **number-points** $y'_1 z'_1$ $y'_2 z'_2$... $y'_n z'_n$ { x'_{b1}
 x'_{b2} } (TR)

where:

surf-ID# is the user-defined surface ID# for the surface;

PYRAMID (or **PYR**) is the keyword identifying the surface type;

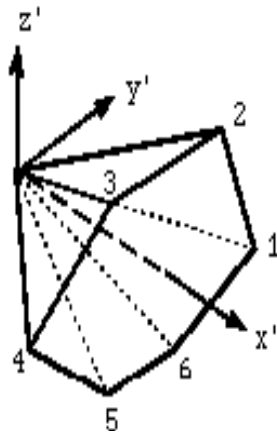
x'_p is the distance from the origin to the $y'z'$ plane where the corners of the pyramid are to be defined;

number-points is the number of corner points (and the number of pyramid edges) in the $y'z'$ plane at $x' = x'_p$ (see figure);

$y'_1 z'_1$... $y'_n z'_n$ specify the corners of the prism in the $y'z'$ plane at $x' = x'_p$.

Note: The corner points must be given in the order shown in the figure. The defined pyramid must be non-reentrant—i.e., any straight line drawn so it starts within the figure and passes through a side cannot again enter the figure.

{ x'_{b1} x'_{b2} } specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a **PYRAMID** defined at $x' = 2$ with 6 pairs of ($y' z'$) points, rotated to point along the y -axis.

```
27 PYRAMID 2.0 6 1 -2 2 -1 3 1 0 2 -6 0 -2 -1.75
0. 5. TR 0 0 0 0 1 0 0 0 1
```

This input is listed in the COG outputfile as:.

```
SURFACE NUMBER      27      RIGHT PYRAMID
X' VALUE WHERE THE CROSS-SECTION IS SPECIFIED
2.00000E+00
CORNER AT Y'= 1.00000E+00  Z'=-2.00000E+00
CORNER AT Y'= 2.00000E+00  Z'=-1.00000E+00
CORNER AT Y'= 3.00000E+00  Z'= 1.00000E+00
CORNER AT Y'= 0.00000E-01  Z'= 2.00000E+00
CORNER AT Y'=-6.00000E+00  Z'= 0.00000E-01
CORNER AT Y'=-2.00000E+00  Z'=-1.75000E+00
SURFACE BOUNDED BY PLANES AT X'= 0.0 AND X'= 5.0
```


3.8 The Whole Family of Second-Order Surfaces

3.8.1 Sphere

A sphere is specified by:

surf-ID# $\left[\begin{array}{c} \text{SPHERE} \\ \text{SPH} \\ \text{S} \end{array} \right] \text{radius } \{x'_{b1} \ x'_{b2}\} (TR \dots)$

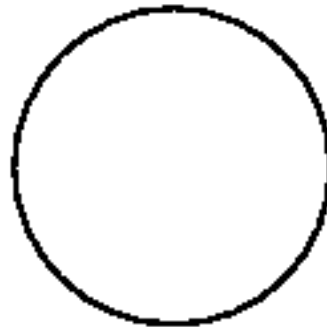
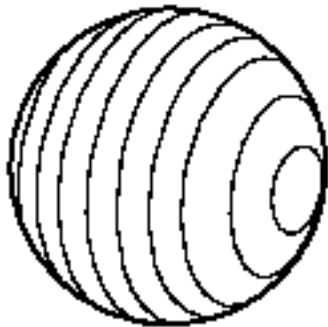
where:

surf-ID# is the user-defined surface ID# for the surface;

SPHERE (or **SPH** or **S**) is the keyword identifying the surface type;

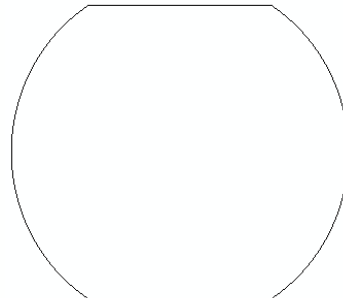
radius is the sphere radius;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a SPHERE of radius 9.5, truncated by x' -planes at $x' = -8$ and $x' = 8$, rotated into the y -direction.

```
17 SPHERE 9.5 -8. 8.
   TR 0 0 0    0 1 0    0 0 1
```



3.8.2 Ellipsoid

An ellipsoid is specified by:

surf-ID# **ELLIPSOID**
ELL
E *a b c {x'_{b1} x'_{b2}} (TR)*

where:

surf-ID# is the user-defined surface ID# for the surface;

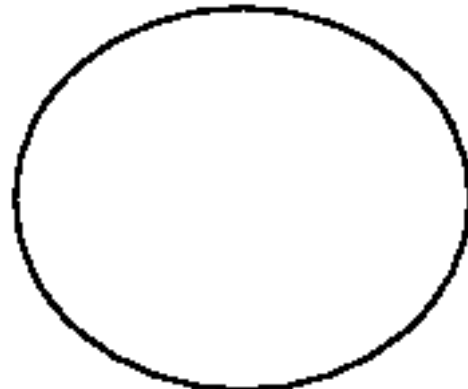
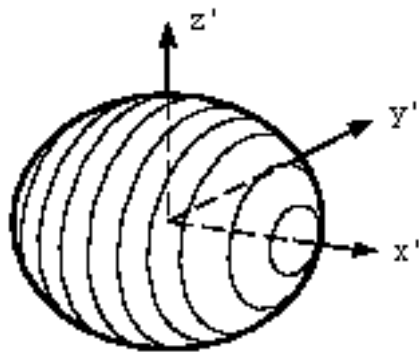
ELLIPSOID (or **ELL** or **E**) is the keyword identifying the surface type;

$\pm a$ are the x' -axis intercepts of the surface;

$\pm b$ are the y' -axis intercepts of the surface;

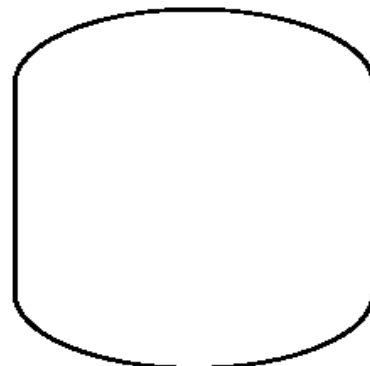
$\pm c$ are the z' -axis intercepts of the surface;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of an ELLIPSOID with x' , y' , and z' intercepts of 9, 7, and 5 respectively. The surface is truncated by x' -axis endplanes at $x' = -7$ and $x' = 7$.

27 ELL 9 7 5 -7 7



3.8.3 Right Circular Cylinder

A right circular cylinder is specified by one of these two methods.

Coordinate cylinder (Axis coincident with a coordinate axis)

$$surf-ID\# \begin{bmatrix} \text{CYLINDER} \\ \text{CYL} \\ \text{C} \end{bmatrix} \begin{bmatrix} \text{X} \\ \text{Y} \\ \text{Z} \end{bmatrix} \text{ radius } \{x'_{b1} \ x'_{b2}\} \text{ (TR)}$$

where:

surf-ID# is the user-defined surface ID# for the surface;

CYLINDER (or **CYL** or **C**) is the keyword identifying the surface type;

X (or **Y** or **Z**) specifies the coordinate axis coincident with the cylinder axis;

radius is the cylinder radius;

$\{x'_{b1} \ x'_{b2}\}$ specifies the axis intercepts of two surface-truncating endplanes, normal to the cylinder axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.

TR Note: For **coordinate** cylinders, only **translation** is allowed via the **TR** command. This type of cylinder may *not* be **rotated** into some other orientation.

General cylinder (Axis coincident with the x' -axis)

$$surf-ID\# \begin{bmatrix} \text{CYLINDER} \\ \text{CYL} \\ \text{C} \end{bmatrix} \text{ radius } \{x'_{b1} \ x'_{b2}\} \text{ (TR)}$$

where:

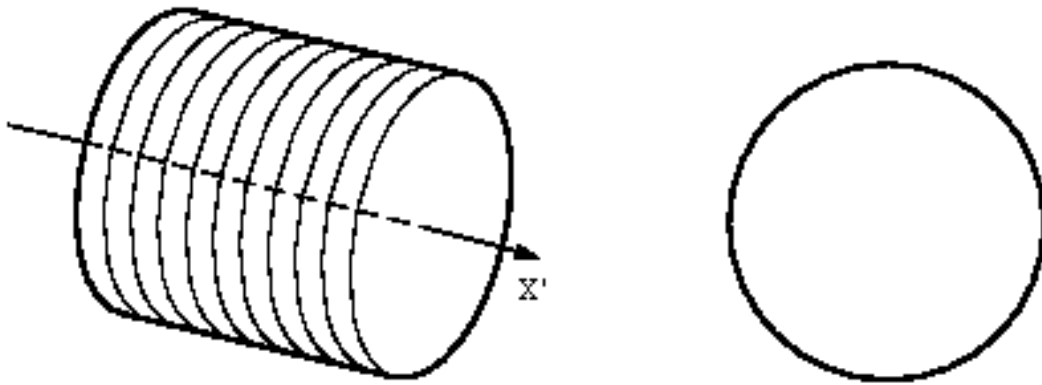
surf-ID# is the user-defined surface ID# for the surface;

CYLINDER (or **CYL** or **C**) is the keyword identifying the surface type;

radius is the cylinder radius;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.

The general cylinder is defined with its axis coincident with the x' -axis. It may be translated and rotated via the **TR** command to any desired position and orientation in problem x,y,z coordinates.



Example of a CYLINDER of radius 0.75 extending from $x' = -1$ to $x' = 1.9$, which has been rotated into the y-direction by a TR command.

```
27 CYL 0.75 -1 1.9    TR 0. 0. 0.    0. 1. 0.    0. 0. 1.
```

3.8.4 Right Elliptical Cylinder

A right elliptical cylinder is specified by:

$$\text{surf-ID\#} \left[\begin{array}{c} \text{ECYLINDER} \\ \text{ECYL} \\ \text{EC} \end{array} \right] b \ c \ \{x'_{b1} \ x'_{b2}\} \ (TR \)$$

The cylinder axis is coincident with the x' -axis.

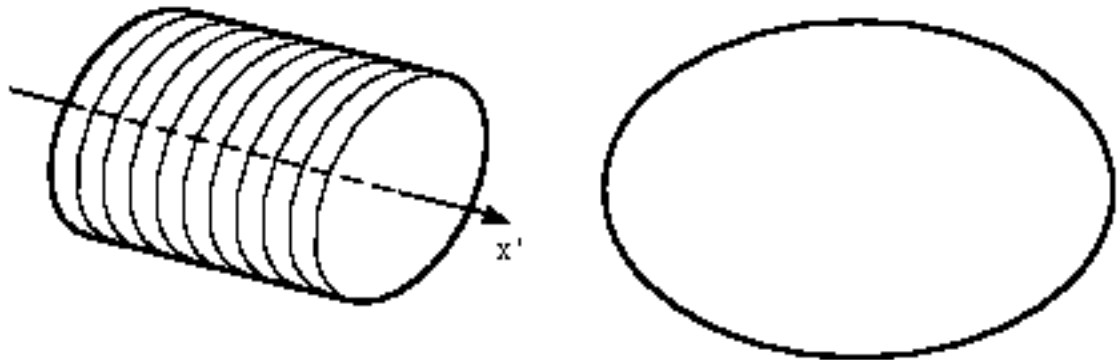
surf-ID# is the user-defined surface ID# for the surface;

ECYLINDER (or **ECYL** or **EC**) is the keyword identifying the surface type;

$\pm b$ are the y' -axis intercepts of the surface;

$\pm c$ are the z' -axis intercepts of the surface;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of an elliptical cylinder, *ECYLINDER*, with y' , z' intercepts of .35 and .8. The surface has been truncated by endplanes at $x' = 2$ and $x' = 4.4$. The surface has been rotated via a *TR* command into the y -direction.

```
27 EC .35 .8 2. 4.4 TR 0 0 0 0 1 0 0 0 1
```

3.8.5 Hyperboloid of One Sheet

A hyperboloid of one sheet is specified by:

surf-ID# **HYP1**
H1 *a b c {x'_{b1} x'_{b2}} (TR)*

The hyperboloid axis is coincident with the x' axis.

The figure is centered so that the minimum cross-sectional area occurs at $x' = 0$. Planes perpendicular to the x' -axis intersect the figure to form ellipses with semi-axes along the y' and z' directions.

surf-ID# is the user-defined surface ID# for the surface;

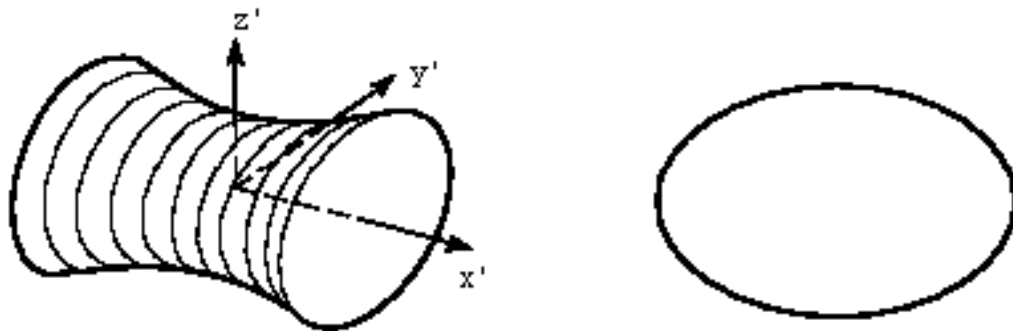
HYP1 (or **H1**) is the keyword identifying the surface type;

$\pm b$ are the y' -axis intercepts of the surface in the plane $x' = 0$;

$\pm c$ are the z' -axis intercepts of the surface in the plane $x' = 0$;

a determines the size of the ellipses in other x' planes. At $x' = x'_l$, the ellipses are enlarged by the factor: $\sqrt{1 + x'^2_l/a^2}$;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example Hyperboloid of One Sheet, HYP1. The *a*-parameter = 2.5, and the y' and z' intercepts are 1.5 and 1.0 respectively. Truncating endplanes occur at $x' = 1.7$ and $x' = 3.5$. The surface has been rotated into the z -direction.

32 HYP1 2.5 1.5 1.0 1.7 3.5 TR 0 0 0 0 0 1 0 1 0

3.8.6 Hyperbolic Cylinder

A hyperbolic cylinder is specified by:

$$\text{surf-ID\#} \begin{bmatrix} \text{HCYLINDER} \\ \text{HCYL} \\ \text{HC} \end{bmatrix} b \ c \ \{x'_{b1} \ x'_{b2}\} \ (TR \dots)$$

The hyperbolic cylinder is a hyperbola in the $y'z'$ plane extended along the x' -axis without change.

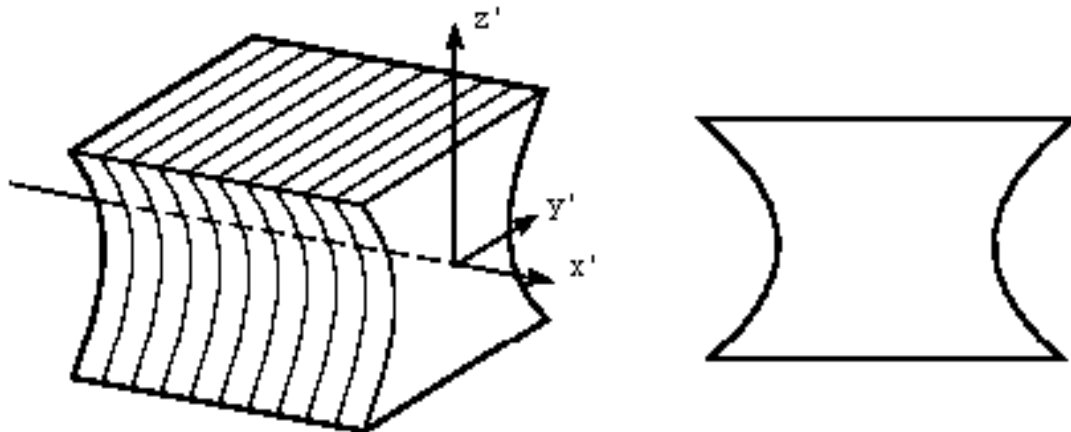
surf-ID# is the user-defined surface ID# for the surface;

HCYLINDER (or **HCYL** or **HC**) is the keyword identifying the surface type;

b is the y' distance to the surface in the plane $z' = 0$;

c determines the y' distance to the surface in other z' planes. At $z' = z'_l$, the y' distance to the surface is : $b\sqrt{1 + z'^2_l/c^2}$;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a hyperbolic cylinder **HCYLINDER** with *b*-parameter = 1 and *c*-parameter = 1.1. The surface is translated to (6.36, 6.36, 0) and rotated into the z -direction, via a **TR** command.

```
30 HCYLINDER 1 1.1 -2 2 TR 6.36 6.36 0 6.36 6.36 1
6.36 7.36 0
```

This input is listed in the COG outputfile as:

GEOMETRICAL SURFACE SPECIFICATIONS: UNIT LENGTH IS ONE
CENTIMETER

SURFACE NUMBER 30 HYPERBOLIC CYLINDER

CONSTANT B = 1.00000E+00

CONSTANT C = 1.10000E+00

SURFACE BOUNDED BY PLANES AT X' = -2.00000E+00 AND

X' = 2.00000E+00

T/R DATA

CENTER OF (X',Y',Z') SYSTEM (6.36, 6.36, 0.0)

POINT ON X'-AXIS (6.36, 6.36, 1.0)

POINT ON Y'-AXIS (6.36, 7.36, 0.0)

3.8.7 Parabolic Cylinder

surf-ID# $\left[\begin{array}{l} \text{PCYLINDER} \\ \text{PCYL} \\ \text{PC} \end{array} \right] y'_1 z'_1 \{x'_{b1} x'_{b2}\} (TR \dots)$

The parabolic cylinder is a parabola in the (y', z') plane, which extends indefinitely in the x' direction.

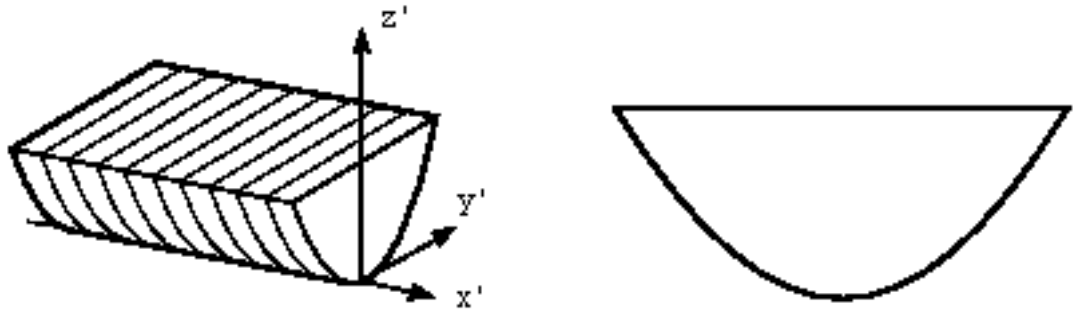
surf-ID# is the user-defined surface ID# for the surface;

PCYLINDER (or **PCYL** or **PC**) is the keyword identifying the surface type;

$y'_1 z'_1$ specifies a point on the surface;

Note: This surface extends indefinitely in the $+z'$ and the $\pm x'$ directions, and must be terminated by other surfaces.

$\{x'_{b1} x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of a parabolic cylinder **PCYLINDER** defined by point $(y', z') = (1.2, 0.7)$.

```
31 PCYLINDER 1.2 0.7
```

3.8.8 Right Circular Cone

surf-ID# $\begin{bmatrix} \text{CONE} \\ \text{CON} \\ \text{K} \end{bmatrix} a \text{ radius } \{x'_{b1} \ x'_{b2}\} (TR \dots)$

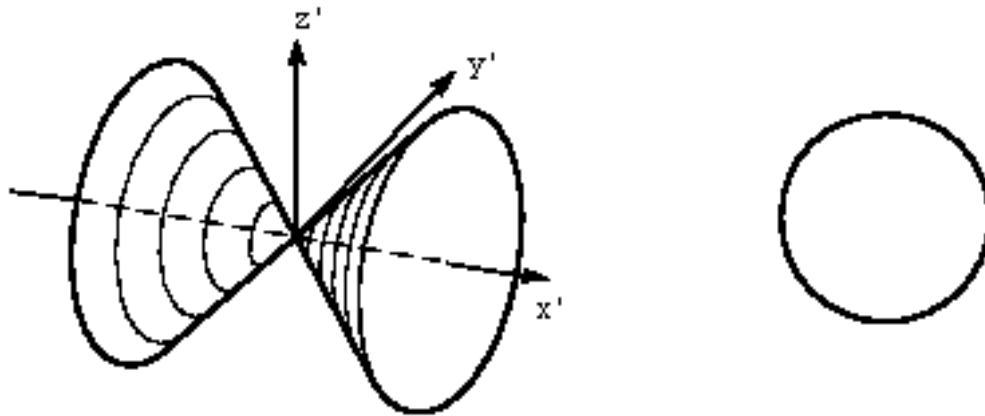
The axis of the right circular cone is coincident with the x' -axis, with the apex at the origin.

surf-ID# is the user-defined surface ID# for the surface;

CONE (or **CON** or **K**) is the keyword identifying the surface type;

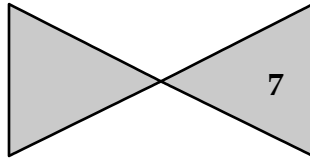
radius specifies the cone radius at $x' = a$;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.

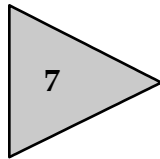


Example of a CONE whose radius at a distance = 1 along the x' -axis, is 1. (a 45° cone)

```
SURFACES
7 CONE 1. 1.
```



7 cone 1. 1. \$ 45° cone



7 cone 1. 1. -5. 0. \$ Truncated cone

If a single cone, rather than the default double cone, is desired, you must use the $\{x'_{b1}$
 $x'_{b2}\}$ specification to eliminate one cone.

3.8.9 Elliptical Cone

surf-ID# $\begin{bmatrix} \text{ECONE} \\ \text{ECON} \\ \text{EK} \end{bmatrix}$ *a b c {x'_{b1} x'_{b2}} (TR)*

The axis of the elliptical cone is coincident with the x' -axis, with vertex at the origin. In cross-section, the surface describes an ellipse with semi-axes along the y' and z' directions.

surf-ID# is the user-defined surface ID# for the surface;

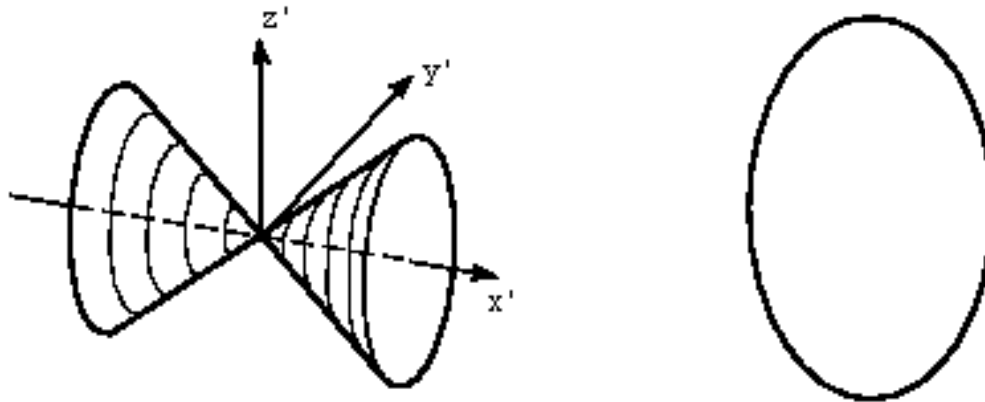
ECONE (or **ECON** or **EK**) is the keyword identifying the surface;

b is the y' semi-axis in the plane $x' = a$;

c is the z' semi-axis in the plane $x' = a$;

$\{x'_{b1} x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.

Note: If a single cone, rather than the default double cone, is desired, you must use the $\{x'_{b1} x'_{b2}\}$ specification to eliminate one cone.



Example of an elliptical cone **ECONE** with y' and z' semi-axes of 1.1 and 0.55 respectively, measured in plane at $x' = 1.0$. Via a **TR** command, the vertex (origin) of the cone is translated to (0.5,0,0).

```
42 ECONE 1.0 1.1 0.55 TR 0.5 0. 0.
```

3.8.10 Hyperboloid of Two Sheets

surf-ID# **[HYP2
H2]** *a b c {x'_{b1} x'_{b2}} (TR)*

The axis of rotation of a hyperboloid of two sheets is coincident with the x' -axis. The origin lies at the point of symmetry. In cross-section, the surface describes an ellipse with semi-axes along the y' and z' directions.

surf-ID# is the user-defined surface ID# for the surface;

HYP2 (or **H2**) is the keyword identifying the surface;

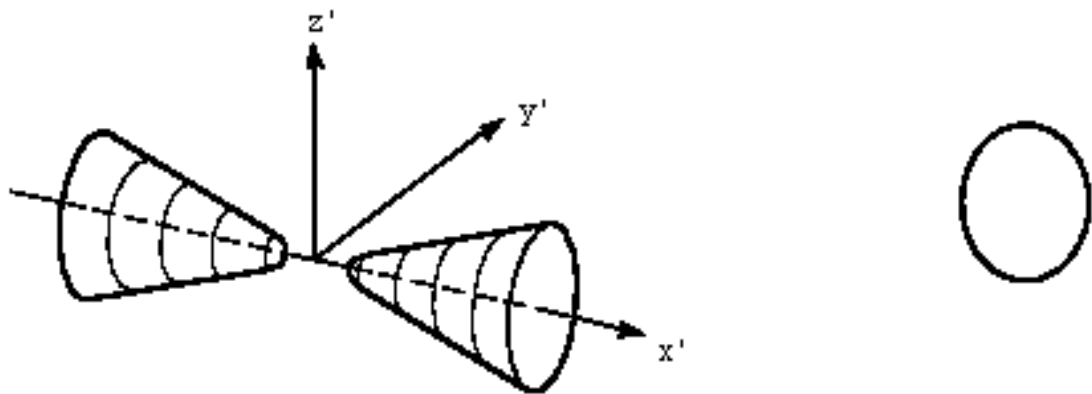
$\pm a$ are the x' -axis intercepts of the sheets;

b is the y' semi-axis in the plane $x' = a \sqrt{2}$;

c is the z' semi-axis in the plane $x' = a \sqrt{2}$;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.

Note: If a single-sheet hyperboloid, rather than the default double-sheet hyperboloid, is desired, you must use the $\{x'_{b1} \ x'_{b2}\}$ specification to eliminate one sheet.



Example of a two-sheet hyperboloid HYP2 with a -, b -, and c -parameters of .5, .35, and .35 respectively.

```
63 HYP2 .5 .35 .35
```

3.8.11 Elliptic Paraboloid

surf-ID# **EPAR**
EP *a b c {x'_{b1} x'_{b2}} (TR)*

The axis of rotation of the elliptic paraboloid is coincident with the x' -axis, with its vertex at the origin. In cross-section, the surface describes an ellipse with semi-axes along the y' and z' directions.

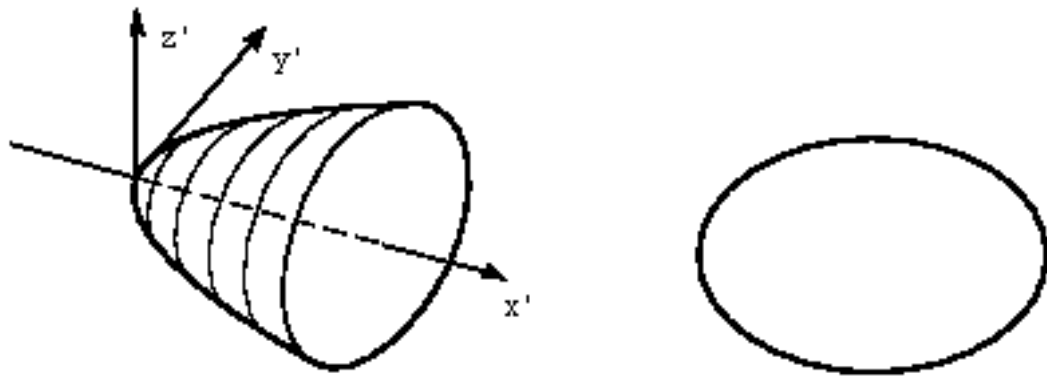
surf-ID# is the user-defined surface ID# for the surface;

EPAR (or **EP**) is the keyword identifying the surface;

b is the y' semi-axis in the plane $x' = 1/a$;

c is the z' semi-axis in the plane $x' = 1/a$;

$\{x'_{b1} \ x'_{b2}\}$ specifies the x' -axis intercepts of two surface-truncating endplanes, normal to the x' -axis. The order that the two x'_b values are given is not important—the volume exists at all points between the two.



Example of an elliptic paraboloid EPAR with a -, b -, and c -parameters of 1.0, 0.8, and 0.5 respectively.

43 EPAR 1.0 0.8 0.5

3.8.12 Hyperbolic Paraboloid

surf-ID# **HPARABOLA**
HPAR
HP *a b c {x'_{b1} x'_{b2}} (TR)*

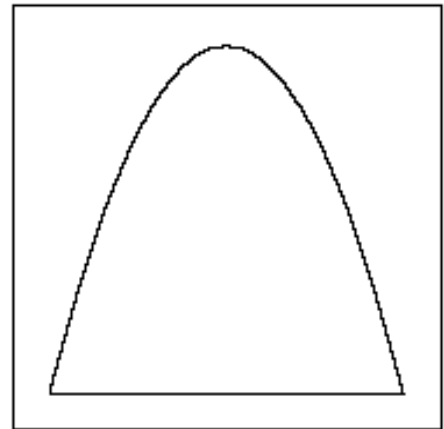
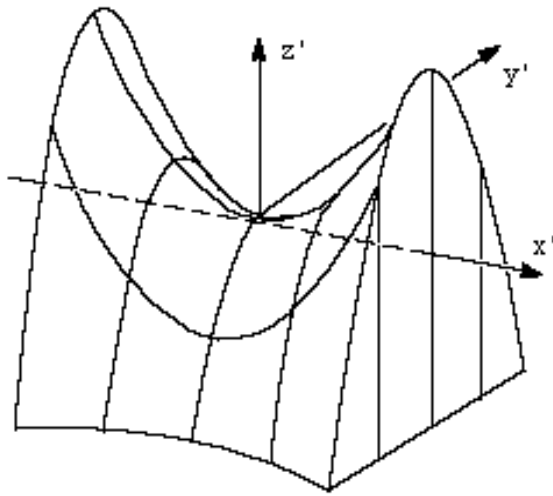
The axes of symmetry of the hyperbolic paraboloid are centered at the origin and coincide with the coordinate axes. This figure is described by the equation:

$$(x'/a)^2 - (y'/b)^2 - cz' = 0$$

surf-ID# is the user-defined surface ID# for the surface;

HPARABOLA (or **HPAR** or **HP**) is the keyword identifying the surface;

{x'_{b1} x'_{b2}} specifies the *x'*-axis intercepts of two surface-truncating endplanes, normal to the *x'*-axis. The order that the two *x'_b* values are given is not important—the volume exists at all points between the two.



Example of a hyperbolic paraboloid **HPARABOLA** with *a*-, *b*-, and *c*-parameters of 1.1, 2.0, and 3.0 respectively.

```
44 HPARABOLA 1.1 2.0 3.0
```

3.9 Special Figures of Rotation

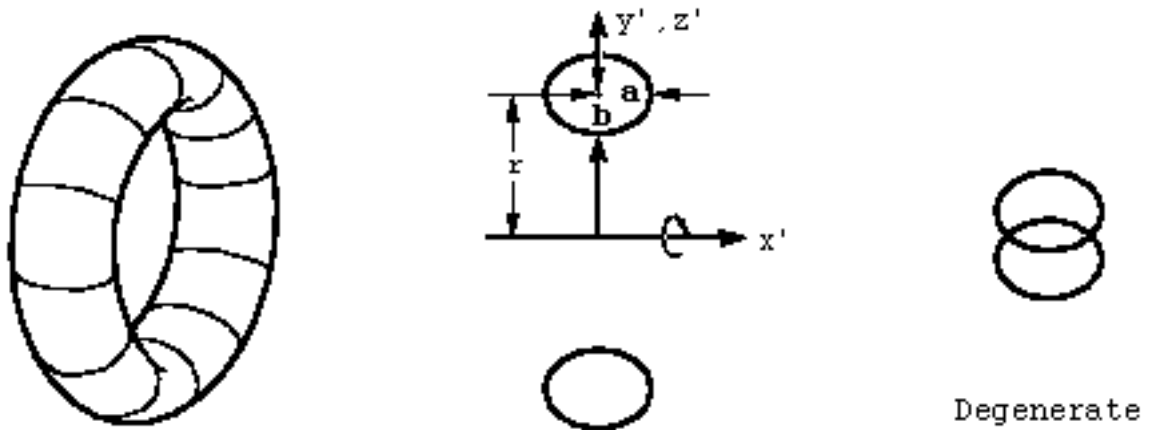
3.9.1 Torus

surf-ID# $\begin{bmatrix} \text{TORUS} \\ \text{TOR} \\ \text{T} \end{bmatrix} r \ a \ \{b\} \ (TR \dots)$

The elliptical torus is formed by the rotation around the x' -axis of an ellipse defined in the (x', y') plane. This ellipse has its center on the y' -axis a distance r from the origin, a semi-axis parallel to the x' -axis equal to a , and a semi-axis along the y' -axis equal to b . If b is omitted in the input, b is set equal to a , and a circular torus is defined. Note that the degenerate form (when $b > r$) excludes the central region.

surf-ID# is the user-defined surface ID# for the surface;

TORUS (or **TOR** or **T**) is the keyword identifying the surface;



Example of an elliptical **TORUS**, with a major radius $r = 1.0$, and ellipse semi-axes of 0.8 in x' and 0.5 in y' .

```
43 TORUS 1.0 0.8 0.5
```


3.9.2 Curve Made of Straight-Line Segments Rotated About the X'-axis

surf-ID# $\left[\begin{array}{c} \text{REVOLUTION} \\ \text{REV} \\ \text{R} \end{array} \right] \text{number-points} \quad x'_1 r'_1 \quad x'_2 r'_2 \quad \dots \quad x'_n r'_n$
(TR)

OR

surf-ID# $\left[\begin{array}{c} \text{REVOLUTION} \\ \text{REV} \\ \text{R} \end{array} \right] \text{number-points} \quad \text{POLAR} \quad r_1 \theta_1 \quad r_2 \theta_2 \quad \dots \quad r_n \theta_n$
(TR)

The given points are connected by straight lines, forming a continuous curve. This curve is then rotated about the x' -axis to form a solid figure.

surf-ID# is the user-defined surface ID# for the surface;

REVOLUTION (or **REV** or **R**) is the keyword identifying the surface;

number-points is the number of line endpoints which follow;

$x'_1 r'_1 \quad x'_2 r'_2 \quad \dots \quad x'_n r'_n$ are the line endpoints, where r' is the perpendicular distance from the point to the x' -axis. Alternatively, the line endpoints may be given as (r, θ) pairs and the code will convert them to

$x' = r \cos \theta$; $r' = |r \sin \theta|$.

Note: The x' values of the points must be given in increasing order. If the first point does not have a zero radius, the figure will be bounded by a plane at the first value of x' . A similar termination is also done for the last given value of x' .

Alternatively, the surface points are specified as a set of (r, θ) pairs, if the word **POLAR** appears following *number-points*. For this case:

r = radial distance of the point from the surface definition origin;

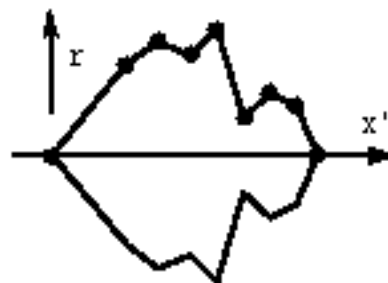
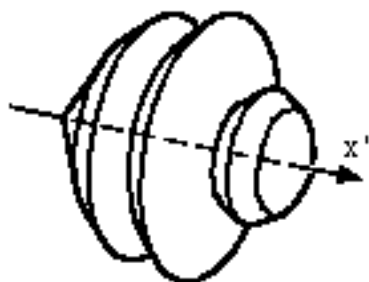
θ = polar angle, as measured from the positive x -axis to r .

These polar coordinates are converted to equivalent x', r' pairs, via

$$x' = r \cos(\theta)$$

$$r' = r$$

Successive x' values must be in increasing order.



Example of a Surface of Revolution defined by seven pairs of points.

```
64 REVOLUTION 7  -2.4 0  -2.3999 1.5  -2 2.3  0 2.3
                0.5 1.0  1.7 1.0  2.4 0      TR 37.5 -2.5 0
```

3.10 Special Surfaces

3.10.1 General Analytic Surfaces

A surface may be defined by an analytic equation containing terms up to the fourth degree.

$$\text{surf-ID\#} \left[\begin{array}{c} \text{ANALYTIC} \\ \text{ANA} \\ \text{A} \end{array} \right] c_1 \text{ T-ID}_1 \quad c_2 \text{ T-ID}_2 \dots c_n \text{ T-ID}_n \text{ (TR)}$$

The analytic equation of the surface is of the form:

$$(a_1x^4 + a_2y^4 + a_3z^4) + (a_4x^3 + a_5y^3 + a_6z^3) + (a_7x^3y + a_8x^3z + a_9y^3z) + \dots a_n = 0$$

where only some of the coefficients a_j are non-zero.

The user need specify just the non-zero coefficients and the terms in which they occur.

surf-ID# is the user-defined surface ID# (integer) for the surface;

ANALYTIC (or **ANA** or **A**) is the keyword identifying the surface type;

c_j is a non-zero coefficient in the analytic equation defining the surface;

T-ID_j is an ID word that identifies the equation term to which c_j applies. For example, if c_j is the coefficient of the xy term, then **T-ID_j** is just the word XY . If c_j is the coefficient of the xy^2z term, then **T-ID_j** is the word $XYYZ$.

Any permutation of the letters in a **T-ID** is recognized as the same word: XYZ XZY YZX YXZ ZXY and ZYX all represent the xyz term.

Constant terms are represented by the **T-ID** word **CONSTANT**, or **CONST**, or, in cases where no misunderstandings could result, just by omitting the **T-ID** altogether. The same term may be entered more than once—in this case the code will sum the input coefficients for like terms.

Example. A sphere of radius 5 located at (5,-4,0) is defined by the following surface equation:

$$(x - 5)^2 + (y + 4)^2 + z^2 - 5^2 = 0$$

or expanding the equation:

$$X^2 - 10X + 25 + Y^2 + 8Y + 16 + Z^2 - 25 = 0$$

This could be specified to COG as follows:

```
10 ANALYTIC 1.0 XX -10.0 X 25.0 CONST 1.0 YY 8.0 Y
      16.0 CONST 1.0 ZZ -25.0 CONST
```

3.10.2 Topographical Surface

surf-ID# **[TOPOGRAPHICAL
TOP]** $X = x_1 \ x_2 \ x_3 \ x_4 \ \dots \ x_n$
 $Y = y_1 \ y_2 \ y_3 \ y_4 \ \dots \ y_m$

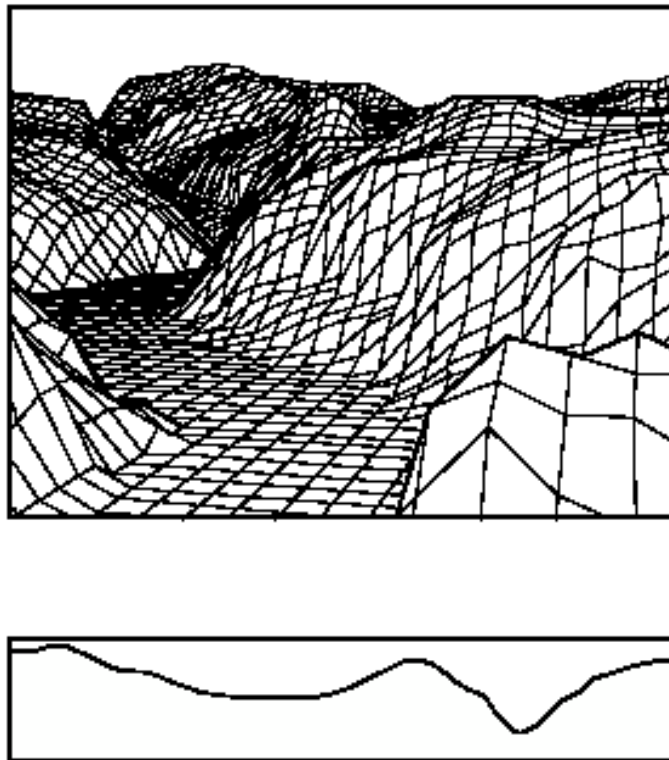
$Z = z(x_1, y_1) \ z(x_2, y_1) \ z(x_3, y_1) \ \dots \ z(x_n, y_1)$
 $z(x_1, y_2) \ z(x_2, y_2) \ z(x_3, y_2) \ \dots \ z(x_n, y_2)$
 $\dots$
 $z(x_1, y_m) \ z(x_2, y_m) \ z(x_3, y_m) \ \dots \ z(x_n, y_m)$

where:

surf-ID# is the user-defined surface ID# (integer) for the surface;

TOPOGRAPHICAL (or **TOP**) is the keyword identifying the surface type. This approximates a topographical surface by specifying an n-value **X**-grid (E-W direction) and an m-value **Y**-grid (N-S direction), and then specifying the z-value (altitude) at every intersection point of the *x*- and *y*-grid. The surface fit in each section of this grid assumes that the curve is linear in *x* and, independently, linear in *y*. This yields a second-order surface valid within each section of the defined grid.

Note: No rotation or translation (TR) is allowed with this figure.



4 The GEOMETRY Data Block

In the geometry Data Block, the user must specify all the volumes (SECTORS) that constitute the three dimensional geometry of the problem. The basic COG volume element is the SECTOR, which is a volume bounded by surfaces described in the SURFACES Data Block. The sector is the smallest volume element of COG geometry. You specify a sector by giving a sector equation, which names each of the surfaces bounding the sector (i.e. those surfaces forming the faces of the volume) and states on which side of each surface the sector lies (i.e. which half-space forms the volume interior). The sector volume is defined by the intersection of the specified surface half-space volumes. Experience will indicate how finely the geometry of any specific problem must be described to yield satisfactory results. The GEOMETRY Data Block is started with the word:

GEOMETRY

and followed by specifications of SECTORS, FILLS, BOUNDARIES, and UNITS.

4.1 Sector Definitions

The format for a SECTOR is: **SECTOR** *sect-ID# name ±surf-ID#1
±surf-ID#2 . . .*

where:

sect-ID# is a numerical label assigned by the user to identify this sector. The ***sect-ID#*** can be any integer from minus one to 999999.

(A ***sect-ID#*** of -1 is predefined as a perfect absorber, and a ***sect-ID#*** of 0 is predefined as a perfect vacuum; these special ***sect-ID#s*** are normally not used);

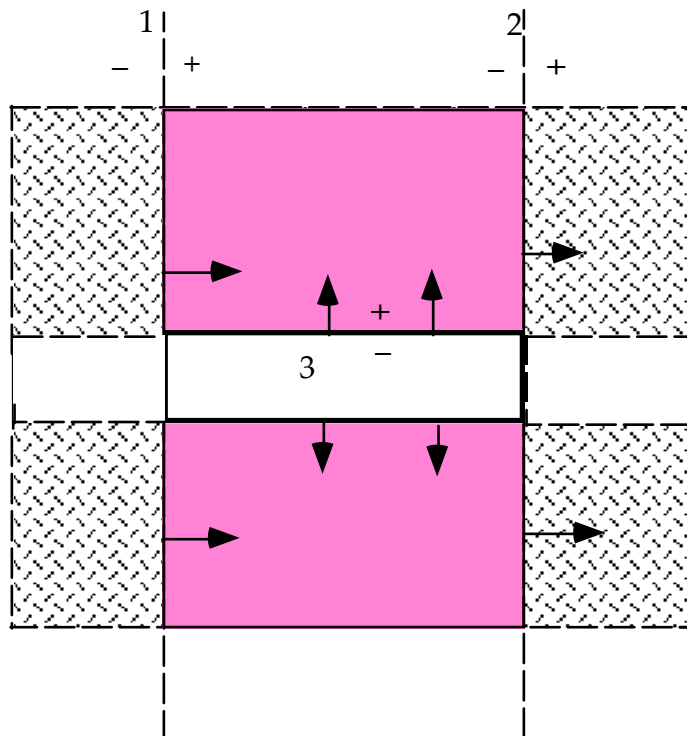
name = user-assigned 8-character name for the sector;

±surf-ID#1 ±surf-ID#2 . . . constitute the defining ***sector equation***. This is just a list of the surf-ID#s of all the surfaces which bound the sector. Each surf-ID# is prefaced with a sign ($\pm$) to indicate on which side of the surface the sector volume lies. The sector volume is defined as the ***logical intersection of the signed half-space volumes***.

Example:

```
SECTOR 10 CUBE -1 -2 -3 +4 +5 +6
```

specifies that the volume known to the user as #10, "cube", lies on the negative sides of surfaces 1, 2, and 3, and on the positive sides of surfaces 4, 5, and 6. (Plus signs may be omitted.) These surfaces must have been previously defined by statements in the SURFACES Data Block. (See **Positive and Negative Sides of Surfaces** for more information on signed surfaces.)



Example: We wish to specify a finite cylindrical volume. We assume we have already given a description in the SURFACES Data Block of two planes (with surf-ID#s 1 and 2) and a cylinder (surf-ID# 3). All surfaces are of unbounded extent. A sector definition for this finite cylinder would be:

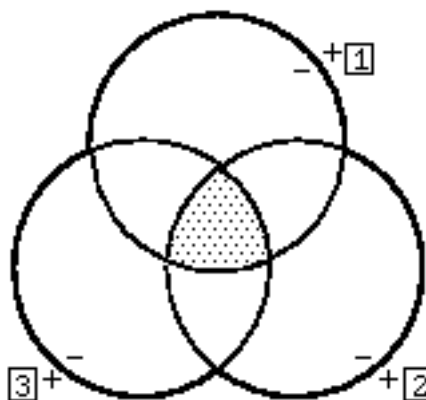
```
SECTOR 6 A_CYL 1 -2 -3
```

This sector equation says that our desired cylindrical volume lies in the positive (right) half-space of plane 1, in the negative (left) half-space of plane 2, and in the negative (inside) half-space of cylinder 3; i.e., the sector is formed from the intersection of these half-spaces.

Careful thought needs to be given in defining sectors so that there are no ambiguous sectors or multiply-defined sectors. Each point of a user's space must fall in one and only one sector. Many of the mathematical surfaces which COG provides are infinite in extent. The user will have to specify other bounding surfaces in order to form a sector of finite volume.

Internal code operations, invisible to the user, re-order the sequence of surfaces that define a sector so that those most quickly calculated occur first in the list. This saves running time when determining if a particle is inside a given sector. In the same manner, the code learns which sectors are the most probable ones to be entered when a particle leaves a given sector through a given surface. The search for a new sector starts with these most-likely sectors. Therefore, in setting up a problem, no particular thought needs to be given to the ordering of sectors or to the ordering of surfaces which bound sectors.

In the example below, the SECTOR that is within all three spherical surfaces, named insideit, would be defined as:



SECTOR 7 insideit -1 -2 -3

As we mentioned above, no point may fall in more than one sector.

Example: *If we specified these two overlapping sectors:*

SECTOR 7 INSIDE -1 -2 -3

SECTOR 8 ERR7 -2

COG would produce an error message, since sector 8 and sector 7 have some of the same volume included in each. The best way to check for overlapping sectors is to use the PICTURE statement

4.1.1 The OR (Union) Operator

In many cases it is necessary to combine several, often overlapping, SECTOR definitions into one, in order to describe a complex part. To accomplish this, the sector definition is augmented by the OR (union) operator. This is of form:

```
SECTOR sect-ID# name  $\pm$ surf-ID#1  $\pm$ surf-ID#2 . . .
      OR  $\pm$ surf-ID#11  $\pm$ surf-ID#12 . . .
      OR  $\pm$ surf-ID#21  $\pm$ surf-ID#22 . . .
      . . .
```

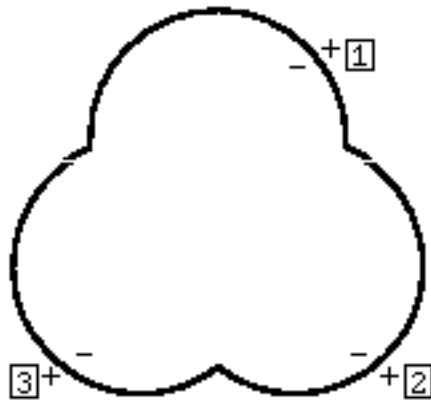
The **OR** operator causes the total sector definition to be the union of all the separate pieces. These individual OR volumes may overlap if desired, without causing any error.

The OR operator applies to all *surf-ID#s* following the OR until terminated by another OR statement or the next SECTOR definition.

Example: Consider the three overlapping spheres defined in the prior example by surfaces 1, 2 and 3. The statement:

```
SECTOR 7 INSIDE3 -1 OR -2 OR -3
```

specifies that SECTOR 7 is the union of the three spherical volumes.



Example of a more complex geometry: This example geometry consists of 3 horizontal cylinders, 1, 2 and 3, and one vertical cylinder, 4, all enclosed by spherical surface 10. In the figure below, surfaces are labeled with numbers (surf-ID#s) near them, while sectors are labeled with circled numbers (sect-ID#s).

Sector 1 is the inside of cylinder 2, bounded by planes 1 and 3.

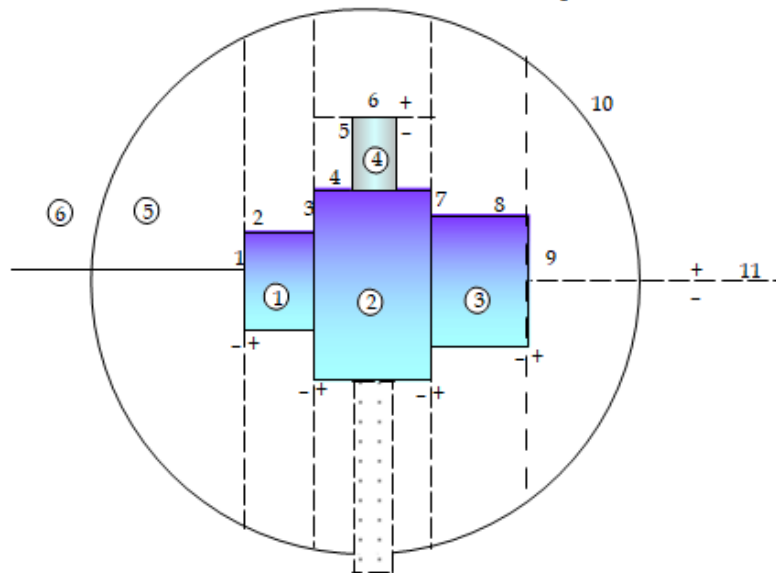
Sector 2 is the inside of cylinder 4, bounded by planes 3 and 7.

Sector 3 is the inside of cylinder 8, bounded by planes 7 and 9.

Sector 4 is the inside of cylinder 5, bounded by planes 4 and 6.

Sector 6 is the volume outside of sphere 10.

Sector 5 is the volume between sectors 1-4 and sphere 10.



```

SECTOR 1 LEFTCYL      +1 -2 -3  $ FAR LEFT CYL BETWEEN 1 & 3
SECTOR 2 MIDCYL       +3 -4 -7   $ MIDDLE CYLINDER BETWEEN 3 & 7
SECTOR 3 RIGHTCYL     +7 -8 -9   $ FAR RIGHT CYLINDER BETWEEN 7 & 9
SECTOR 4 TOPCYL       -6 -5 +4 +11 $ TOP CYL IN TOP HALF SPACE (+11)
                                   $ ABOVE 4 & -6
SECTOR 6 OUTSIDE +10      $ WORLD OUTSIDE OF SPHERE
SECTOR 5 LEFTOVER -10 +9 $ FAR RIGHT SECTOR
    OR  -10 +1 +2 -3      $ SECTOR FROM 1 TO 3 OUTSIDE 2
    OR  -10 +3 +4 -7 -11  $ LOWER MID SECTOR ABOVE 4 FROM 3-7
    OR  -10 +7 +8 -9      $ SECTOR FROM 7 TO 9 OUTSIDE 8
    OR  -10 +11 -6 +5 +4  $ SECTOR AROUND TOP CYL BELOW 6
    OR  -10 +3 -7 +6      $ SECTOR ABOVE TOP CYL BET. 3 AND 7
    OR  -10 -1            $ FAR LEFT SECTOR

```

***Note:** For sector 4, we need an auxiliary plane (11) to confine the definition of sector 4 to the upper half space. Without this constraint it is also defined as an infinite cylinder on the other side of 4 (dotted area in drawing).*

Caution: Surfaces, being mathematical constructs, are often infinite in one or more dimensions and may well extend where they are not wanted. The solution to unwanted surface extension is to add an auxiliary plane (or other surface) that constrains the definition to just the volume desired.

4.1.2 The NOT (Exclusion) Operator

It is sometimes useful to describe a sector as a volume that excludes another specified volume(s). This exclusion volume can be defined explicitly, in terms of its bounding surfaces, or implicitly, in terms of other previously-defined sectors.

Explicit NOT Operator

The explicit form is given by a standard sector definition followed by the exclusion volume defined in terms of its bounding surfaces:

```
SECTOR sect-ID# name ±surf-ID#1 ±surf-ID#2 . .
                        NOT ±surf-ID#11 ±surf-ID#12 . . .
                        NOT ±surf-ID#21 ±surf-ID#22 . . .
                        . . .
```

The NOT operator causes the total sector definition to be the intersection of the standard sector volume with the complement of each of the associated NOT volumes. The logical equivalent of this operation is, where A,B,C are defined volumes:

A AND (NOT B) AND (NOT C) ... , meaning that a particle will be found in the sector only if it is inside A and not inside B and not inside C.

NOT volumes apply to the immediately preceding standard sector volume definition. There may be as many NOT volume s as desired.

Implicit NOT Operator

The implicit form is given by a standard sector definition followed by the exclusion volume described in terms of previously-defined sectors:

```

SECTORsect-ID#    name    ±surf-ID#1    ±surf-ID#2    .    .
NOT SEC ±sect-ID#11
NOT SEC ±sect-ID#21    .    .    .
.    .    .

```

Here the keyword S E C immediately follows keyword NOT.

Note that sectors *sect-ID#11* and *sect-ID#21* must have been defined earlier in the GEOMETRY Data Block (no forward references allowed, to avoid loops). The referenced volumes are allowed to have NOT volumes of their own.

The two types of NOT operators may be intermixed in a sector definition.

Example: Consider a simple system of three nested spheres:

SURFACES

102 SPHERE 2.

103 SPHERE 3.

104 SPHERE 4.

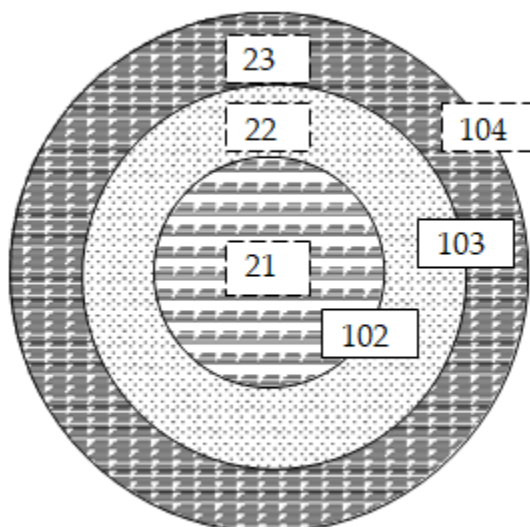
GEOMETRY \$ STANDARD COG SECTORS

SECTOR 21 SPH1.2 -102 \$ SPHERE OF RADIUS 2

SECTOR 22 SHELL23 +102 -103 \$ SHELL BETWEEN R=2,3

SECTOR 23 SHELL34 +103 -104 \$ SHELL BETWEEN R=3,4

BOUNDARY VACUUM +104



Using the NOT operator, the sector definitions can alternatively be written as:

- Explicit form:

GEOMETRY

SECTOR 21 SPH1.2 -102

SECTOR 22 SHELL23 -103 NOT -102

SECTOR 23 SHELL34 -104 NOT -103

BOUNDARY VACUUM +104

- Implicit form:

GEOMETRY

SECTOR 21 SPH1.2 -102

SECTOR 22 SHELL23 -103 NOT SEC 21

SECTOR 23 SHELL34 -104 NOT SEC 21 NOT SEC 22

BOUNDARY VACUUM +104

Note that computational efficiency will decrease as the number of surfaces COG must examine to locate a particle in the geometry increases. The explicit form of the NOT operator should not entail more surfaces than the standard sector definition, but the implicit form may well do so.

4.2 Void and Infinite–Absorber Sectors

COG recognizes two special *sect-ID#*s:

sect-ID# = -1 signifies that the sector contains: ***an infinite absorber***.

sect-ID# = 0 signifies a void or perfect vacuum : ***zero absorber***.

An ***infinite absorber*** is a fictitious material with an infinitely large absorbing cross section. Any particle entering the region will be terminated automatically by the code, and any point-flux estimation made through such a region will result in a zero contribution.

A ***void*** is a region of vacuum with zero cross section and the particle passes through it without interaction.

4.3 FILL Sectors

A complete geometry specification describes every volume of the user's space. It frequently is tedious and/or difficult to describe all the small "interstitial" volumes between the major parts of the geometry. COG helps you out by assigning all undefined volumes in your problem to a special default sector called FILL, which a default *sect-ID#* = 0 – a *void*. If this condition is satisfactory, you need not specify an alternative FILL material. *The FILL sector always exists*, even if you don't specify it.

If you like, you can use some other material as a FILL for all undefined volumes by specifying:

FILL *mat-ID#*

This tells COG to use the MIX Block material *mat-ID#* for the FILL material. Only one such FILL statement is normally allowed per problem (see UNIT definition for exception).

Default: FILL *mat -ID#* = 0, a void. *See the MIX Data Block for information on how to specify materials in your problem.*

Example of specifying a FILL Material:

Here, the MIX Data Block associates Material "air" with mat-ID# 2.

The FILL statement specifies that COG will use Material 2.

```

GEOMETRY
  SECTOR  10  AL          -1  2  -3          $ AL SCATTER PLATE
  SECTOR  11  NOT_HOLE   -7  4  -5  6        $ AREA OUTSIDE APERTURE
  SECTOR  12  HOLE       -6  4  -5          $ APERTURE
  BOUNDARY VACUUM  1          $ VACUUM BOUNDARY OUTSIDE 1
  FILL  2                $ USE MATERIAL 2 (AIR) FOR FILL

MIX
  MAT 1  AL  2.7          $ ALUMINUM
  MAT 2 AIR 1.29E-3    $ AIR IS ASSIGNED A MATERIAL NUMBER OF 2
  MAT 3  TA  16.6         $ TA PLATE
ASSIGN-M          $ ASSIGN SECI TO MATI, ETC.
  10  1          $ ASSIGN MATERIAL 1 (AL) TO SECTOR 10
  11  3          $ ASSIGN MATERIAL 3 (TA) TO SECTOR 11
  12  2          $ ASSIGN MATERIAL 2 (AIR) TO SECTOR 12
  20  2        $ ASSIGN MATERIAL 2 (AIR) TO SECTOR 20

```

*An example of how you can assign different FILLs to different regions of the geometry. In this case, you would have to use the UNIT Data Block (see **Geometry UNITS** for details):*

```

DEFINE  UNIT  4
  SECTOR  17  BB1  -1  2  3  -4
  SECTOR  18  BB2  -5  2  3  -4
  FILL  3                $ FILL FOR UNIT #4
DEFINE  UNIT  7
  SECTOR  20  CC1  -7
  SECTOR  21  CC2  8  -9
  SECTOR  22  CC3  -8  9
  FILL  1                $ FILL FOR UNIT #7

```

Fill Note: The disadvantage to using fill is that run time is increased. COG has a learning mechanism which allows it to quickly find the next sector following a boundary crossing. But if either the prior or the new sector is FILL, then COG must check every sector in the current unit level to find the containing sector.

4.4 Specifying Boundary Conditions

A boundary condition imposes a constraint on the problem being run. The type of the boundary will determine what happens to a particle that hits it.

4.4.1 Boundary Vacuum

The random walk of a particle ceases if it hits a BOUNDARY VACUUM. This event tells the code that the particle is leaving the user-defined geometry, never to return. When this occurs, COG terminates the particle trajectory and computes detector scores. In most problems, the use of the vacuum boundary condition saves calculational time by not tracking particles beyond regions of interest. If desired, a source can be specified in a vacuum boundary inside the vacuum. The code lets the particles move as if they were in a vacuum, and will commence tracking them if they enter the defined geometry.

The user specifies the BOUNDARY VACUUM as the volume lying outside and bounding the user's problem geometry. This volume is defined in the same way as for a SECTOR. The appropriate statement is:

```
BOUNDARY VACUUM      ±surf-ID#1 ±surf-ID#2 . . .
                      { OR ±surf-ID#11 ±surf-ID#12 . . . }
                      . . .
```

Example of a BOUNDARY VACUUM specification:

```
$      TWO NESTED SPHERES
SECTOR 21 SPH1      -10
SECTOR 22 SPH2   +10 -20
BOUNDARY VACUUM +20 $ THE VOLUME OUTSIDE SECTOR 22
```

If necessary, more than one boundary specification may be made.

If the FILL material for the problem is defaulted to void, a default vacuum boundary condition exists at the outside of each problem. But as noted in the Fill description, the problem will run faster if a BOUNDARY VACUUM is explicitly inserted into the Geometry.

Note: COG **requires** that each problem be bounded by a BOUNDARY VACUUM, or alternatively that a void fill material be specified for the problem. This ensures that particles will not enter a physical medium of unbounded extent, which would cause COG to expend an indefinite amount of useless computational effort tracking these particles.

Note: The user needs to ensure that his defined sectors don't penetrate the BOUNDARY VACUUM, or COG can waste time tracking particles inside the BOUNDARY VACUUM sector.

4.4.2 Boundary Reflecting

A reflecting boundary produces specular (mirror) reflection of incident particles back into the problem, with no change to particle weight or energy. Specular reflection is defined so that the returned particle is reflected at the same angle from the normal to the boundary as the incident particle. Hence, the incident direction, the normal, and the reflected direction all lie in the same plane.

In a physical sense, this boundary lies at a place where the same particle flux distribution exists on each side of the boundary. Thus, if your problem is known to have a symmetrical flux about a surface, you may be able to restrict the calculation to volumes on one side only of the surface, and save computation time.

The BOUNDARY REFLECTING is specified like a sector, and describes the volume on the outside of your reflecting surfaces.

The specification for the BOUNDARY REFLECTING is:

BOUNDARY REFLECTING $\pm surf-ID\#_1$ $\pm surf-ID\#_2$...
 {OR $\pm surf-ID\#_{11}$ $\pm surf-ID\#_{12}$...}
 ...

A source placed outside a reflecting boundary will yield a **fatal error** message and terminate the problem.

Example of a BOUNDARY REFLECTING specification.

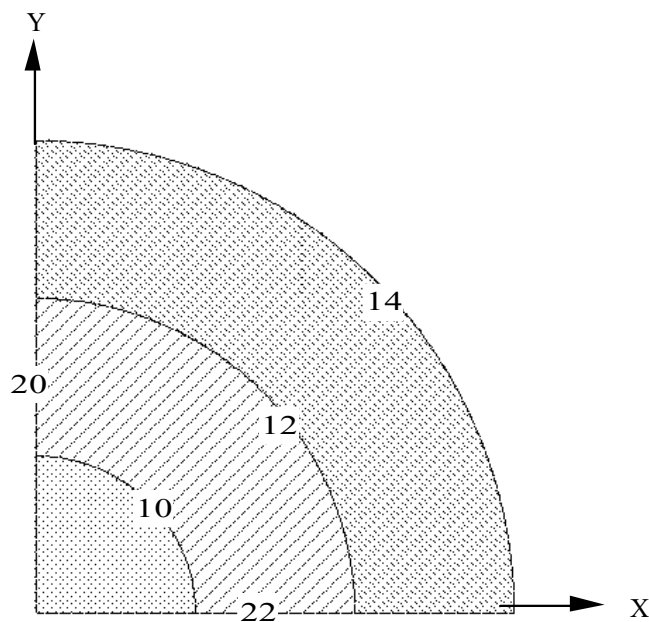
```

SURFACES
$ RADIAL ZONES
 10 CYL Z 10.  -20. 20.
 12 CYL Z 20.  -20. 20.
 14 CYL Z 30.  -20. 20.
$ REFLECTING PLANES
 20 PLANE X 0.
 22 PLANE Y 0.

GEOMETRY
SECTOR 10 ZONE1 -10      +20 +22
SECTOR 12 ZONE2 +10 -12  +20 +22
SECTOR 14 ZONE3 +12 -14  +20 +22

BOUNDARY VACUUM +14      +20 +22 $ FOR R > 30
BOUNDARY REFLECTING      -20 OR -22
$ DESCRIBES THE VOLUME TO THE LEFT OF PLANE 20 OR BELOW PLANE 22
$ PARTICLES REFLECT AT THE SURFACE(S) OF THE BOUNDARY VOLUME

```



This quarter-round geometry with reflecting planar boundaries is equivalent to a 360 - degree geometry.

Warning: Any COG surface can be used as a REFLECTING BOUNDARY, but you are almost sure to get the wrong answer unless the reflecting surface is a plane. If inserting the reflection surface into your “whole” problem would change the angular flux at that location, then it cannot be used.

Warning: COG imposes a restriction on BOUNDARY REFLECTING problems when a POINT DETECTOR is used. The answers so calculated are nearly always *wrong*. The point flux estimator looks at each collision site in the random walk and makes an estimate of the contribution to the flux at the specified point. If there is a reflecting boundary in the problem, collisions on the other side of the boundary are **not** evaluated. The point detector results will be *wrong* (too low) because of these missing collisions. If there were always just one boundary, and it were always a plane, it would be simple to have the code evaluate the point estimation correctly. However, it is possible to specify many such boundaries in one problem; there are, consequently, many difficulties in making the code operate correctly under such cases. When a problem contains both reflecting boundaries and point detectors, an error warning is printed to remind the user of these facts. If the contribution from the missing volume would be small, the problem solution could be acceptable. Users must decide for themselves.

4.4.3 Boundary Periodic

The periodic boundary is specified by:

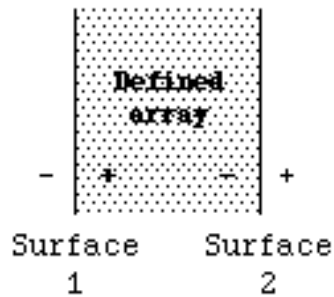
BOUNDARY *surf-ID#_p* **PERIODIC** $\pm s_{c1}$ { $\pm s_{c2}$...} {**OR** ...}

where *surf-ID#_p* specifies a surface (previously defined in the SURFACES Data Block) which will become a periodic boundary.

$\pm s_{c1}$ specifies the *surf-ID#* of a surface which will become a *conjugate periodic boundary*.

Whenever a particle penetrates a *conjugate periodic boundary*, by moving to the signed side of the surface *s_{c1}*, the particle is translated to surface *surf-ID#_p*, at the point where the normal to the conjugate surface, drawn through the particle penetration point, hits the periodic boundary.

The periodic boundary is used occasionally in criticality problems when the multiplication of an infinite number of stacked arrays is desired. A simple one-dimensional problem would look like:



Periodic boundaries are defined on the left (the negative half-space of surface one), and on the right (the positive half-space of surface two) thusly:

```
BOUNDARY 1 PERIODIC +2
BOUNDARY 2 PERIODIC -1
```

Whenever a particle penetrates surface 2 (the surface conjugate to boundary 1), it is moved to a point on surface 1 where the normal to the struck boundary intercepts surface 1. The direction, weight, and energy of the particle are unchanged, and the particle is allowed to continue on its trajectory. Likewise, when a particle penetrates surface 1 (the surface conjugate to boundary 2), it is moved to surface 2.

The user may extend this example to four or six planes or to surfaces that are not planes. **Fatal errors** are encountered if a source is located outside the boundary or if point detectors are used.

4.5 Geometry UNITS

4.5.1 The UNIT Definition

A large or complex problem can require hundreds of SECTOR statements to fully define the GEOMETRY. In many cases, the COG geometry setup can be considerably simplified by using UNITS, or geometric sub-assemblies. The UNIT may be thought of as a "packaged" part or sub-assembly, with its own internal structure. Each UNIT is defined separately from all other parts. Once defined, it may be used repeatedly in the problem geometry in different orientations or locations.

UNITS have other advantages as well. Complex geometries can be subdivided into smaller and easier-to-use sub-sections. Major components, such as accelerator sections or diagnostic lines-of-sight, often have their own "natural" axis or center-line. Describing such a component in COG terms is greatly simplified if the component axis lies along a COG coordinate axis. The UNIT feature allows any component to be described using its "natural" or most convenient coordinate system. This UNIT or sub-assembly can then be translated and rotated into its proper position in the problem geometry. This is analogous to building some piece of apparatus on your workbench, where construction is most convenient, then installing it, as an assembly, where it is to be used. Once a sub-assembly or UNIT is described it can be re-used as many times as needed in different locations. UNITS also allow a different FILL material to be assigned in each UNIT, which is often an advantage.

A UNIT is just a collection of the same kind of SECTOR and FILL statements which are used in ordinary COG geometry setup. The only difference is that the UNIT geometry does not automatically become part of the problem geometry. The user must provide one or more USE UNIT statements to insert a DEFINED UNIT into his problem.

The UNIT DEFINITION has the form:

```
DEFINE UNIT unit-ID#1
SECTOR . . .
FILL . . .
USE UNIT UNIT-ID#2 . . .
. . .
```

where:

DEFINE UNIT is a keyword denoting the start of the UNIT definition;

unit-ID#₁ is a UNIT identification number (integer), assigned by the user to identify this UNIT;

This is followed by the same kind of **SECTOR** and **FILL** statements used in an ordinary COG GEOMETRY Data Block. As shown above, the UNIT definition may include other UNITS, via the USE UNIT statement. Here, *unit-ID#₂* is the UNIT identification number of another DEFINED UNIT.

UNITS may be nested to a level of 50. The UNIT definition is terminated by another DEFINE UNIT statement, a PICTURE statement, or by the end of the GEOMETRY Data Block.

Note: BOUNDARY statements are *not* allowed in a DEFINE UNIT definition.

Note: A UNIT's geometry does not automatically become part of the problem geometry. The user must employ a USE UNIT statement to insert a DEFINED UNIT into the problem.

Note: Electrons and protons cannot be transported inside UNITS.

Example of UNIT DEFINITIONS:

```

GEOMETRY
  SECTOR    1    ...
  SECTOR    2    ...
  USE UNIT  7    ...      $ INSERTS UNIT 7 (AND NESTED UNIT 4)
                           $  INTO PROBLEM
      . . .
      . . .
DEFINE UNIT  4          $ START DEFINITION OF UNIT 4
  SECTOR    17    ...
  SECTOR    18    ...
  FILL      3          $ FILL FOR UNIT 4
DEFINE UNIT  7          $ TERMINATES UNIT 4 DEF,BEGINS UNIT 7 DEF.
  SECTOR    20    ...
  SECTOR    21    ...
  USE UNIT  4    ...    $ NESTED UNIT
  SECTOR    24    ...
PICTURE CS MATERIAL ...$PICTURE STATEMENT TERMINATES UNIT 7 DEF.

```

Once you have created a UNIT, how do you insert it where you want? The basic idea is to specify a "shell" or bounded volume in your problem and insert the UNIT inside the shell. If you specify a shell smaller than the UNIT, the UNIT will be "clipped" to the size of the shell. UNITS may be moved, rotated, inserted and reused anywhere in the problem geometry you desire, even inside other UNITS. The following format (identical in form to the SECTOR definition) is used to insert a DEFINED UNIT into the problem geometry:

where

unit-name = user-assigned 8-character name for this USE UNIT instance;

Note: the unit-instance bounding surface must be closed; i.e., a line drawn in any direction from any point of the unit must encounter one of the surfaces specified in the USE UNIT statement.

defines a Translation/Rotation specification for the UNIT, which is identical in form to the one used for SURFACES. The TRU specification will be applied to move the UNIT to the desired location and orientation in the problem. (See Section **SURFACE Data Block: Translation/Rotation (TR) of Surfaces** for more details).

Note: The USE UNIT's TRU statement does *not* apply to the shell-forming SURFACES used in the USE UNIT statement, but *only* to the DEFINED UNIT. The shell-forming SURFACE locations are determined by their definitions in the SURFACES Data Block (including any possible TR statements there).

*Example of using one DEFINED UNIT in several places in a problem. Note that the TRU statement only applies to the location of DEFINED UNIT 100 and **not** to SURFACES 300, 301, etc.*

```

SURFACES
$-----THESE ARE THE SURFACES USED IN UNIT 100-----
  100 BOX  156.44  1.63575  1.63575  TR  78.22001  0  0
  101 CYL X              0.602996
  102 CYL X              0.521716
  103 CYL X              0.514858
  104 P X                0.318
  105 P X                153.62
  106 P X                156.122
  107 P X                156.44

$--THESE ARE THE SURFACES USED TO FORM "SHELL" SECTORS FOR
$      INSERTING UNIT 100 INTO-----
300 BOX  156.44  1.635755  1.635755
              TR  78.22001  18.81124  -2.45364
301 BOX  156.44  1.635755  1.635755
              TR  78.22001  18.81124  -0.81788
302 BOX  156.44  1.635755  1.635755
              TR  78.22001  18.81124   0.81788
303 BOX  156.44  1.635755  1.635755
              TR  78.22001  18.81124   2.45364
304 BOX  156.44  1.635755  1.635755
              TR  78.22001  18.81124   4.08940

$-----DEFINITIONS OF SECTORS AND UNITS-----
GEOMETRY
  USE UNIT  100  ELE300   -300  TRU  0  18.81124  -2.45364
  USE UNIT  100  ELE301   -301  TRU  0  18.81124  -0.81788
  USE UNIT  100  ELE302   -302  TRU  0  18.81124   0.81788
  USE UNIT  100  ELE303   -303  TRU  0  18.81124   2.45364
  USE UNIT  100  ELE304   -304  TRU  0  18.81124   4.08940
DEFINE UNIT 100
  SECTOR 110 WATER2 -100 +101
  SECTOR 111 CLAD  -101 +102 +12  -107
  SECTOR 112 WATER3 -102 +103 +104 -105
  SECTOR 113 FUEL   -103 +104 -105
  SECTOR 114 AL1    -102  +12 -104
  SECTOR 115 WOOL   -102 +105 -106
  SECTOR 116 AL2    -102 +106 -107

```

In each USE UNIT instance, the DEFINED UNIT 100 is translated according to the TRU statement and inserted into the specified "shell" SURFACES (300,301, etc.). The "shell" surface(s) must form a completely bounded volume (here a box). Note that we have used the TR statement in each SURFACE statement to move the "shell" SURFACES 300, 301, ... into position in the problem. After translation, each of these BOXes extends from $x = 0$ to $x = 156.44$. Only that part of UNIT 100 which fits inside each "shell" BOX will appear in the problem.

When COG processes a USE UNIT statement, it takes the DEFINED UNIT (with all its sub-structure, including perhaps other nested UNITS), and performs the TRU operation specified. COG then inserts into the problem, as much of the DEFINED UNIT as will fit inside the shell specified in the USE UNIT statement. In other words, the shell acts as a clipping boundary for the DEFINED UNIT. When a particle moves from outside the shell to the inside, it enters the DEFINED UNIT geometry at the point specified by the intersection of the trajectory with the shell. The coordinates of the particle are then transformed (through the inverse TRU operation) into the UNIT's coordinate system of definition. Tracking continues within the UNIT until the particle passes out through the USE UNIT shell (or until it enters a nested UNIT).

In the definition of a DEFINE UNIT, you can USE one or more other DEFINE UNITS. Thus, UNIT 4 could be defined as:

```

DEFINE UNIT 4
  SECTOR      17  BB1  . . .
  SECTOR      18  BB2  . . .
  USE UNIT 30 unit30 -300      TRU  0 112. -22.
  . . .
  FILL        3

```

The user would, of course, have to provide a definition for UNIT 30. UNITS may be nested 50 levels deep. When an error occurs in the geometric setup, the level number is also printed in the resulting fatal error statement. In COG terminology, level-0 is the level of the normal problem geometry. If a particle enters a UNIT USED in the level-0 geometry, it passes to level-1. If the particle enters another UNIT nested within the first one, it passes to level-2, etc.

One consequence of using UNITS is that sectors with the same *sect-ID#s* will appear at multiple locations in the problem. If the user defines a detector to include any UNITized sector, then that detector will score particles for all of the sector instances. If separate detector results are wanted for identical sub-assemblies in different locations, UNITS will **not** work. The detector REGIONS will have to be given different *sect-ID#s/region-ID#s* and explicitly treated as separate entities.

***Example** of a UNIT inserted multiple times into a problem geometry, and clipped by the USE UNIT shell. UNIT 4 consists of a sphere surrounded by a specified fill material, which extends outward indefinitely. We have two cylindrical annuli*

(SECTOR 10, on the left, and SECTOR 11, on the right) into which we want to insert the UNIT. We specify the finite cylinders 2 and 4 as the 'shell' surfaces to bound the UNIT instances.

```

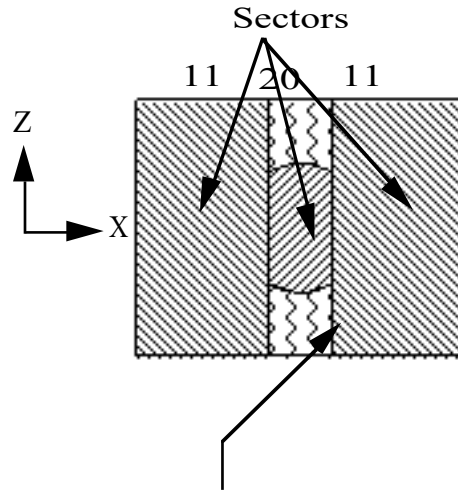
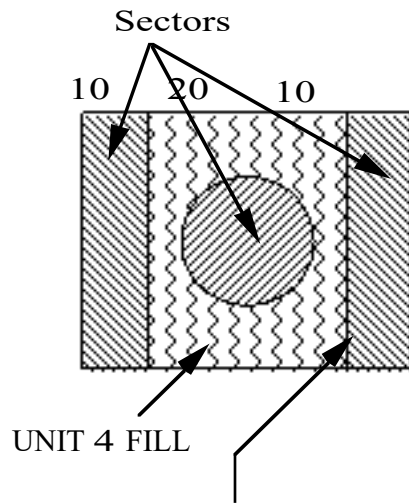
SURFACES
$ INCLUDES SURFACES FOR MAIN GEOMETRY AND UNITS.
$ MAIN GEOMETRY SURFACES:
  1 CYL Z 5.  -4. 4. $ CYL ALONG Z, R = 5, ENDPLANES AT +-4.
    TR -8. 0. 0. $ TRANSLATED TO X = -8.
  2 CYL Z 3.  -4. 4. $ CYL ALONG Z, R = 3, ENDPLANES AT +-4.
    TR -8. 0. 0. $ TRANSLATED TO X = -8.
  3 CYL Z 5.  -4. 4. $ CYL ALONG Z, R = 5, ENDPLANES AT +-4.
    TR  8. 0. 0. $ TRANSLATED TO X = 8.
  4 CYL Z 1.  -4. 4. $ CYL ALONG Z, R = 1, ENDPLANES AT +-4.
    TR  8. 0. 0. $ TRANSLATED TO X = 8.
$ UNIT SURFACES:
  5 SPHERE 2.          $ SPHERE AT ORIGIN, RADIUS = 2.

GEOMETRY
SECTOR 10 LEFTCYL -1 +2    $ LEFT CYLINDRICAL ANNULUS
SECTOR 11 RIGHTCYL -3 +4   $ RIGHT CYLINDRICAL ANNULUS
USE UNIT 4 LEFT -2$ USE UNIT 4 INSIDE CYLINDER 2
    TRU -8. 0. 0.          $ BUT TRANSLATE UNIT FIRST TO
                           $ X = -8.
USE UNIT 4 RIGHT -4        $ ALSO USE UNIT 4 INSIDE CYLINDER 4
    TRU  8. 0. 0.          $ BUT TRANSLATE UNIT FIRST TO
                           $ X = +8.
FILL  0                    $ SPECIFY A FILL MATERIAL OF VOID

DEFINE UNIT 4
  SECTOR 20 SPHERE -5
  FILL  4                    $ SPECIFY A DIFFERENT FILL MATERIAL
                           $ FOR UNIT 4
  PICTURE CS M -17. 0. 6. -17. 0. -6. 17. 0. -6.
  MIX
    MAT = 1 H2O 1.
    MAT = 2 C 1.
    MAT = 3 AIR 0.001
    MAT = 4 H 0.001
  ASSIGN-M                  $ ASSIGN TO EACH SECTOR,
                           $ A MATERIAL NUMBER

  10 1 11 1
  20 2

```



Note the clipping action of the bounding 'shell' in the USE UNIT statements. Both cases are legal. You must check that the resulting physical model is what you want.

4.5.3 The UNIT FILL

Each UNIT will have its own particular FILL, which may be set by the user to a desired material, or defaulted to void. FILL exists in the "interstitial" gaps between defined sectors in the UNIT and extends outward to where it is clipped by the confining shell of the USE UNIT statement. To specify a FILL for any UNIT, use the form:

FILL *mat-ID#*

inside the UNIT DEFINITION.

mat-ID# is the material number of a material in the MIX Data Block. The previous examples showed different FILL statements in different UNITS. **The FILL sector always exists in each UNIT**, even without user specification and will be assigned a *default sect-ID# = 0*; thus, the fill sector is a void.

Default: FILL *sect-ID#* = 0, a void.

4.6 General Lattice Feature

A pre-processor has been added to COG to read and process a string of data between the braces {} within a UNIT definition to generate and automatically run a new input file with the suffix *.cnvrt*.

4.6.1 Rectangular Lattice

If the first symbol in the {} is **x** then the data represents a rectangular lattice which contains a UNIT containing a general three-dimensional lattice constructed of multiple USE UNIT statements centered between X, Y, and Z planes defined in the string in a specified arrangement.

The 'shorthand' input consists of...

x data, describing the x grid, followed by...

y data, describing the y grid, followed by...

z data, describing the z grid, followed by...

the word **fill**, followed by...

fill data - describing the lattice fill pattern, entered left-to-right, top-to-bottom.

The input is free form, in that there can be any number of words per line as long as the line length, including blanks, is less than 512 characters. Only the order is important.

Example 1: defines unit 6 as a 15X15 array of fuel (unit 1) and empty (unit 2) cells.

```
define unit 6
  {x 1501 16 [-6.00075 6.00075]
  y 1601 16 [-6.00075 6.00075]
  z 1701 -70. 70.
  fill

  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 2 1 1 2 1 1 1 2 1 1 2 1 1
  1 1 1 1 1 1 1 2 1 1 1 1 1 1 1
  1 1 1 1 2 1 1 1 1 1 2 1 1 1 1
  1 1 2 1 1 1 1 1 1 1 1 1 2 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 2 1 1 1 2 1 1 1 2 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 2 1 1 1 1 1 1 1 1 1 2 1 1
  1 1 1 1 2 1 1 1 1 1 2 1 1 1 1
  1 1 1 1 1 1 1 2 1 1 1 1 1 1 1
  1 1 2 1 1 2 1 1 1 2 1 1 2 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1}
```

Line 1 generates 16 x planes numbered from 1501 to 1516, equally spaced from -6.00075 to 6.00075.

Line 2 generates 16 y planes numbered from 1601 to 1616, equally spaced from -6.00075 to 6.00075.

Line 3 generates 2 z planes numbered 1701 and 1702 at -70. and 70.

The remaining lines fill the 15X15 array (left to right in x, top to bottom in y and z) with the units indicated.

Example 2: 3X3 array of unit 6, together with Example 1 forming a complete 45X45 assembly...

```
{x 1801 -18.00225 -6.00075 6.00075 18.00225
y 1901 -18.00225 -6.00075 6.00075 18.00225
z 2001 -70. 70.
fill 9*6}
```

This example shows that the surfaces can be entered either as a list (x 1801 -18.00225 -6.00075 6.00075 18.00225) or as an interpolating scheme (x 1501 16 [-6.00075 6.00075]). Surfaces entered as a list do not need to be equally spaced. In the fill input the multiplier is recognized, i.e., 9*6 is equivalent to 6 6 6 6 6 6 6 6 6.

Other fill options include...

A '-' in the place of a unit designator indicates that the lattice cell will be empty, i.e., no unit will be inserted in the cell. The cell may be filled with the usual COG fill option.

Example 3: fill '--' option

```
fill
6 6 6
6 - 6
6 6 6
```

8 cells of the 3X3 array are filled with unit 6, the center cell is empty.

A **r90**, **r180**, or **r270** following a unit designator indicates that the cell will be rotated 90, 180, or 270 degrees clockwise.

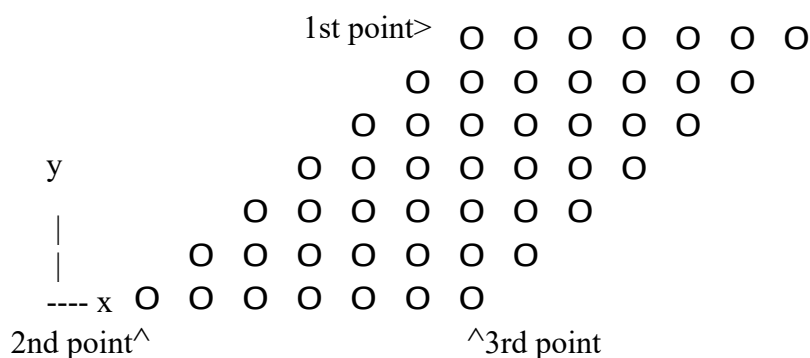
Example 4: fill **r** option

```
fill
6 5 6 r90
5 5 5
6 r270 5 6 r180
```

The 3X3 array will be filled, left to right, top to bottom, with unit 6, unit 5, unit 6 rotated 90 degrees, unit 5, unit 5, unit 5, unit 6 rotated 270 degrees, unit 5, and unit 6 rotated 180 degrees.

4.6.2 X-Oriented Triangular Lattice

If the first symbol in the *{}* is **tri-x** then the data represents an x-oriented triangular lattice. The following is a graphical representation of a 7X7 x-triangular lattice.



Four sets of plans are generated; equally spaced x-planes, equally spaced positive-slope planes, equally spaced negative-slope planes, and two or more bounding z-planes. These planes define one or more tiers of nested cells of hexagonal cylinders. The input required to define this type of lattice is.

x1 y1	\$ (x,y) of point 1, center of upper left cell
x2 y2	\$ (x,y) of point 2, center of lower left cell
x3 y3	\$(x,y) of point 3, center of lower right cell
nx ny	\$ number of cells per row, number of rows
nz z(1)...z(nz+1)	\$ number of tiers followed by nz+1 bounding z planes.
lx ly1 ly2 lz	\$ initial surface numbers for x, > 0 slope, < 0 slope, and z planes
fill	\$ keyword indicating the start of the fill data
nx*ny*nz entries	\$ unit#'s to fill lattice, left-to-right (in x), top-to-bottom (in y), \$ starting with 1st point, ending with the 3rd point, this is \$ repeated starting with bottom tier, proceeding to top tier (in z)

Note 1: this fill pattern different than for **Y-Oriented Triangular Lattice**.

The input free form in that there can be any number of words per line as long as the line length, including blanks, is less than 512 characters. Only the order is important.

Note 2: The y-pitch, $P_y = (y_1 - y_2)/(n_y - 1)$, the x-pitch, $P_x = (x_3 - x_2)/(n_x - 1)$, and perforce $y_2 = y_3$. The resultant value of x1 is completely determined by y1, x2, y2, x3, y3, nx, and ny. The actual input x1 is ignored and is included only to preserve the point definition as an (x,y) pair.

Note 3: The COG requirement that all surface numbers be unique places some not readily obvious restrictions on the **Tri-X** input parameters. The internally generated surfaces are numbered sequentially starting at lx, with $2n_x + n_y$ x planes; continue sequentially from ly1 (hence ly1 must be $> lx + 2n_x + n_y$), with $n_x + n_y + 1$ positive slope planes; continue sequentially from ly2 (hence ly2 must be $> ly1 + n_x + n_y + 1$), with $n_x + 2n_y$ negative slope planes; and finally continue sequentially from lz (hence lz must be $> ly2 + n_x + 2n_y$), with $n_z + 1$ z planes.

Example 5: tri-x lattice

```
{ tri-x          $ denoting x-oriented triangular lattice
-2. 10.392304845 $ x1 y1
-8. 0.           $ x2 y2
4. 0.            $ x3 y3
7 7              $ nxny
1 -70. 70.       $ nz z1...
101 201 301 401 $ lx ly1 ly2 lz
fill
1 1 1 1 1 1 1    $ 1st row, tier 2
2 2 2 2 2 2 2    $ 2nd row
1 1 - - - 1 1
2 2 - - - 2 2    $ a '-' rather than a unit# indicates this cell is empty
1 1 - - - 1 1
2 2 2 2 2 2 2
1 1 1 1 1 1 1 }  $ last row, tier 2
```

4.6.3 Y-Oriented Triangular Lattice

If the first symbol in the `{}` is **tri-y** then the data represents an y-oriented triangular lattice. The following is a graphical representation of a 7x7 y-triangular lattice.

```
1st point> O
           O O
y          O O O
|          O O O O
|          O O O O O
|          O O O O O O
----- x  O O O O O O O
2nd point^ O O O O O O O
           O O O O O
           O O O O
           O O O
           O O
           O
           ^3rd point
```

Four sets of plans are generated; equally spaced y-planes, equally spaced positive-slope planes, equally spaced negative-slope planes, and two or more bounding z-planes. These planes define one or more tiers of nested cells of hexagonal cylinders. The input required to define this type of lattice is...

x1 y1	\$ (x,y) of point 1, center of upper left cell
x2 y2	\$ (x,y) of point 2, center of lower left cell
x3 y3	\$ (x,y) of point 3, center of lower right cell
nx ny	\$ number of columns, number of cells per column
nz z(1)... z(nz+1)	\$ number of tiers followed by nz+1 bounding z planes
ly lx1 lx2 lz	\$ initial surface numbers for y, > 0 slope, < 0 slope, and z planes
fill	\$ keyword indicating the start of the fill data
nx*ny*nz entries	\$ unit#'s to fill the lattice, top-to-bottom (in y), left-to-right (in x), \$ starting with 1st point, ending with the 3rd point, this is repeated \$ starting with bottom tier, proceeding to top tier (in z)

Note 1: this fill pattern different than for **X-Oriented Triangular Lattice**.

The input free form in that there can be any number of words per line as long as the line length, including blanks, is less than 512 characters. Only the order is important.

Note 2: The y-pitch, $P_y = (y_1 - y_2)/(n_y - 1)$, the x-pitch, $P_x = (x_3 - x_2)/(n_x - 1)$, and perforce $x_1 = x_2$. The resultant value of y_3 is completely determined by x_1 , y_1 , x_2 , y_2 , x_3 , n_x , and n_y . The actual input y_3 is ignored and is included only to preserve the point definition as an (x,y) pair.

Note 3: The COG requirement that all surface numbers be unique places some not readily obvious restrictions on the Tri-X input parameters. The internally generated surfaces are numbered sequentially starting at l_y , with $n_x + 2n_y$ y planes; continue sequentially from l_{x1} (hence l_{x1} must be $> l_y + n_x + 2n_y$), with $2n_x + n_y$ positive slope planes; continue sequentially from l_{x2} (hence l_{x2} must be $> l_{x1} + 2n_x + n_y$), with $n_x + n_y + 1$ negative slope planes; and finally continue sequentially from l_z (hence l_z must be $> l_{x2} + n_x + n_y + 1$), with $n_z + 1$ z planes.

Example.6: tri-y lattice..

{tri-y	\$ denoting y-oriented triangular lattice
-5.2 10.	\$ x1 y1
-5.2 -2.	\$ x2 y2
5.19230485 -8.	\$ x3 y3
7 7	\$ nxny
1 -50. 50.	\$ nz z1...
101 201 301 401	\$ ly lx1 lx2 lz
fill	
1 1 1 1 1 1 1	\$ 1 st column
2 2 2 2 2 2 2	\$ 2 nd column
1 1 1 1 1 1 1	
2 2 2 - 2 2 2	\$ a '-' rather than a unit# indicates this cell is empty
1 1 1 1 1 1 1	
2 2 2 2 2 2 2	
1 1 1 1 1 1 1 }	\$ last column

4.7 PICTURES of the Geometry

4.7.1 Introduction to PICTURES

COG provides pictures of the user's geometry. These pictures allow the user to visually determine if the geometry is correctly set up and allow COG to check the user's geometry for errors. It is important to note that extensive geometry error-checking is **only** done during the picture-making phase and for sweeps and volume calculations (see “Pictures and Error Checking” section also). During the actual particle transport, geometry error checking is turned off to save time.

Many of the PICTURES that can be prepared by COG are good enough for documentation or for presentations. The PICTURES created are contained in the **...ps** file and can be seen or printed by using the **gs ghostscript** utility or **ps2pdf** converter. With an *X- Windows* program you can view these PICTURES directly on the desktop.

To get COG to draw PICTURES of the geometry, put a PICTURE statement at the end of the GEOMETRY Data Block, i.e., *after* the SECTORs, UNITs and FILLs have been specified. On the first few COG runs of a problem, it is desirable to look at pictures that represent COG's interpretation of your specifications. Geometry errors as simple as typos or as complex as misunderstanding the desired specifications can be detected and corrected before excessive amounts of computer and user time are wasted. PICTURE statements should routinely be included with all GEOMETRY specifications. After the geometry has been thoroughly checked, these statements can be deactivated (by using the \$ comment symbol in front of each line or by moving the whole PICTURE block to follow the END statement). This technique allows each PICTURE statement to be quickly re-activated if the geometry is later changed and needs further checking.

4.7.2 PICTURE — Cross-Section

The simplest PICTURE is the cross-section (CS) cut. The user specifies a plane passing through the geometry, and (in the general case) the corners of a window on this plane. All the surfaces and sectors that intersect this plane, within the window borders, are plotted. The area associated with each SECTOR (or MATERIAL or REGION) is identified by a distinctive pattern or color (see **ASSIGN-MC—Assign Plotting Colors to Materials**).

A cross-section PICTURE is specified by one of the following two methods:

Coordinate Plane Cross-Section PICTURE

(In a plane parallel to a coordinate plane)

PICTURE $\left[\begin{array}{c} \text{CS} \\ \text{SECTIONAL} \end{array} \right]$
 $\left[\begin{array}{ccc} \text{SECTOR} & \text{or SEC} & \text{or S} \\ \text{MATERIAL} & \text{or MAT} & \text{or M} \\ \text{REGION} & \text{or REG} & \text{or R} \end{array} \right] \left\{ \left[\begin{array}{c} \text{LABELSEC} \\ \text{LABELMAT} \\ \text{LABELREG} \end{array} \right] \right\} \{ \text{LABELSURF} \}$
 $\{ \text{COLOR} \} \{ \text{NOPAT} \} \{ \text{RES } resval \} \left[\begin{array}{cccccc} \text{XY} & z_d & x_1 & x_2 & y_1 & y_2 \\ \text{XZ} & y_d & x_1 & x_2 & z_1 & z_2 \\ \text{YZ} & x_d & y_1 & y_2 & z_1 & z_2 \end{array} \right]$
 $\{ \text{TITLE} = " \dots " \}$

where:

CS (or **SECTIONAL**) specifies that a cross-section picture with pattern fill is wanted (see **Patterns for Cross-Section Pictures**, below);

SECTOR (or **MATERIAL** or **REGION**) (or the shorter aliases) specifies what type of area will be drawn in the PICTURE;

XY (or **XZ** or **YZ**) specifies which plane the cross-section picture lies in;

z_d (or y_d or x_d) specifies the distance of the specified plane from the origin;

x_1 x_2 y_1 y_2 (or corresponding values for the other planes) specify the minimum and maximum values for the abscissa and ordinate of the plot.

The Optional $\{ \dots \}$ cross-section picture specifications are these:

LABELSEC (or **LABELMAT** or **LABELREG**) indicates that areas are to be labeled with their *sect-ID*#s (or *mat-ID*#s or *reg-ID*#s).

LABELSURF indicates that surfaces are to be labeled with their *surf-ID*#s.

COLOR enables color fill of material areas. The color used for each material is that assigned by an ASSIGN-MC statement in the MIX Data Block. Without an ASSIGN-MC statement, a default color assignment is chosen from a palette of 20 colors. If COLOR fill is specified there is no pattern fill.

NOPAT specifies that no pattern fill is desired (normally a default pattern is assigned to every different area). This command causes drawing to be significantly faster but user identification of different components will be more difficult, unless the LABELSEC and/or LABELSURF options are used.

RES *resval* specifies the resolution of the picture, in terms of the number of lines desired in the x or y direction. Default for *resval* is 500, which is the highest useful resolution on most monitors. For PostScript files printed to Laserwriters, 1000 gives good results. The smaller the value of *resval*, the quicker the PICTURE will be drawn (at the cost of lower resolution).

TITLE specifies a line of text which will be written at the top of the plot.

4.7.3 General Cross-Section PICTURE

(In a plane with arbitrary orientation)

$$PICTURE \left[\begin{array}{c} \text{CS} \\ \text{SECTIONAL} \end{array} \right] \left[\begin{array}{c} \text{SECTOR or SEC or S} \\ \text{MATERIAL or MAT or M} \\ \text{REGION or REG or R} \end{array} \right] \left\{ \left[\begin{array}{c} \text{LABELSEC} \\ \text{LABELMAT} \\ \text{LABELREG} \end{array} \right] \right\}$$

{LABELSURF}

{COLOR} {NOPAT} {RES *resval*} x_{tl} y_{tl} z_{tl} x_{bl} y_{bl} z_{bl} x_{br} y_{br} z_{br}

{TITLE = "..."}

Here we describe just those entries which differ from those described above for the Coordinate plane Cross-Section PICTURE.

These are:

x_{tl} y_{tl} z_{tl} are the coordinates of a point in the **top left** corner of the picture;

x_{bl} y_{bl} z_{bl} are the coordinates of a point in the **bottom left** corner of the picture;

x_{br} y_{br} z_{br} are the coordinates of a point in the **bottom right** corner of the picture.

The angle described by these three points must be a right angle. The picture scaling will be uniform in both directions and will be computed from these input points.

top left

(x, y, z)

These three data points describe the location of the top left, bottom left, and bottom right (in that order) of the picture plane.

bottom left

(x, y, z)

bottom right

(x, y, z)

Example of a cross-section PICTURE specification:

GEOMETRY

SECTOR 1 IRONBOX -1 2 3 4

. . .

. . .

. . .

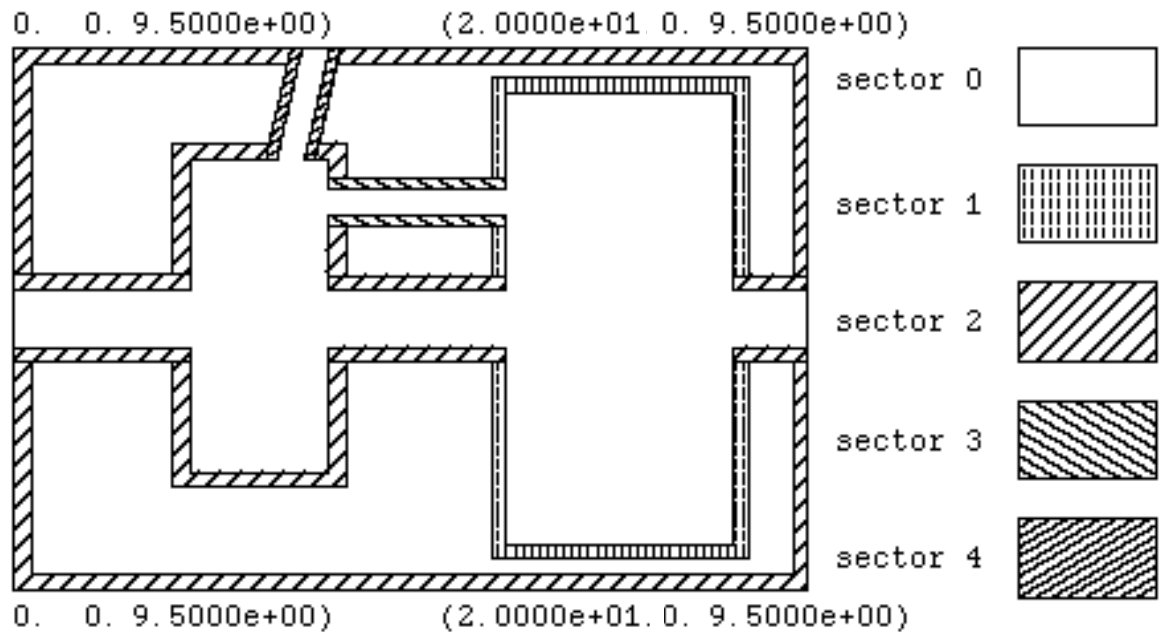
PICTURE CS SECTOR COLOR X Y 10 -6 6 -6 6 \$ XY PLANE @ Z=10

PICTURE CS MATERIAL X Z -50 10 20 -20 40 \$ XZ PLANE @ Y= -50

PICTURE CS SECTOR -6 0 6 -6 0 -6 6 0 -6 \$ Y=0 CROSS SECTION

PICTURE CS SECTOR -6 -6 0 -6 6 0 6 6 0 \$ Z=0 CROSS SECTION

Here is a sample cross-section picture of sectors. The key identifies the sectors and their patterns.



Each picture requires its own statement starting with the word PICTURE. **Note:** The PICTURES are *not* produced in the order listed. All cross-section SECTOR pictures will be drawn before MATERIAL or REGION pictures. The cross-section pattern for a given sector will be the same throughout the entire set of such pictures and will be illustrated on a printed key.

4.7.4 Patterns for Cross-Section Pictures

COG assigns patterns to each plot area based on the user-assigned SECTOR (or MATERIAL or REGION) number. The Pattern number used is the SECTORS/REGIONS/MATERIAL number, modulo 100. **Note:** COG color palette can be specified in ASSIGN-MC Section –Assign Plotting Colors to Materials.

**The Pattern Table
Used in the Geometry PICTURE**

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

4.7.5 PICTURE — Perspective

A perspective picture is the kind you would get if you stood back and took a “photograph” of your geometry. The edges and boundaries between SECTORS (or MATERIALS or REGIONS) are drawn, and the surfaces optionally colored in. The user must specify which SECTORS (or MATERIALS or REGIONS) are visible. The remainder are invisible. Thus, a picture can illustrate just one part of the geometry, or the user can look *inside* outer surfaces to see inner structure.

Perspective pictures are requested by:

PICTURE $\left[\begin{array}{c} \mathbf{P} \\ \mathbf{PERSPECTIVE} \end{array} \right] \left[\begin{array}{ccc} \mathbf{SECTOR} & \text{or} & \mathbf{SEC} & \text{or} & \mathbf{S} \\ \mathbf{MATERIAL} & \text{or} & \mathbf{MAT} & \text{or} & \mathbf{M} \\ \mathbf{REGION} & \text{or} & \mathbf{REG} & \text{or} & \mathbf{R} \end{array} \right]$

$\{COLOR\} \{NOFRAME\} \{RES \text{ resmin resmax}\} \ x_c \ y_c \ z_c$

$r \ d \ \theta \ \phi \ s_1 \ s_2 \ s_3 \dots \quad \{\text{TITLE} = " \dots "\}$

where:

P (or **PERSPECTIVE**) specifies that a perspective picture is wanted;

SECTOR (or **MATERIAL** or **REGION**) (or the shorter aliases) specifies what volumes will be drawn in the PICTURE;

$x_c \ y_c \ z_c$ specify the point P_c at the center of the perspective view;

r specifies the radius of a sphere centered on P_c . Everything within the sphere will be included in the picture.

d, θ, ϕ are spherical coordinates identifying where the viewer will stand to look at the geometry.

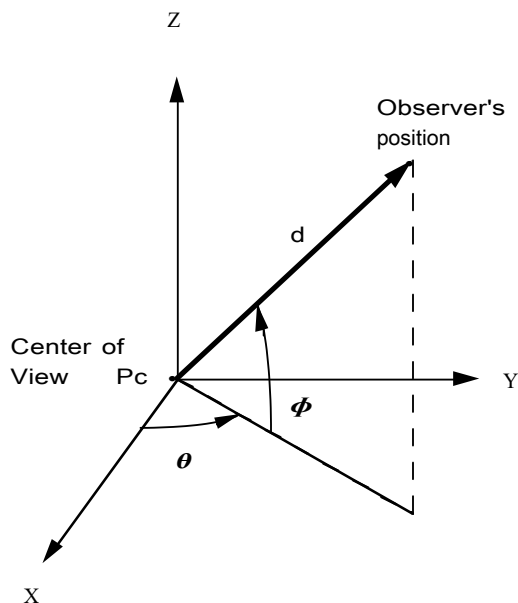
d is the distance of the viewer from the P_c ;

θ is the azimuthal angle measured in the x,y plane, from the x-axis to the Observer's position (degrees);

ϕ is the elevation angle measured upward from this plane toward the +z- axis (degrees).

$s_1 \ s_2 \ s_3 \dots$ is a list of SECTORS (or MATERIAL or REGIONS) that the viewer wishes to see. Any sector that is not in this list is transparent.

Note: The picture perspective drawing rule of thumb resolution should be 4 times away the observer from the object (i.e. $r > 4xd$) for best results.



Perspective View Coordinates

The optional { . . . } PERSPECTIVE PICTURE specifications are these:

COLOR enables color fill of material areas. The color used for each material is that assigned by an ASSIGN-MC statement (else a default color assignment is chosen from a palette of 20 colors).

NOFRAME specifies that the next picture drawn will overlap this one (i.e., no frame advance will occur). This option is useful for making "double exposure" pictures. For example, the first picture could show just the outside of an object (by listing only the exterior sector numbers as visible). The next overlapping picture could show just the object's interior structures in color. The net effect is of a view of a "solid" interior seen through a "transparent" exterior which shows only the surface edges.

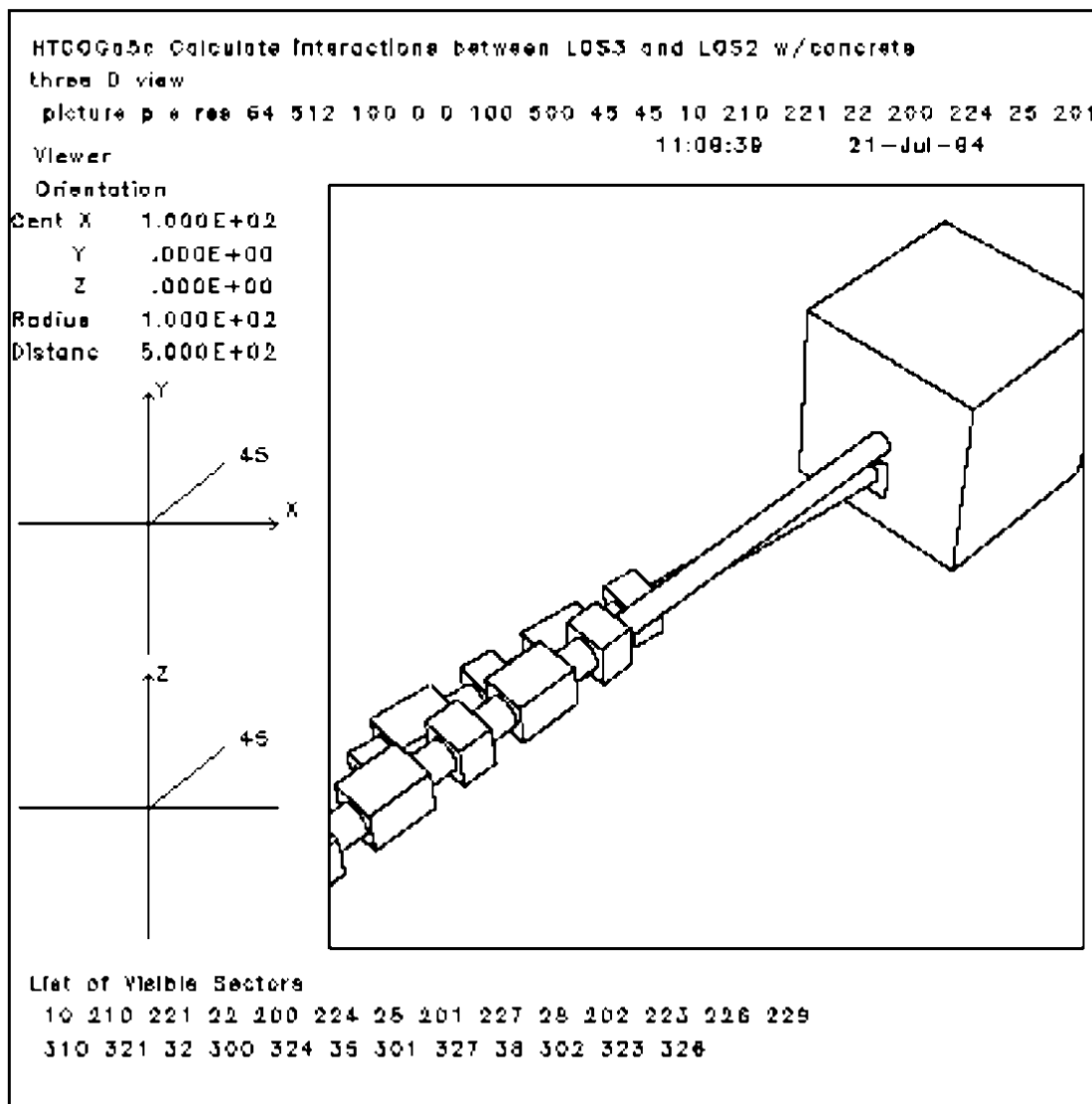
RES *resmin resmax* specifies the minimum and maximum resolution to use in making the picture. *resmin* is the initial (coarse) resolution. The resolution will be successively increased as needed, up to a value of *resmax* in order to resolve fine detail. Default values are *resmin* = 32, *resmax* = 512. Decreasing *resmin* and *resmax* will result in the fast production of blocky pictures, perhaps with some features missing. Increasing *resmin* and *resmax* will produce higher-quality output at higher computational and time costs.

TITLE specifies a line of identification which will be written at the top of the plot.

Example of a PERSPECTIVE PICTURE input line:

```
PICTURE P S COLOR RES 32 512 100 0 0 100 500 45 45 10 210 ...
```

Sample perspective picture seen in ...ps file:



Perspective picture-making proceeds by projecting a uniform mesh set up in the image plane, through a computational pinhole, and into the user's geometry. The initial number of mesh points in the image x- or y- direction is given by *resmin*. If the rays projected from the four corners of a mesh cell all strike the same surface, then the code assumes that the surface is continuous through the cell. If the rays strike different surfaces, the code knows that an edge passes through the cell. It subdivides the cell into four sub-cells, then re-checks each sub-cell for surface continuity. The subdivision process continues as necessary, until the maximum specified resolution *resmax* is reached. Then points are drawn in between cells of differing surface number, to represent geometry edges.

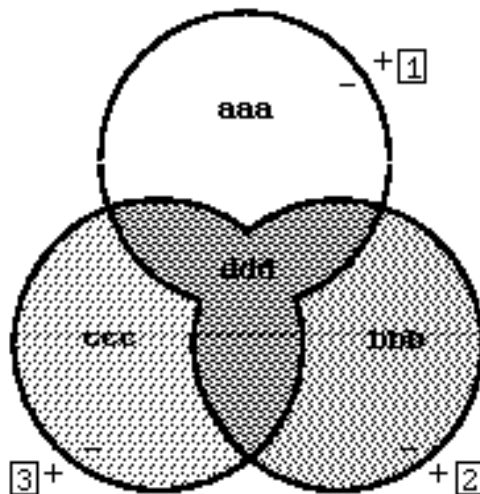
This adaptive resolution process generally works well to resolve fine detail at minimum computational cost. However, it may miss some edges, particularly if the initial resolution *resmin* is set too coarse. You can try the following fixes:

- Increase *resmin* and *resmax* values (limit 512). If detail is still missing, you will have to "blow-up" the picture by decreasing *r*, so that a smaller region is viewed at high resolution.
- Shift the picture just a little by moving the center or one of the viewing angles and then letting the code try again.

Like cross-sectional pictures, if an error in the geometry input is determined, PICTURE drawing will stop at that point and an error printout will be given.

4.7.6 PICTURES and Error Checking

COG can detect many different kinds of errors while processing the geometry during PICTURE generation. These errors are printed out in a form that should be self-explanatory. While drawing PICTUREs or calculating VOLUMEs, the code detects accidental overlap of SECTORs. If this occurs, an error message is printed which gives all the information that the code knows about this problem, including the boundary just crossed (if any) and the relationships to all surfaces in current use. With this information, the user should be able to identify and fix the overlap problem. To illustrate this error procedure, the following case was run. Note that the definition of sector 3 missed a reference to surface number 2.



BASIC GEOMETRICAL DESCRIPTION

```
SECTOR 1  AAA  -1  +2  +3
SECTOR 2  BBB  +1  -2  +3
SECTOR 3  CCC  +1  -3  $  OMIT  +2
SECTOR 4  DDD  -1  -2
OR
                -2  -3  +1
OR
                -1  -3  +2
```

This error generated the following error message in the output file:

```
ERROR --point found which is in at least two sectors
      given point in level 0 coordinates
      x = 1.4917534e +00
      y = -6.1780699e -01
      direction cosines
      u = 1.000000000e +01
      v = 0.
      w = 0.
      relation to each surface at this point
      equation  evaluation
      1          +
      2          - (just crossed this surface)
      3          -
numbers and names of sectors which contain this point
      3      CCC
      4      DDD
```

4.8 VOLUME Calculations

COG can compute the VOLUME of user's sectors. In this option, you specify a box-shaped volume within your geometry. COG performs a Monte Carlo calculation of the volume and the mass of each MATERIAL, REGION, or SECTOR within that box. This can be an expensive calculation, but it is also the best method for finding errors associated with overlapping sectors. It is the only way to detect geometry errors which result in volume or mass discrepancies from the original physical model. You may request any number of VOLUME calculations. Each VOLUME specification has this form:

$$VOLUME \begin{bmatrix} \text{SECTOR} & \text{or} & \text{SEC} & \text{or} & \text{S} \\ \text{MATERIAL} & \text{or} & \text{MAT} & \text{or} & \text{M} \\ \text{REGION} & \text{or} & \text{REG} & \text{or} & \text{R} \end{bmatrix} \{RES \ nres\} x_0 \ y_0 \ z_0 \ x_1 \ y_1 \ z_1 \\ x_2 \ y_2 \ z_2 \ length-x' \ length-y' \ length-z' \{TITLE = "..."\}$$

where:

SECTOR (or **MATERIAL** or **REGION**) (or the shorter aliases) specifies what volume will be determined;

x_0, y_0, z_0 is a point in the reference corner of the box (in problem coordinates);

x_1, y_1, z_1 is any point along one edge, or its extension, of the box. This, along with the reference corner, defines the +x'-axis;

x_2, y_2, z_2 is any point along another edge, or its extension, of the box. This, along with the reference corner, defines the y'-axis. The z'-axis is constructed to form a right-handed box coordinate system;

length-x', *length-y'*, and *length-z'* are the lengths of the defined box along each of the three box axes;

RES nres specifies an optional higher precision for the volume calculation (*nres* a positive integer). When *nres* is specified, the standard error of the calculation is decreased by a factor of $1/\sqrt{nres}$.

The following is an example of the result of a VOLUME calculation.

Example of VOLUME option:

```
VOLUME  S      -3. -3. -3.  3. -3. -3.  -3.  3. -3.  6. 6. 6.
TITLE "RES DEFAULT"
VOLUME  S RES  2 - 3. -3. -3.  3. -3. -3.  -3.  3. -3.
                        6. 6. 6.

TITLE "RES 2"
VOLUME  S RES  4 -3. -3. -3.  3. -3. -3.  -3.  3. -3.
                        6. 6. 6.

TITLE "RES 4"
```

Example of a VOLUME output.

Example of a VOLUME output.

```
VOLUME CALCULATION
VOLUME TEST
POINT ON CORNER (ORIGIN) (6.00000E+00,-6.00000E+00,6.00000E+00)
POINT ON X'-AXIS          (7.00000E+01,-6.00000E+01,-6.00000E+00)
POINT ON Y'-AXIS          (6.00000E+01,-5.00000E+01,-6.00000E+01)
      Z'-AXIS FORMS A RIGHT-HANDED SYSTEM WITH ABOVE
      LENGTH OF X' SIDE = 6.00000E+00
      LENGTH OF Y' SIDE = 1.20000E+01
      LENGTH OF Z' SIDE = 1.20000E+01
```

SECTOR NUMBER	VOLUME	FSD	MASS (KG)	FSD
1	2.543E+02	0.007	2.543E+02	0.007

4.9 The SWEEP Statement

The SWEEP statement sweeps a line through the user's geometry between two specified points. All sectors and their boundary surfaces which intersect the SWEEP line will be listed in the output file, along with the distance of each intercept from the "start" point. This statement is very helpful in analyzing the exact COG placement of surfaces in a complex region of the geometry.

The format is:

SWEEP x_0 y_0 z_0 x_1 y_1 z_1

where:

x_0 y_0 z_0 represents the starting point of the sweep;

x_1 y_1 z_1 represents the ending point of sweep.

Example of a SWEEP statement:

```
SWEEP 5. -10. 3.9813 15. -10. 3.9813
```

Example of the output of the above statement:

```
SWEEP 5. -10. 3.9813 15. -10. 3.9813
```

X	Y	Z	DIST	CROSS SURF	LEAVE SECTOR	NAME	MAT	LEV
5.0000E+00	-1.0000E+01	3.9813E+00						
6.2221E+00	-1.0000E+01	3.9813E+00	1.2221E+00	40	0	FILL	0	0
7.1783E+00	-1.0000E+01	3.9813E+00	2.1783E+00	30	104	SPH4	1	0
8.2789E+00	-1.0000E+01	3.9813E+00	3.2789E+00	20	103	SPH3	6	0
9.4000E+00	-1.0000E+01	3.9813E+00	4.4000E+00	91	0	FILL	0	1
9.6434E+00	-1.0000E+01	3.9813E+00	4.6434E+00	92	911	SPHU1	2	1
9.8009E+00	-1.0000E+01	3.9813E+00	4.8009E+00	93	0	FILL	0	2
1.0199E+01	-1.0000E+01	3.9813E+00	5.1991E+00	93	201	SPH5	3	2
1.0356E+01	-1.0000E+01	3.9813E+00	5.3565E+00	92	0	FILL	0	2
1.0600E+01	-1.0000E+01	3.9813E+00	5.6000E+00	91	911	SPHU1	2	1
1.1721E+01	-1.0000E+01	3.9813E+00	6.7210E+00	20	0	FILL	0	1
1.2000E+01	-1.0000E+01	3.9813E+00	7.0000E+00	52	103	SPH3	6	0
1.2037E+01	-1.0000E+01	3.9813E+00	7.0374E+00	60	0	FILL	0	0
1.2822E+01	-1.0000E+01	3.9813E+00	7.8217E+00	30	109	CONE1	5	0
1.3778E+01	-1.0000E+01	3.9813E+00	8.7779E+00	40	109	CONE1	5	0
1.4000E+01	-1.0000E+01	3.9813E+00	9.0000E+00	60	109	CONE1	5	0
1.5000E+01	-1.0000E+01	3.9813E+00	1.0000E+01	0	0	FILL	0	0

Where

X,Y,Z are the coordinates of the point, in problem coordinates;

DIST is the distance of the point from the starting point;

SURF is the *surf-ID#* of each surface intercept with the SWEEP line;

SECTOR is the *sect-ID#* of the sector just exited;

NAME is the sector name;

MAT is the *mat-ID#* of the material filling the sector;

LEVEL is the UNIT level of the intercept; level-0 is the topmost level of the COG problem; level-1 is inside a UNIT used at level-0; level-2 is inside a UNIT nested inside a level-1 UNIT, etc.

5 The MIX Data Block

5.1 Specifying Problem Materials

SECTORs are the geometrical volumes in the user's problem. To complete the simulation of real parts, these SECTORs must be filled with materials. The problem's materials are specified in the MIX Data Block, which we now discuss. Once problem materials have been specified, we then assign them to SECTORs with ASSIGN statements (see **The ASSIGN Data Blocks**).

COG materials are the physical materials (such as lead or water or steel) in the problem. Each physical material is composed of one or more elements, isotopes, or mixtures, each with an associated density. COG extracts the reaction cross-sections for each material from a disk-file library. These cross-sections are the fundamental physics data used to determine the outcome of particle random walks.

Format of a **MIX** Data Block, showing the alternative means of specifying component amounts by density, atomic fraction, weight percent/fraction, and bunches (atoms/barn-cm). Weight percent and atom fraction are relative values (unnormalized).

MIX

MAT = mat-ID#₁

component-1 density-1

{*component-2 density-2* }

. . .

MAT = mat-ID#₂

ATOM-FRACTION *material-density*

component-1 atom-fraction-1

{*component-2 atom-fraction-2* }

. . .

MAT = mat-ID#₃

WEIGHT-PERCENT *material-density*

component-1 weight-percent-1

{*component-2 weight-percent-2* }

. . .

MAT = mat-ID#₄

BUNCHES

```

component-1 bunches-1
{component-2 bunches-2 }
. . .
{NLIB = libname}

{TEMP = <temperature in Kelvin>}

```

For each material, the user specifies the composition in terms of density, atom fraction, weight percent or bunches. For example, the following are all equivalent...

- 1) Density of components (gm/cc)

```
MAT 1 h 0.11527 o16 0.91473
```

Material 1 consists of 0.11527 gm of h1 per cc of material, and 0.91473 gm of o16 per cc of material.

- 2) Atom fraction (can use # or A-F instead of ATOM-FRACTION)

```
MAT 2 ATOM-FRACTION 1.03 h1 2. o16 1.
```

Material 2 has a total density of 1.03 gm/cc and consists of H1 and O16 in the ratio of 2. atoms of H1 to 1. atom of O16.

- 3) Weight percent (can use W-P instead of WEIGHT-PERCENT)

```
MAT 3 WEIGHT-PERCENT 1.03 h1 11.1915 o16 88.8085
```

Material 3 has a total density of 1.03 gm/cc and is 11.1915% by weight h1 and 88.8085% by weight o16 (code will normalize weight percents to 100).

- 4) Bunches or atoms/barn-cm (can use BUN instead of BUNCHES)

```
MAT 4 BUNCHES h1 0.06888 o16 0.03444
```

Material 4 consists of 0.06888 atoms/barn-cm of h1 and 0.03444 atoms/barn-cm of o16 (these are the units used internally by COG)

MIX identifies the following lines as the MIX Data Block;

MAT designates the start of a MAT definition. Each MAT definition describes one problem material. Note that more than one MAT definition can describe the same material, if desired. More materials may be specified than used in the problem.

mat-ID# is the user-assigned integer identifying the material definition which follows. *mat-ID#s* can occur in the MIX Data Block in any desired order.

component-1 is the cross-section library ASCII name for the first compound, elements and/or isotope in the material definition. See **COGLEX Listing** for a list of the available component names. Many synonyms are available for common components. Isotopes can alternatively be specified by their ZAID number, which is an integer formed as:

ZAID = 1000*Z + A;

where Z is the isotope's atomic number and A is its mass number. For example, 82208 is the ZAID for Pb208.

density-1 is the density of *component-1*, as used in this material, in units of gm/cc. **Note that density values are always in units of gm/cc, regardless of units specified by the user in the BASIC Block.**

Other *component* and *density* specifications follow, as needed, to complete the definition of the material.

ATOM-FRACTION (or # or **A-F**) designates that the following material definition is in terms of atom fractions.

material-density is the material density in gm/cc.

atom-fraction-1 is the relative atom fraction of *component-1*, as used in this material.

Other *component* and *atom-fraction* specifications follow, as needed, to complete the definition of the material.

WEIGHT-PERCENT (or **W-P**) designates that the following material definition is in terms of weight percents.

weight-percent-1 is the relative weight percent of *component-1*, as used in this material.

Other *component* and *weight-percent* specifications follow, as needed, to complete the definition of the material.

BUNCHES (or **BUN**) designates that the following material definition is in terms of bunches or atoms/barn-cm.

bunches-1 is the bunches of *component-1*, as used in this material.

Other *component* and *bunches* specifications follow, as needed, to complete the definition of the material.

NLIB *libname* tells COG to use a specified neutron cross-section library. **Note: There is NO default neutron library; the desired neutron library *must* be specified for a neutron transport problem.**

TEMP = <temperature in Kelvin> is optional parameter to tell COG to adjust Free-Gas thermal temperature. **Note: The default temperature is 293° K.** To find file *libname*, COG will look first in the user's local directory (which contains the user's input file), then in the standard COG library directory.

To use **ENDL90**:

NLIB = ENDL90

To use **ENDFB8R0**:

NLIB = ENDL90 TEMP=348.

Note: If an ENDFB library is specified, the user cannot specify the Energy-Deposition response function for any of the DETECTORS, because the necessary energy deposition data are not present in the ENDF library.

Note: The ENDFB library temperature specification is dependent on the individual library's processed temperature, i.e. ENDFB8R0.500K, ENDFB8R0.400K, and ENDFB8R0.300K.

See the section on **Cross-Section Libraries** for more information on mixtures and cross-section models.

Example: Assume that material number 7 in your COG problem is silicon dioxide (SiO_2) with a density of 2.64 gm/cc. Its elemental composition is silicon (At. Wt. = 28.09) and oxygen (At. Wt. = 16.00). The silicon density would be:

$$(2.64) [(28.09) / (28.09 + 32.00)] = 1.23$$

and that of oxygen would be:

$$(2.64) [(32.00) / (28.09 + 32.00)] = 1.41$$

The SiO_2 material definition would be:

```
MAT=7  SI  1.23  O  1.41
```

Using the atomic number fraction specification instead, the SiO_2 material definition would be:

```
MAT=7  # 2.64 SI  1.0  O  2.0
```

Example of a MIX Data Block: The materials include elements (Cu, Al), compounds (water) and mixtures (air, stainless steel)

```
MIX
MAT=1  CU  8.96                $ COPPER @ 8.96 G/CM3
MAT=12 SS304 7.78              $ STAINLESS STEEL 304 AT 7.78 G/CM3
MAT=3  H  0.1101276  O  0.8810218  BORON  0.0000138  $ BORATED WATER
MAT=4  WATER 1E-5              $ STEAM AT 1E-5 G/CM3
MAT=50  AL  2.7                $ ALUMINUM
MAT=6  AIR  0.00129            $ MIXTURE OF N, O, Ar @.00129 G/CM3
MAT=7  BORON 0.0402112  AL  2.6633123  FE  0.0042681  SI  0.0016007
```

5.2 Special Material Numbers 0 and -1 (VOID and Infinite Absorber)

Two *mat-ID#*s are reserved for special pre-defined materials. *mat-ID#* = 0 is a void (a perfect vacuum); all sectors assigned a *mat-ID#* of 0 will contain void. Particles can never scatter in void. *mat-ID#* = -1 denotes a (fictitious) perfect absorber. Particles entering a sector assigned a *mat-ID#* of -1 are immediately absorbed.

Zero Cross Sections for Missing Neutron Data

If a specified material cannot be found in the neutron library, COG ordinarily quits with an error message. To enable such a problem to run in the absence of neutron data, place the word

USEZEROXS

as the first word in the MIX Data Block. This will cause COG to use zero as the cross section for any material not found in the neutron library.

5.3 Thermal Treatment

The definitions in the COG dictionary COGLEX have been expanded to provide an approximate thermal treatment for many materials. For dictionary materials that have 'water-like' Hydrogen bonds, we added a corresponding definition where the Hydrogen was replaced by Hydrogen bound in water, e.g. in addition to a definition for Blood we now have a definition for (Blood), where the ()'s indicate a thermal treatment. We made similar additions for those materials that had 'CH₂-like' bonds, e.g. Acetone and (Acetone).

5.4 Data Libraries

The following data libraries are available in this release:

Neutron Activation Libraries

ACTL92	LLNL's ACTL 1992
EAF2010	European Activation File 2010

Photon Libraries

COGGXS	defaults to EPDL97
EPDL89	LLNL's EPDL-1989
EPDL97	LLNL's EPDL-1997
EPICS2017	IAEA EPICS library 2017
EPICS2023	IAEA EPICS library 2023

Nuclear Resonance Fluorescence Library

COGNRF

LLNL's Dr. James Hall data

Photonuclear Libraries

COGPNUC

defaults to IAEAPNUC

IAEAPNUC

IAEA

PN.ENDFB7R1

ENDF/B-VII.1

PN.MCNP.70u

MCNP's .70u

Radiation Simulation Library

COGRS

developed by LLNL's Dr. E. M. Lent

Delayed Fission Gamma Libraries

developed by LLNL's Dr. E. M. Lent

DFG.ENDFB7R1

ENDF/B-VII.1

DFG.JEFF3.1.1

JEFF3.1.1

DFG.JENDL4

JENDL4

Neutron Libraries

ENDFB6R7

ENDF/B-VI.7

ENDFB6R8

ENDF/B-VI.8

ENDFB7R0

ENDF/B-VII.0

ENDFB7R0.BNL

BNL's ENDF/B-VII.0

ENDFB7R1

ENDF/B-VII.1

ENDFB7R1.BNL

BNL's ENDF/B-VII.1

ENDFB8R0

ENDF/B-VIII.0

ENDFB8R0.ACE

ENDF/B-VIII.0 ACE

ENDL2008

LLNL's ENDL-2008

ENDL90

LLNL's ENDL-1990

ENDL99

LLNL's ENDL-1999

JEFF3.1	JEFF3.1
JEFF3.1.1	JEFF3.1.1
JEFF3.1.2	JEFF3.1.2
JEFF3.2	JEFF3.2
JEFF3.3	JEFF3.3
JENDL3.3	JENDL3.3
JENDL4	JENDL4
JENDL5.ACE	JENDL5 ACE
MCNP.50c	MCNP's .50c
MCNP.51c	MCNP's .51c
MCNP.55c	MCNP's .55c
MCNP.66c	MCNP's .66c
MCNP.70c	MCNP's .70c
RED2002	Hybrid ENDFB/ENDL library by Dr. D. Cullen (2002)

Probability Table Libraries

PT.ENDFB7R0.BNL	BNL's ENDF/B-VII.0
PT.ENDFB7R1.BNL	BNL's ENDF/B-VII.1
PT.ENDFB8B3.ACE	ACE's ENDF/B-VIII beta 3
PT.ENDFB8R0.ACE	ACE's ENDF/B-VIII
PT.JEFF3.1	JEFF3.1
PT.JEFF3.1.1	JEFF3.1.1
PT.JEFF3.1.2	JEFF3.1.2
PT.JEFF3.2	JEFF3.2
PT.JEFF3.3	JEFF3.3
PT.JENDL5.ACE	JENDL5 ACE

PT.MCNP.66c	MCNP's .66c
PT.MCNP.70c	MCNP's .70c
PT.MCNP.71nc	MCNP's .70nc
PT.MCNP.80nc	MCNP's .80nc

Thermal Libraries

T.HZIce	Holmes & Zerkle's ACE
T.IAEA2007	IAEA 2007
T.NCSU	NCSU ENDF
T.NCSU.ACE	NCSU ACE
T.H.H2O.NCSU.25C	NCSU 25C
T.H.H2O.NCSU.25C.ACE	NCSU 25C ACE
T.CAB2015.ACE	CAB2015 ACE
T.ENDFB3R0	ENDF/B-III.0
T.ENDFB6R0	ENDF/B-VI.0
T.ENDFB6R2	ENDF/B-VI.2
T.ENDFB7R0	ENDF/B-VII.0
T.ENDFB7R0.BNL	BNL's ENDF/B-VII.0
T.ENDFB7R0.LANL	LANL's ENDF/B-VII.0
T.ENDFB7R1	ENDF/B-VII.1
T.ENDFB7R1.BNL	BNL's ENDF/B-VII.1
T.ENDFB7R1.LANL	LANL's ENDF/B-VII.1
T.ENDFB8B4	ENDF/B-VIII beta 4
T.ENDFB8R0	ENDF/B-VIII.0
T.ENDFB8R0.ACE	ENDF/B-VIII.0 ACE
T.FUDGE21.07.21	LLNL's FUDGE2021
T.FUDGE21.09.02	LLNL's FUDGE2021
T.FUDGE2024	LLNL FUDGE2024

T.JEF2.2	JEF2.2
T.JEFF3.0	JEFF3.0
T.JEFF3.1	JEFF3.1
T.JEFF3.1.1	JEFF3.1.1
T.JEFF3.1.2	JEFF3.1.2
T.JEFF3.2	JEFF3.2
T.JEFF3.3.ACE	JEFF3.3 ACE
T.JEFF3.3.ENDF	JEFF3.3 ENDF
T.JENDL4	JENDL4
T.JENDL5	JENDL5
T.JENDL5RmTmp	JENDL5 293° K
T.MCNP.71	MCNP's ENDF71SaB
T.MCNP.80	MCNP's ENDF80SaB2

Dosimetry Libraries

IRDF2002	IRDF-2002
IRDFF1.02	IRDFF Release 1.02
IRDFF1.05	IRDFF Release 1.05
IRDFF-II	IRDFF Release 2

Alpha Libraries

A.DEDX
A.JENDL2005
A.TENDL2013
A.RSCOG

Deuteron Library

D.ENDFB7R1

Proton Library
P.DEDX

6 Cross Section Libraries

COG uses a set of cross-section data files which provides the physics information needed to simulate particle collisions with materials. See Appendix for a complete listing of available cross-section data files.

Because most of the COG physics resides in these cross-sections, the user should understand the limitations and assumptions in these libraries. Each material cross section may have a different uncertainty and should be evaluated separately.

Results obtained from transport calculations using these data will only be as good as the cross-sections allow. *The knowledgeable user will have acquired a feeling for the cross-section information and its accuracy and the consequent accuracy of his calculated results.* To assist the user in this process, COG can plot the total cross section of each material (see **The I/O Data Block**). Other routines external to COG are available which will list all or part of any cross-section library.

Neutron data in the COG cross-section files are for materials at room temperature (i.e., the cross sections have been Doppler broadened). It is possible to generate files that contain different evaluations of cross sections or cross sections at other than room temperature. Normally, the gamma cross section library has a much smaller uncertainty in it and is not changed from the default.

At the beginning of the MIX Data Block, you can insert one or more statements of the following form:

lib-type = *file-name*

where:

lib-type is a cross-section library file type and;

file-name is the name of the file you wish to use. Your selected cross section library file must reside on disk at the time the problem is run.

6.1 Neutron Libraries (NLIB)

NLIB= *filename*

where:

filename is the name of the desired COG neutron library (e.g., ENDFB8R0), required to do neutron tracking. There is no default neutron library.

6.1.1 Secondary Neutron Libraries (NLIB2 and NLIB3)

COG has the capability of reading from a second and/or third neutron data file. To use this option insert statement(s) of the following form:

NLIB2 = *filename2 isotope2.1 isotope2.2 ...*

NLIB3 = *filename3 isotope3.1 isotope3.2 ...*

where:

filename2, filename3 are names of COG neutron libraries (e.g., **ENDFB8R0**);
isotope2.1 isotope2.2 ... are the names (or ZAIDs) of the desired isotopes to be read from the specified library.

Examples of NLIB2 option:

```
MIX
NLIB ENDFB6R7
NLIB2 JEFF3.3 C
MAT 1 H 0.112 O 8.888
MAT 2 AL 2.7
MAT 3 C 2.2
```

The above MIX block will read all isotopes from ENDFB6R7 except C, which will be read from library JEFF3.3.

Examples of NLIB2/NLIB3 option:

```
MIX
NLIB ENDFB6R7
NLIB2 JEFF3.3 C
NLIB3 ENDL90 AL27
MAT 1 U235 10.
MAT 2 AL 1. C 1.
```

The above MIX block will read U235 from ENDFB6R7, C from JEFF3.3, and Al27 from ENDL90.

Note that the isotope entries allowed in **NLIB2** and **NLIB3** are more restrictive than in the **MAT** definitions. That is, if we had entered just Al (as in the **MAT** definition) rather than Al27 on the **NLIB3** line, the dictionary would have translated Al to 13000 and the code would return an error message because 13000 is not available in **ENDL90** (Al is available in **ENDL90** as 13027).

6.2 $S(\alpha,\beta)$ Libraries (SABLIB and SABLIB2)

The default COG model for thermal neutron elastic scattering is the free gas model. For a neutron reaction, COG samples the neutron relative speed distribution. For some compounds, COG can alternatively compute elastic scattering using the $S(\alpha,\beta)$ model derived from the ENDF/B data set (Reference 5, **Introduction** Section). This model attempts to account for molecular binding effects in elastic scattering off the target element. The $S(\alpha,\beta)$ model is invoked by using special material names in the MIX Data Block. Some thermal neutron problems, such as computing k_{eff} for highly-moderated critical assemblies, require the $S(\alpha,\beta)$ thermal scattering treatment to yield a correct answer.

Each data set in each library was computed for a temperature of 296°K. In each file, the data takes the form of cross sections followed by associated properties. The cross sections are the inelastic (from the double integral of the $S(\alpha,\beta)$ data), and, if applicable, either the coherent elastic (from the Debye-Waller coefficient) or the incoherent elastic (from the Bragg edge-strength terms). The latter two cross sections have no properties associated with them, as we assume zero energy loss and isotropic scattering to characterize the scattered neutron. The property associated with the inelastic cross section is the raw $S(\alpha,\beta)$ data along with integrals of same over b (which represent energy distributions for the scattered neutron).

The ‘bound’ cross section(s) described above replace the ‘unbound’ elastic cross section below some energy limit, E_{max} , currently set to 25 eV. The remaining cross sections for the material are obtained from the specified neutron library.

SABLIB = *filename* ;

where:

filename is one of the $S(\alpha,\beta)$ libraries listed in Appendix, required to use thermal treatment. There is no default $S(\alpha,\beta)$ library.

To specify a secondary $S(\alpha,\beta)$ library:

SABLIB2=*filename2 material.1 material.2 ...*;

where

filename2 is one of the $S(\alpha,\beta)$ libraries listed,
material.1 material.2 ... are the names of the desired materials to be read from the specified library.

6.2.1 Using $S(\alpha,\beta)$ data in a COG run

To use the $S(\alpha,\beta)$ data, the user must specify one of the special $S(\alpha,\beta)$ names in a COG Material definition.

For instance, this Material definition in the **MIX** Data Block

MAT 1 water 1.

causes COG to use the (default) free gas model for this material (water); while

MAT 2 (h.h2o) 0.111 O16 0.889

invokes the $S(\alpha,\beta)$ model for thermal neutron scattering off H in this material (also water).

One could also use the $S(\alpha,\beta)$ model for H in water, but apply it to another H-containing compound. For example, one could specify:

MAT 3 (h.h2o) 0.027 cl 0.973

This causes COG to use the $S(\alpha,\beta)$ model for H in water, for thermal neutron scattering on H in HCl.

A complete list of thermal materials by $S(\alpha,\beta)$ library is given in Appendix.

6.3 Deuteron Libraries (DLIB)

DLIB= *filename*

where:

filename is the name of the desired COG deuteron library, required to do deuteron tracking. There is no default deuteron library.

6.4 Alpha Libraries (ALIB)

ALIB= *filename*

where:

filename is the name of the desired COG alpha library, required to do alpha tracking. There is no default alpha library.

6.5 Photon Libraries (PLIB)

PLIB= *filename*

where:

filename is the name of the desired COG photon library. If this option is not used the photon library defaults to EPDL97.

6.6 Photonuclear Libraries (PNLIB)

PNLIB= *filename*

where:

filename is the name of the desired COG photonuclear library, required to process photonuclear reactions (i.e., keyword ***photonuclear*** in BASIC block). There is no default photonuclear library.

6.7 Probability Table Libraries (PTLIB)

PTLIB= *filename*

where:

filename is the name of the desired COG probability table library, required to do unresolved resonance region self-shielding (i.e., keyword ***urrpt*** in BASIC block). There is no default probability table library.

6.8 Delayed Fission Gamma Libraries (DGLIB)

DGLIB= *filename*

where:

filename is the name of the desired COG delayed fission gamma library, required to track delayed fission gammas (i.e., keyword ***delayedgamma ...*** in BASIC block). There is no default delayed fission gamma library.

6.9 Dosimetry Libraries (DOSLIB)

DOSLIB= *filename*

where:

filename is the name of the desired COG dosimetry library, required to do dosimetry calculation (i.e., ***irdf-r-r*** detector response function in DETECTOR block). There is no default dosimetry library.

6.10 Compounds and Mixtures

COGLEX Listing lists the available COG cross-section data for isotopes, elements, compounds, and mixtures. Often there is more than one way to specify a compound or mixture. Recalling our silicon dioxide example in the MIX Data Block, we could specify SiO₂ as:

```
MAT=7 SI 1.23 O 1.41
```

or as:

```
MAT=7 SIO2 2.64
```

because SIO2 is a recognized name from the list. We can also find these alternative names for silicon:

```
SI
14000
SILICON And,
for oxygen:
o
o16
8000
8016
OXYGEN
```

The 8016 or o16 signifies that natural oxygen is entirely composed of the isotope with a weight of sixteen. From the separate photon and neutron library listings, you can also see that both neutron and photon data are available for these two elements.

For some elements, COG has isotopic cross-sections available to construct "natural" compositions. For instance, specifying:

```
MAT=10 nitrogen .2
```

will result in the use of 'natural' nitrogen, a mixture of N¹⁴ and N¹⁵. But you could instead ask for:

```
MAT=10 n14 .2
```

or

```
MAT=10 n15 .2
```

if isotope-specific cross sections were what you wanted. Note that isotope cross sections do *not* apply to gamma cross sections.

The COG dictionary of materials was revised to use updated values of "natural" isotopic abundances for the elements. The relative abundances were taken from the "Handbook of Chemistry and Physics, 46th Edition".

Typos in the Handbook were corrected and the abundances renormalized where necessary. Small abundances (less than 0.1%) were ignored. User neutron transport jobs which specify materials composed of "natural" elements will now generally

invoke a different set of isotopes than in previous versions of COG. For example, in earlier versions C was interpreted to mean C12. Now C will invoke a mixture of the isotopes C12 and C13 in the ratio .9889/.0111 . If you wish to rerun old COG neutron jobs containing C, and you wish to get the same results as before, then you will have to change C => C12. Similar changes will be needed for most other elements as well. All standard COG mixtures such as air and stainless steel ss304 now use the revised natural compositions.

But note: not all isotopes invoked for a "natural" element can necessarily be found in a given neutron library (e.g., the default ENDL90 library). COG follows these rules to generate the "natural" element cross sections:

- If the library has a cross-section already formulated for the "natural" element, then it is used;
- Otherwise, the mix of isotopes for that element from the COG dictionary is used, if these cross sections are available in the library;

The user should check the MIXTURE COMPOSITIONS section of the COG output file to determine which isotopes were used in each job material.

In addition to the elements, COG has cross-sections for compounds such as water, silicon dioxide and benzene, and for common mixtures such as air, steel, fiberglass, and concrete.

The user must be careful that these COG mixtures accurately represent the materials of the physical problem being modeled. The compositions given in **COGLEX Listing** may not be correct for all problems. For example, concrete is different in different geographical regions, because the composition of the sand and gravel used to make it varies from region to region. It even varies within one region, because slight variations occur in the relative amounts of sand, gravel, lime, and water used. It even varies within the same batch, since the temperature during hardening varies due to heat transfer (thus varying the amount of retained water). Therefore, no one *concrete* composition correctly represents the composition of all concretes. Those compositions given in the library are representative concretes and should be used with care. The same statement is true for other compositions built into the dictionary. In the same way, trace amounts of elements are not included in the standard definitions. For most cases, this does not affect the calculated results. In other cases, the code results may arise *solely* from reactions occurring in the trace elements, and large errors could result by using the *standard* definition in the library. The classic example of this is the thermal neutron activation of structural aluminum. The aluminum itself does not activate, so all the activation comes from the less-than-1% impurities. Again, the user must exercise a certain amount of expertise and not just blindly use the built-in mixture definitions. **Note:** The densities given in **COGLEX Listing** are *not* used by COG and may not be correct in all cases. They exist in the COG dictionary, because this same dictionary is used with other codes that do require these data.

A typical MIX Data Block for a problem with mixtures might look like:

```

MIX                                     $ EXAMPLE #1
MAT=1  WATER  1.0
MAT=2  U238O2  18.5
MAT=5  N  0.00125

MIX                                     $ EXAMPLE #2
NLIB = ENDL90
MAT 1 PU238 3.31E-6 PU239 0.011653 PU240 0.00069283
PU241 4.3E-5 PU242 8.8256E-6 U235 1.9747E-4 U236 5.516E-6
U238 0.02970277
H 0.106093 O 0.927068 N 0.025298376
MAT 2 SS304 7.78
MAT 3 (H.H2O) 0.112 O 0.888

MIX                                     $ EXAMPLE #3
MAT 1 FE 7.87
MAT 2 CONCRETE 2.25
MAT 3 AIR 1.29E-3
MAT 4  H      0.066
      B      0.143
      C      0.158
      O      0.633

ASSIGN-ML  1      1 2 3 4 5 6 7 8 9 10 11 /
           2      21 /
           3      99 /
           4      25

```

7 The I/O Data Block

The I/O (Input/Output) Data Block allows the user to request the output of additional problem information, beyond what is normally provided by COG.

The I/O Data Block has the form

**I/O (PLOTXS) (PLOTMXS) (PRINTXS) (PRINTMXS)
(PRINTPDEDX) (PRINTPMDEDX)**

Where the Option keywords following I/O are:

PLOTXS — To plot isotope/element Cross Sections of the materials specified in the MIX Data Block;

PLOTMXS — To plot the total material Cross Sections of the materials specified in the MIX Data Block;

PRINTXS — To write all isotope/element Cross Sections of the materials specified in the MIX Data Block as ASCII text files;

PRINTMXS — To write total material (compound) Cross Sections of the materials specified in the MIX Data Block, as ASCII text files.

If this is a proton transport problem, then use:

PRINTPDEDX — To write dE/dX information for protons, for each element in the problem;

PRINTPMDEDX — To write dE/dX information for protons, for each material in the problem.

Example of an I/O Data Block and the results:

```
MIX
  MAT 1  CU 6.5 ZN 2.1
  MAT 2  AL 2.70
I/O PRINTXS PRINTMXS
```

results in the following total cross section files being created in the directory where the COG input file resides.

Files of neutron and photon total cross sections for elements Al, Cu, Zn:

NXS.13027 NXS.29000 NXS.30000

GXS.13000 GXS.29000 GXS.30000

Files of neutron and photon total cross sections for Materials 1 and 2:

NXSMat.0001 NXSMat.0002

GXSMat.0001 GXSMat.0002

The naming convention used is **NXS.zzaaa** for a file containing total **neutron** element/isotope cross sections. The first two digits of the name, **zz**, represent the atomic number *Z* of the element and the last three digits, **aaa**, the isotope number. Files containing total **gamma** element cross sections are represented by **GXS.zz000** (no isotope cross sections). Files containing total cross sections for each **mixture** are labeled **NXSMat.nnnn** or **GXSMat.nnnn**, where **nnnn** is the mixture number.

Each file consists of a few header lines, followed by two columns of data. The first column is the incident particle energy in MeV. The second column is total cross section in barns (for the element/isotope files), or inverse centimeters (for the mixture files). The columns are separated by a tab character to make them easier to import into a plotting program or spread sheet. Each **gamma** isotope listing is approximately 500-1000 lines long and ranges in energy from 1.0×10^{-5} to 1.0×10^5 MeV. The **neutron** cross section listing ranges in energy from 1.0×10^{-11} to 20, 30 or 150 MeV. The length of this listing varies considerably with isotope. Well-researched materials may have listings with more than 18,000 data points.

Example of cross section data text files. The first file is the total gamma cross section file for zinc, GXS.30000. The second file is the total neutron cross section file for mixture 1, NXSMat.0001.

```
(FILE GXS.30000)
TOTAL GAMMA CROSS SECTION FOR ELEMENT 30000
```

ENERGY (MEV)	BARNs
1.000000E-05	1.152182E+06
1.075580E-05	9.351646E+05
1.121240E-05	7.971454E+05
1.162780E-05	6.635631E+05
1.200390E-05	5.406849E+05
1.234290E-05	4.337398E+05
1.245370E-05	3.974940E+05
. . .	
. . .	

```
(FILE NXSMAT.0001)
TOTAL MACROSCOPIC NEUTRON CROSS SECTION FOR MATERIAL 1
FROM COG PROBLEM PRINTCSTEST
```

ENERGY (MEV)	INVERSE CM
9.999900E-11	.000000E+00
1.000000E-10	6.542603E+00
1.000020E-10	6.210052E+00
1.000040E-10	5.946624E+00
1.000060E-10	5.738155E+00
1.000080E-10	5.573505E+00
1.005450E-10	5.474635E+00
1.135960E-10	5.153988E+00
1.259610E-10	4.892130E+00
1.423110E-10	4.620738E+00
. . .	
. . .	

Example: If in the above example we used the I/O statement:

```
I/O PLOTXS PLOTMXS
```

COG would generate plots of the total cross sections, which would be found in the **...ps** graphics file. But the default scaling may not be what the user would desire. For complete control of plotting, the user is advised to use the PRINTXS and PRINTMXS commands to generate the ASCII cross-section listings described above, which can be readily imported into many plotting programs.

8 The ASSIGN Data Blocks

8.1 Assigning Attributes to Sectors

An ASSIGN Data Block allows the user to give sectors physical attributes such as material composition, density, region, and some non-physical attributes such as plotting color. If you do not use ASSIGN statements, COG will make some default assignments for you. These are the following:

reg-ID# = *sect-ID#*; (region ID # = sector ID #)

mat-ID# = *sect-ID#*; (material ID # = sector ID #)

df=1. (relative density factor = 1.)

ASSIGN Data Blocks come in several "flavors" to simplify the assignment process.

8.1.1 ASSIGNING Physical Properties to Sectors

We first discuss all the types of ASSIGN statements which you can use to assign material properties to sectors. Materials are given *mat-ID#(s)* in the MIX Data Block. Sectors are given *sect-ID#s* in the GEOMETRY Data Block.. The statements which link *mat-ID#s* and/or *density factors* to *sect-ID#s* are:

ASSIGN-M;

ASSIGN-

MD;

ASSIGN-D;

ASSIGN-

ML;

ASSIGN (General ASSIGN)

8.1.2 ASSIGN-M — Assign Material to a Sector

The ASSIGN-M Data Block associates a specified sector(s) (with sector number *sect-ID#*) with a material number *mat-ID#*. *sect-ID#* and the desired *mat-ID#* are given in pairs.

ASSIGN-M *sect-ID#₁* *mat-ID#₁* { *sect-ID#₂* *mat-ID#₂* }

. . .

Where:

sect-ID#₁ is the *sect-ID#* of a sector;

mat-ID#₁ is the *mat-ID#* to be assigned to this sector;

This results in sector *sect-ID#₁* containing material *mat-ID#₁*, etc.

Example

```

ASSIGN-M
1 2    3 7  $ ASSIGNS TO SECTOR 1, MATERIAL 2,  AND TO
           $ SECTOR 3, MATERIAL 7
    
```

Default: If ASSIGN statement(s) are *not* used: $mat-ID\# = reg-ID\# = sect-ID\#$, and density factor $df = 1$. :

8.1.3 ASSIGN-MD—Assign *Material and Density* to a Sector

Some problems use the same material in a several sectors with differing *densities*. An example might be a reactor in which water is boiled and converted to steam. The *density* of water would then be different in various parts of the reactor, according to the amount of steam present.

To obtain a density in a SECTOR which is different from the density specified in the MIX Data Block for the material filling the sector, specify a density factor **df** for the sector. The density factor is a multiplier which is applied to the density of the material as defined in its MIX statement. The result is the density for the specified sector. For example, a density factor of two results in twice the density of the material as specified in its MIX statement, for the sector. COG initially sets all density factors to 1, but the user can reset it with one of the ASSIGN statements given below. The ASSIGN-MD Data Block specifies both material number and density factor for a sector(s).

```

ASSIGN-MD sect-ID#1 mat-ID#1 df1
          { sect-ID#2 mat-ID#2 df2 }
          . . .
    
```

Where:

sect-ID#₁ is the **sect-ID#** of a sector;

mat-ID#₁ is the **mat-ID#** to be assigned to this sector;

df₁ is the **density factor** to be applied to the material filling this sector.

This ASSIGN-MD statement results in sector **sect-ID#₁** containing material **mat-ID#₁**, at a relative density of **df₁**, etc.

Example:

```

ASSIGN-MD 1  2  1.5 $ ASSIGNS TO SECTOR 1, MATERIAL 2
                $ WITH DENSITY FACTOR 1.5
    
```

Default: If ASSIGN statement(s) are *not* used: $mat-ID\# = sect-ID\#$, and density factor $df = 1$.

8.1.4 ASSIGN-D — Assign *Density* to a Sector

The ASSIGN-D Data Block assigns density factors to sectors.

ASSIGN-D *sect-ID#₁* *df₁* {*sect-ID#₂* *df₂* }

. . .

Where:

sect-ID#₁ is the *sect-ID#* of a sector;

df₁ is the **density factor** to be assigned to this sector. The density factor is a multiplier which is applied to the density of the material assigned to this sector, to produce the density for this sector. For example, a density factor of two results in twice the density of the material as specified in its MIX statement, for this sector.

This ASSIGN-D statement results in sector *sect-ID#₁* containing its assigned material at a density of *df₁*, relative to the density of its material as specified in the MIX statement.

Example:

```
ASSIGN-D
1  1.5      3  1.7      $ ASSIGNS SECTOR 1 A DF OF 1.5
                                $      AND SECTOR 3 A DF OF 1.7
```

Default: If ASSIGN statement(s) are not used, density factor *df* = 1.

8.1.5 ASSIGN-ML — Assign *Materials* to a *List* of Sectors

ASSIGN-ML provides a *list* of sectors to be assigned a specified material.

ASSIGN-ML *mat-ID#₁* *sect-ID#₁₁* *sect-ID#₁₂* *sect-ID#₁₃* . . .
 { / *mat-ID#₂* *sect-ID#₂₁* *sect-ID#₂₂* *sect-ID#₂₃* . . . }
 . . .

Where:

mat-ID#₁ is the *mat-ID#* to be assigned to the following sector list;

sect-ID#₁₁ *sect-ID#₁₂* *sect-ID#₁₃* . . . is a list of *sect-ID#* s; : #.

This causes the listed sectors to contain material *mat-ID#₁*.

Other *mat-ID#*s and *sect-ID#* lists may follow, separated by a "/" .

Example

```
ASSIGN-ML
1 2 4 5 7      $ ASSIGNS MATERIAL 1 TO SECTORS 2 4 5 7
/ 2 10 11 12   $ ASSIGNS MATERIAL 2 TO SECTORS 10 11 12
```

Default: If ASSIGN statement(s) are not used: *reg-ID#* = *sect-ID*

8.1.6 ASSIGNing REGIONS and COLORS to Sectors

Other forms of the **ASSIGN** statement are used to assign other properties to user's sectors. The following **ASSIGN** statements are described here:

- **ASSIGN-R, ASSIGN-RL**: To associate **REGIONS** with sectors;
- **ASSIGN**: To associate **MATERIALs and REGIONS and density factors** with sectors;
- **ASSIGN-MC**: To associate plotting colors with sectors;

A **REGION** is a set of sectors, specified by the user, which is to be treated as a single volume, for purposes of controlling the random-walk or analyzing results. A user may wish to define one or more **REGIONS** in order to:

- Use the same random-walk modification parameters throughout the **REGION**;

- Tabulate the events that occur in the **REGION** as if it were a single sector;
- Define a multi-sector detector volume for scoring results.

Each **REGION** is identified by a user-assigned *reg-ID#*, a positive integer. It may be helpful to use a **REGION** numbering scheme which differs from the **SECTOR** numbering (*sect-ID#s*), in order to avoid confusing these two entities. .

Note that a **REGION** specification does not change the geometry or the material assignments in any way. It is purely a sector classification convention for convenience. **REGION** numbers are frequently required in the **WALK, ANALYSIS,** and **DETECTOR** Data Blocks.

Default: Each sector is "born" with a *reg-ID#* = *sect-ID#*. This may be changed with an **ASSIGN** statement

8.1.7 ASSIGN-R — Assign Region Number to a Sector

The **ASSIGN-R** Data Block associates a specified sector with a region:

```
ASSIGN-R sect-ID#1 reg-ID#1 { sect-ID#2 reg-ID#2 }
. . .
```

Where:

sect-ID#₁ is the *sect-ID#* of a sector;

reg-ID#₁ is the *region-ID#* to be assigned to this sector;

Example:

```
ASSIGN-R
1 2      3 7  $ ASSIGNS SECTOR 1 TO REGION 2 AND
               $ SECTOR 3 TO REGION 7
```

Default: If **ASSIGN** statement(s) are *not* used: *reg-ID#* = *sect-ID#*.

8.1.8 ASSIGN-RL — Assign Region Number to a List of Sectors

ASSIGN-RL provides a *list* of sectors to be assigned a specified statistical REGION number.

```
ASSIGN-RL reg-ID#1 sect-ID#11 sect-ID#12 sect-ID#13 . . .
          { / reg-ID#2 sect-ID#21 sect-ID#22 sect-ID#23 . . . }
          . . .
```

Where:

reg-ID#₁ is the *region-ID#* to be assigned to this sector list;

sect-ID#₁₁ sect-ID#₁₂ sect-ID#₁₃ . . . is a list of *sect-ID#s*;

This results in the listed SECTORS being assigned to a specified REGION.

Other *reg-ID#s* and *sect-ID#* lists may follow, separated by a "/" .

Example:

```
ASSIGN-RL
  20 2 4 5 7 $ ASSIGNS REGION 20 TO SECTORS 2 4 5 7
/ 22 5 8      $ AND ASSIGNS REGION 22 TO SECTORS 5 8
```

Default: If ASSIGN statement(s) are *not* used: *reg-ID#* = *sect-ID#*.

8.1.9 ASSIGN (General)— Assign Material Number, Region Number, and Density Factor to a Sector

The general ASSIGN Data Block associates a specified sector with a material number *mat-ID#*, a region number *reg-ID#*, and a density factor *df*.

```
ASSIGN sect-ID#1 mat-ID#1 reg-ID#1 df1
      { sect-ID#2 mat-ID#2 reg-ID#2 df2 }
      . . .
```

Where:

sect-ID#₁ is the *sect-ID#* of a sector;

mat-ID#₁ is the *mat-ID#* to be assigned to this sector;

reg-ID#₁ is the *reg-ID#* to be assigned to this sector;

df₁ is the *density factor* to be assigned to this sector. The density factor is a multiplier which is applied to the density of the defined material, to produce the density for this sector. For example, a density factor of two results in twice the density of the material as specified in its MIX statement, for this sector.

This ASSIGN statement results in sector *sect-ID#₁* containing material *mat-ID#₁* at a relative density of *df₁*. The sector is also assigned a region number of *reg-ID#₁* .

Example:

```

ASSIGN
1 2 30 1.5 $ ASSIGNS TO SECTOR 1, MATERIAL 2, AND
              $ REGION 30, WITH DENSITY FACTOR OF 1.5
    
```

Default: If *ASSIGN* statement(s) are not used: *mat-ID#* = *reg-ID#* = *sect-ID#*, and *density factor df* = 1

8.1.10 ASSIGN-MC— Assign Plotting Colors to Materials

PICTUREs of the geometry may have areas filled with colors, which are keyed to the sector materials. See section **PICTURES of the Geometry** for information on the color option for cross-section and perspective pictures.

By default, colors are chosen by the code from a palette of 20 colors. To assign specific colors to materials, use the ASSIGN-MC statement:

```

ASSIGN-MC mat-ID#1 c1 { mat-ID#2 c2 } . . .
    
```

Where:

mat-ID#₁ is the *mat-ID#* to be assigned a color;

c₁ is the ASCII *color name* to be assigned to material *mat-ID#₁*.

Example:

```

ASSIGN-MC
1 SKY $ ASSIGN TO MATERIAL 1, COLOR SKY
2 YELLOW $ ASSIGN TO MATERIAL 2, COLOR YELLOW
3 ROSE $ ASSIGN TO MATERIAL 3, COLOR ROSE
    
```

When PICTUREs are subsequently drawn, areas in the picture which represent various materials will be drawn in the specified color.

COG Color Table

(parenthesized words are tint descriptions, not Color Names)

BLACK	WHITE	RED	GREEN (dark)	BLUE (dark)
YELLOW	MAGENTA	CYAN (blue)	GRAY	PINK
SKY (blue)	PURPLE	TAN	ORANGE	LAVENDER
BROWN	MOSS (green)	LIME (green)	ROSE	SLATE (blue)

8.1.11 Summary of ASSIGN Data Blocks

ASSIGN Block	Form	Affects
ASSIGN	<i>sect-ID#₁ mat-ID#₁ reg-ID#₁ df₁</i>	all
ASSIGN-M	<i>sect-ID#₁ mat-ID#₁</i>	Material
ASSIGN-R	<i>sect-ID#₁ reg-ID#₁</i>	Region
ASSIGN-D	<i>sect-ID#₁ df₁</i>	Density factor
ASSIGN-MD	<i>sect-ID#₁ mat-ID#₁ df₁</i>	Material, Density
ASSIGN-ML ¹	<i>mat-ID#₁ sect-ID#₁₁ sect-ID#₁₂ . . .</i>	Material List
ASSIGN-RL ¹	<i>reg-ID#₁ sect-ID#₁₁ sect-ID#₁₂ . . .</i>	Region List
ASSIGN-MC	<i>mat-ID#₁ c₁</i>	Material Color

Where df = density factor,
 c_1 = color name

¹Requires "/" to separate successive lists.

Default: If ASSIGN statement(s) are *not* used: *mat-ID#* = *reg-ID#* = *sect-ID#*, and density factor *df* = 1.

8.1.12 Multiple Use of ASSIGN Blocks

There are seven different types of ASSIGN statements that allow the default values of *mat-ID#*, *reg-ID#*, and density factor *df* to be changed. Any, all, or none of these Data Blocks may be employed as the user sees fit. However, ***only one*** Data Block ***of a given type*** may be used. If any assignment is made more than once, the *last* such assignment *prevails*.

8.1.13 Examples of ASSIGN Data Blocks

As an example, let us look at a problem with four sectors. We will change these around (albeit in a very unrealistic way) to illustrate the use of ASSIGN statements.

Without any ASSIGN statements, we have these default values:

$mat-ID\# = sect-ID\# = reg-ID\#$, and density factor $df = 1$.

Default Case:	<i>Sector 1</i>	<i>Sector 2</i>	<i>Sector 3</i>	<i>Sector 4</i>
<i>Mat-ID#</i>	1	2	3	4
<i>Reg-ID#</i>	1	2	3	4
Density factor <i>df</i>	1.0	1.0	1.0	1.0

Let's use an ASSIGN statement to change the *mat-ID#*, *region-ID#*, and density factor *df* for two of our sectors:

```

ASSIGN 1  2  1  1.5  $ ASSIGN TO SECTOR 1, MATERIAL #2,
                        $ REGION #1, DF =1.5
          3  2  1  0.75 $ ASSIGN TO SECTOR 3, MATERIAL #2,
                        $ REGION #1, DF =0.75
    
```

This changes the Default case to Revision 1:

Revision 1	<i>Sector 1</i>	<i>Sector 2</i>	<i>Sector 3</i>	<i>Sector 4</i>
<i>Mat-ID#</i>	2	2	2	4
<i>Reg-ID#</i>	1	2	1	4
Density factor <i>df</i>	1.5	1.0	0.75	1.0

(Note that the properties for sectors 2 and 4 go unchanged).

Let's now use the ASSIGN-M statement to change the material and the ASSIGN-D statement to change the density. We obtain the results shown in REVISION 2.

For Example:

```

ASSIGN-M  2  5  3  6  $ ASSIGN TO SECTOR 2, MATERIAL 5, AND
                        $ TO SECTOR 3, MATERIAL 6
ASSIGN-D  4  2.7      $ ASSIGN TO SECTOR 4, A DENSITY
                        $ FACTOR OF 2.7
    
```

We then have for Revision 2:

Revision 2	<i>Sector 1</i>	<i>Sector 2</i>	<i>Sector 3</i>	<i>Sector 4</i>
<i>Mat-ID#</i>	2	5	6	4
<i>Reg-ID#</i>	1	2	1	4
Density factor <i>df</i>	1.5	1.0	0.75	2.7

Note: Sector 1 was not respecified and therefore did not change.

Let's now use the **ASSIGN-RL** statement to assign several SECTORS to a common REGION.

For Example:

```
ASSIGN-RL 10 1 2 3 $ ASSIGN REGION #10 TO SECTORS 1,2,3
          / 12 4 $ AND ASSIGN REGION #12 TO SECTOR 4
```

This assignment would give us Revision 3:

Revision 3	<i>Sector 1</i>	<i>Sector 2</i>	<i>Sector 3</i>	<i>Sector 4</i>
<i>Mat-ID#</i>	2	5	6	4
<i>Reg-ID#</i>	10	10	10	12
Density factor <i>df</i>	1.5	1.0	0.75	2.7

If by mistake we would now use the additional assignment statement:

```
ASSIGN 2 2 2 1.0
```

COG would produce a *fatal input error message*, since one ASSIGN statement was already specified.

9 The SOURCE Data Block

9.1 Outline of General Method

Each COG problem requires that a SOURCE Data Block be specified. (Exception: a CRITICALITY problem has a special neutron source.) Most sources can be simply described to COG

A complete source description includes

- a **POSITION** definition, i.e. *where* the source is located;
- an **ENERGY** definition, i.e. the particle *type* and *energy* distribution;
- an **ANGLE** definition, i.e. how the source particles are *directed*;
- a **TIME** definition, i.e. the *time* dependence of the source.

These parts are described in the following sections:

Source POSITION Dependence

Source ENERGY Dependence

Source ANGLE Dependence

Source TIME Dependence

A general particle source can be written as

where spatial (x, y, z) position of the source, **e** is the particle energy,

$S(\bar{\mathbf{r}}, \mathbf{e}, \bar{\boldsymbol{\Omega}}, t)$

Where $\bar{\mathbf{r}}$ is the spatial (x,y,z) position of the source, **e** is the particle energy, $\bar{\boldsymbol{\Omega}}$ direction (θ, ϕ) of the emitted particles, and **t** is the time of emission. In nearly all cases, the source is separable in the individual variables. For example, we often speak of a source at a given position, with a specific energy, radiating isotropically, and at a fixed time. This ability to separate the parameters implies that the source can be written as a product of terms:

$$S(\bar{\mathbf{r}}, \mathbf{e}, \bar{\boldsymbol{\Omega}}, t) = P(\bar{\mathbf{r}})E(\mathbf{e})A(\bar{\boldsymbol{\Omega}})T(t)$$

where each of these terms is a function of a single source variable and can be specified independently of the other terms. For the occasional source that cannot be exactly separated into independent source terms, it can nearly always be approximated by a set of source "pieces", known to COG as

increments. Each increment is a complete description of some part of the source. The i -th increment S_i is given by:

$$S_i(\bar{r}, e, \bar{\Omega}, t) = P_i(\bar{r}) E_i(e) A_i(\bar{\Omega}) T_i(t)$$

where P_i, E_i, A_i, T_i describe particular parts of the source dependence. The total source is a sum over all increments:

$$S(\bar{r}, e, \bar{\Omega}, t) = \sum_{i=1}^n P_i(\bar{r}) E_i(e) A_i(\bar{\Omega}) T_i(t)$$

where n is the number of source increments.

9.1.1 Format of the SOURCE Data Block

A COG SOURCE specification consists of one or more source INCREMENTS. Each INCREMENT requires a POSITION, ENERGY, ANGLE, and TIME dependence to be DEFINED (some may be omitted). The general format of the SOURCE Data Block is:

SOURCE

miscellaneous source parameters

DEFINE POSITION 1

definition of position-dependence #1

DEFINE POSITION 2

definition of position-dependence #2

...

DEFINE ENERGY 1

definition of energy-dependence #1

DEFINE ENERGY 2

definition of energy-dependence #2

...

DEFINE ANGLE 1

definition of angle-dependence #1

DEFINE ANGLE 2

definition of angle-dependence #2

```

...
DEFINE TIME 1
    definition of time-dependence #1
DEFINE TIME 2
    definition of time-dependence #2
...
INCREMENT strength POSITION=1 ENERGY=2 ANGLE=1
                        TIME=3
INCREMENT strength POSITION=2 ENERGY=2 ANGLE=3
                        TIME=1
...

```

where:

strength is the relative source strength of the INCREMENT;

The **position-, energy-, angle-, and time-dependences** to be associated with each source INCREMENT are given by the ID#s which follow the keywords **POSITION, ENERGY, ANGLE, and TIME**.

Note: The same definition of **POSITION, ENERGY, ANGLE, or TIME** may be used by more than one INCREMENT.

ANGLE and **TIME** dependences may be omitted. COG will use default definitions for them.

9.1.2 Special Free-Form Inputs

COG offers some notational shortcuts to speed input of repeated or interpolated lists of numbers (see the **Introduction** section for a more complete discussion).

To **repeat** a number or a group of numbers, use:

n [m1 m2 m3]

This generates **n** instances of the numbers **m1 m2 m3**.

For example,

3 [1.1 2.2] is equivalent to 1.1 2.2 1.1 2.2 1.1 2.2

To **linearly interpolate** values between given end values, use:

n [m1 i m2]

This generates **n** linearly interpolated values, starting at **m1** and ending at **m2**.

For *example*,

5 [1. i 5.] is equivalent to 1. 2. 3. 4. 5.

To *logarithmically interpolate* values between given end values, use

n [m1 L m2]

This generates **n** logarithmically interpolated values, starting at **m1** and ending at **m2**.

The symbol between **m1** and **m2** is the letter **L** (or **l**).

For *example*,

5 [1. l 5.] is equivalent to 1. 1.49 2.24 3.34 5.

Note: Delimiting spaces are not needed around brackets but are needed around the **i** and **L** inside the brackets.

9.2 Miscellaneous Parameters

After the SOURCE Data Block name and before the DEFINE and INCREMENT specifications, four miscellaneous parameters may be specified. These are:

NPART

WRITESOURCE

READSOURCE

RETRACE

CORRELATED

9.2.1 NPART – Number of Source Particles

NPART defines the number of source particles to be run in this problem.

It must **always** be specified.
$$\left[\begin{array}{c} \text{NPARTICLES} \\ \text{NPARTICLE} \\ \text{NPART} \end{array} \right] = \text{number-of-particles}$$

where:

number-of-particles is the number of particles (*Maximum Particles* = 1E+15) to be run by COG in this problem.

Default: None; NPART must be specified.

Each source particle has its own starting random-number seed used to initiate its random walk. COG will randomly generate the specified *number-of-particles*, choosing particle starting parameters (position, energy, time, direction, etc.) from the distributions specified in the SOURCE Data Block.

Exception: The user may specify that particle starting parameters are to be read from a COG SOURCE file, produced by a previous COG run.

9.2.2 WRITESOURCE – Produce a COG SOURCE File

COG reads the SOURCE Data Block DEFINE and INCREMENT specifications (see below) and translates them into parameters which completely describe the source. If the user has specified the WRITESOURCE option, these parameters are written into an output SOURCE file named *inputfile.sor*. The file will also contain the random-number-generator sequence number used to create each source particle during the COG run. The created SOURCE file will reside in the same directory as the input file. This file may be used as an input SOURCE file by subsequent COG jobs, when the user desires to run COG in the CORRELATED mode (see below). To create a SOURCE file, include the following statement in your SOURCE Data Block after the NPART specification:

WRITESOURCE

9.2.3 READSOURCE – Read a Previously-made SOURCE File

To run in the CORRELATED mode (see below), COG requires as input the SOURCE file created during a prior run when the WRITESOURCE option was specified. To read in a previously-made SOURCE file, include the following statement in your SOURCE Data Block:

READSOURCE *filename*

where *filename* is the name of the SOURCE file to be re-used.

Note: Unlike most COG input, *filenames* are case sensitive, i.e. they must be typed *exactly* as they appear on the computer. No embedded blanks are allowed. And unlike a string used in a COG title, the filename is *not* surrounded by ".

The **READSOURCE** statement must immediately follow the **NPART** statement. Any **DEFINE** and **INCREMENT** specifications which follow the **READSOURCE** statement are ignored, since the source is completely described by the SOURCE file. If the **READSOURCE** option is used, the requested SOURCE file *filename* must exist in the same directory as the input file, and no new SOURCE file will be generated. In addition, the value of **NPART** may not exceed the number of particles actually contained in the input SOURCE file, although it may be less.

Example of an input that uses the READSOURCE option:

```
SOURCE
      NPART 500000
READSOURCE COGSIN01.SOR
```

9.2.4 RETRACE Source Particles

The RETRACE option in the SOURCE Data Block allows you to recalculate the trajectories of previously-run source particles and produce a full listing of all interactions and “events” that happened to the particle. This is useful for understanding in detail how “interesting” or important particles contribute to a detector score. For every DETECTOR in a problem, COG lists the **starting random-number-generator sequence #s** of the ten particles that contributed the most to the score. An example is:

```
SOURCE PARTICLES MAKING THE TEN LARGEST CONTRIBUTIONS TO
TOTAL RESPONSE
```

Event histories for these may be obtained by re-running with RETRACE option

Source Particle RNG Sequence #	Score as % of total result	Cumulative Score as % of total result
190248	.5119	.5119
107022	.4501	.9620
400123	.3792	1.3412
178383	.3180	1.6591
291947	.2943	1.9534
283331	.2889	2.2423
395136	.2853	2.5275
155233	.2817	2.8093
220569	.2802	3.0895
173727	.2408	3.3303

If one or more particles make unusually large contributions to the detector score, you might well be interested in learning how this happened. The RETRACE option allows you to find out.

Modify the SOURCE Data Block in the input file to specify the random-number-generator sequence numbers of the particles you wish to RETRACE. The format of the new line is:

SOURCE

RETRACE seq#1 seq#2 ...

where:

seq#1 seq#2 ... are the random-number-generator sequence numbers to be retraced.

If you did not specify the starting random number seeds in the prior run, you must also modify the BASIC Data Block in the input file to specify the starting random number seeds which COG used in the prior run. These starting seeds are listed in the COG printout. The format of the line to specify the starting seeds is:

BASIC

RN rn1 rn2

where:

rn1 rn2 are the random-number-generator starting seeds.

With these modifications to the input file, COG will retrace the specified particles. No other particles will be run. The RETRACE results will appear in the ...out file.

Make no other changes to the input file.

Example: We RETRACE particles from the COG run listed above whose random-number-generator sequence numbers are 190248 and 107022. In the COG out file we find the following line which gives us the starting seed information:

```
STARTING RANDOM NUMBER SEEDS FOR TRANSPORT PHASE = 2 4
```

Insert the following line after the NPART statement in the SOURCE Data Block in the input file:

```
RETRACE 190248 107022    $ PARTICLE RNG SEQ.#'S TO BE
                        $ RETRACED
```

Insert the following line in the BASIC Data Block in the input file:

```
RN 2 4 $ RNG STARTING SEEDS
```

*This run will **only** RETRACE the particles listed and will do no other calculations.*

These RETRACES can give important clues as to how the problem might be biased or reorganized to give more reliable answers.

The output for a single RETRACED particle is shown below. In going from "event" to "event", only the quantities which change are listed. For example, under "event" 2 the particle parameters x, y, z are listed because they have changed since the previous "event", while particle type, energy, direction, etc. are not listed because they have not changed.

```
COMPLETE EVENT-HISTORY FOR SOURCE PARTICLE NUMBER 466
1 particle taken from initial source specifications
  x= 1.263076E-01    y= 7.377586E+01 z= 1.168666E-01
  u=-1.159619E-02    v=-9.998500E-01 w=-1.286660E-02
  age=      .000000E+00
  photon      E= 5.271478E-02 vel= 2.997925E+10
  statistical weight= 5.749345E-02
  number of collisions=      0
  material number=      0 region number=      0
  density factor=      .000000E+00 tot Xsect=.000000E+00
  mean-free-paths left to travel= 4.485420E-01
  distance to next boundary 6.440608E+01
2 crossed a boundary between two different regions
  x=-6.205577E-01    y= 9.379442E+00 z=-7.118208E-01
  age= 2.148356E-09
  material number=      4 region number= 14
  density factor= 1.000000E+00 tot Xsect= 2.830805E-01
  cosine of angle from trajectory to normal=.000000E+00
  distance to next boundary 1.399015E+01
3 enters into a collision
```

```

x=-6.389319E-01 y= 7.795177E+00 z=-7.322079E-01
age= 2.201209E-09
material number= 4 region number= 14
4 exits from a collision
u=-2.679967E-01 v=-8.493922E-01 w=-4.546544E-01
photon E= 5.195490E-02 vel= 2.997925E+10
number of collisions= 1
material number= 4 region number= 14
density factor= 1.000000E+00 tot Xsect= 2.839803E-
01 reaction number= 72
compton mean-free-paths left to travel= 5.775473E-
01
Energy dep. times statistical wt. = 4.368820E-05
distance to next boundary 1.964484E+00
5 crossed a boundary between two different regions
x=-1.165407E+00 y= 6.126559E+00 z=-1.625369E+00
age= 2.266737E-09
material number= 0 region number= 0
density factor= .000000E+00 tot Xsect=.000000E+00
mean-free-paths left to travel= 1.967233E-02
cosine of angle from trajectory to
normal=.000000E+00
distance to next boundary 1.326312E+00
6 crossed a boundary between two different regions
x=-1.520855E+00 y= 5.000000E+00 z=-2.228383E+00
age= 2.310978E-09
cosine of angle from trajectory to
normal=.000000E+00
distance to next boundary 9.395629E+00
7 crossed a boundary between two different regions
x=-4.038852E+00 y=-2.980574E+00 z=-6.500148E+00
age= 2.624383E-09
cosine of angle from trajectory to
normal=.000000E+00
distance to next boundary 2.377496E+00

```

9.2.5 CORRELATED Source Particles

The **CORRELATED** option lets you run a series of problems which differ only slightly. For example, you might wish to evaluate the effects of different filters placed in some small region of your geometry. In the **CORRELATED** mode a previously-generated **SOURCE** file is read in, and particles are started from the same source with the same initial parameters and the same starting random-number-generator sequence numbers as used in the previous run. The idea is that because the difference in problems is slight, only trajectories affected by the changes will be different. This implies that the difference in results between the **CORRELATED** problems will be

more meaningful than the difference in results between two uncorrelated runs, because the statistical fluctuations caused by using different source samples is eliminated.

The **CORRELATED** option is also used with the Lucky Particle detector option. Running in the Lucky Particle mode causes the**sor** file to contain the random number seeds for only those source particles which reach the designated Lucky Particle detector boundary. Detector design studies can then be efficiently done by modifying the detector properties and rerunning in the **CORRELATED** mode, which will transport only those particles known to reach the detector.

To run in **CORRELATED** mode, add to the SOURCE Data Block the information below:

READSOURCE *infile.sor*

CORRELATED

where:

infile.sor is the name of the source file created in a previous run of COG on input file *infile*.

the **CORRELATED** results appear in the new**out** file.

Example: Do a **CORRELATED** run using the **SOURCE** file from the COG run *COGin1*. Insert the following line after the **NPART** statement in the **SOURCE Data Block**:

```
READSOURCE COGin1.sor
CORRELATED
```

Any of the standard textbooks¹ discuss this very powerful method in greater detail.

9.3 Source INCREMENTs

A SOURCE INCREMENT specifies the position, angle, energy, and time dependence of one part of the total COG SOURCE. The simplest SOURCE has just a single INCREMENT. The INCREMENT statement has the form:

$$\begin{array}{c} \left[\begin{array}{c} \text{INCREMENT} \\ \text{INC} \end{array} \right] \text{source-strength} \left[\begin{array}{c} \text{POSITION} \\ \text{P} \end{array} \right] pd\# \\ \\ ed\# \left\{ \left[\begin{array}{c} \text{ANGLE} \\ \text{A} \end{array} \right] ad\# \right\} \\ \\ \left\{ \left[\begin{array}{c} \text{TIME} \\ \text{T} \end{array} \right] td\# \right\} \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] = i \right\} \end{array}$$

where:

source-strength is the source intensity for this INCREMENT.

For a STEADY state time-dependent source, the *source-strength* is the number of particles emitted by this INCREMENT *per unit time*, using the units of time specified in the BASIC Block. The scores for this problem will be given in the output as results *per second*.

For all other source types, *source-strength* is just the **relative** strength of this INCREMENT compared to other INCREMENTS, and is used to normalize the problem results to results *per source particle, times the increment source strength*. The *source-strength* can be any value greater than zero.

COG samples INCREMENTS with a frequency which depends on the relative *source-strength* of each INCREMENT, weighted by its relative IMPORTANCE. For example, if one INCREMENT has twice the *source-strength* as another (and default IMPORTANCES ($i = 1$) are used), then it will be sampled twice as often; i.e., COG will generate twice as many source particles from this INCREMENT than from the other.

pd# is the ID# of the POSITION definition for this increment;

ed# is the ID# of the ENERGY definition for this increment;

ad# is the ID# of the (optional) ANGLE definition for this increment;

td# is the ID# of the (optional) TIME definition for this increment.

Note: All ID#s must be positive integers.

Defaults:

POSITION dependence: None; must be specified.

ENERGY dependence: None; must be specified.

ANGLE dependence: Isotropic.

TIME dependence: Delta function at $t=0$. This choice results in scores *per source particle*.

In the (optional) IMPORTANCE field, i is the relative importance of particles born in this increment compared to the importance in other increments. INCREMENTS with larger values of i will be sampled more frequently than those with smaller values of i . See the discussion of Source IMPORTANCES in the next Section.

Default: If omitted, the IMPORTANCE is set to 1.

Example: Here three INCREMENT statements are used in the SOURCE Data Block. The POSITION, ENERGY, and ANGLE dependences referred to in these INCREMENT statements must be defined elsewhere in the Data Block. By default, all INCREMENTS have unit IMPORTANCE.

```
INC 1.085172  P 1  E 1  A 1
INC 1.084079  P 1  E 2  A 2
INC 1.080834  P 1  E 3  A 3
```

Note: As shown here, INCREMENTS may share definitions of POSITION, ENERGY, ANGLE, and TIME.

More examples of SOURCE INCREMENTS are given below, following the sections on DEFINING the POSITION, ENERGY, TIME, and ANGLE dependence of the problem sources.

During particle generation at the source, the code keeps track of the minimum, maximum, and average starting parameters for each particle. These values are later printed so the user can check that the source is correct.

Example: Scoring Particles from Multiple Increments

Consider a two-increment source, which produces scores in a detector. All weights are unity. All importances are unity. COG will sample the two increments proportionately to their strengths. The Raw Score for the detector is the sum of weights of scoring particles from the specified increments.

Inc #	Source Strength	Npart per Inc	Fract. Scoring	Raw Score	Score per Inc. Particle	Scaled by Src. Inten.
1	10.e5	100	1/2	50	.5	5.0e5
2	1.e5	10	1/2	5	.5	0.5e5

Tot:	11.e5	110		55		5.5e5

9.4 Source IMPORTANCES

Each SOURCE INCREMENT may be given a specified IMPORTANCE. The format for this input is:

$$\left\{ \begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right\} = \mathbf{i}$$

The purpose of using non-uniform SOURCE IMPORTANCES is to cause COG to create and follow more particles from those SOURCE increments which contribute more to the result at your detector (i.e., have greater IMPORTANCE) than those which contribute relatively little. The code automatically modifies the weight of these more "important" particles so that the correct answer is still obtained, but it is an answer with improved statistics.

COG samples INCREMENTS with a frequency which depends on the relative *source-strength* of each INCREMENT, weighted by its relative IMPORTANCE. Numerically, the expected number of particles $\langle N_j \rangle$ to be generated in an INCREMENT of *source-strength* s_j and *importance* I_j is:

$$\langle N_j \rangle = \text{NPART} \left[\frac{S_j I_j}{\sum_j S_j I_j} \right]$$

For example, if one INCREMENT has twice the *source-strength* as another, but it has one half the IMPORTANCE, then both INCREMENTS will be sampled with equal number of particles; but the particles from the INCREMENT with half the *source-strength* will also have one half the weight. Since the quantity scored by a COG detector is particle weight, this weight adjustment preserves the score. IMPORTANCE biasing of the SOURCE is meant to improve the run efficiency, so that for a fixed number of COG particles, you will get a more precise answer than if you had used the default condition of uniform INCREMENT IMPORTANCES.

COG will adjust the particle weights for any SOURCE INCREMENT, ENERGY, ANGLE and TIME IMPORTANCE biasing, so that unbiased DETECTOR results are obtained, regardless of the IMPORTANCES.

There are two general cases where one would use INCREMENT biasing. Consider a particle source which has a rather large spatial extent but only a few source regions which contribute much to the desired result. The user might then wish to break the volume source into a number of increments, each of which represents just part of the total volume. Those increments

contributing most of the result would then be given large IMPORTANCE values while those contributing little would be given smaller

IMPORTANCE values. This effectively allows importance sampling of the spatial distribution of the source. The second case involves a problem with both neutrons and photons. Generally the source strengths are much larger for the photon sources than for the neutron sources. You can assign IMPORTANCE values to the INCREMENTS to alter the selection process, so approximately the same number of source neutrons as source photons are followed. In both cases, of course, the initial particle weights are modified by COG so that the problem solution remains correct.

Example of specifying IMPORTANCES for INCREMENTS. Here the third INCREMENT has twice the IMPORTANCE, and will be sampled twice as often, as the first and second SOURCE INCREMENTS.

```
INC 1.08 P 1 E 1 A 1 IMP 0.5
INC 1.08 P 1 E 2 A 2 IMP 0.5
INC 1.08 P 1 E 3 A 3 IMP 1.0
```

*Example: Form a two-INCREMENT SOURCE description for a neutron/photon problem. The POSITION, ENERGY, and ANGLE dependences referred to in these INCREMENT statements must be defined elsewhere in the Data Block. If we specified no IMPORTANCES for this problem, the number of neutrons generated would only be 1/5 the number of photons, as dictated by their relative **source-strengths** ($3.70 \times 10^{10} / 1.85 \times 10^{11}$). If we wish to track as many neutrons as photons (in order to improve neutron statistics, say) we can specify increased neutron IMPORTANCE.*

```
INCREMENT 3.70E10 P=1 E=1 T=1 IMP 5. $NEUTRON SOURCE
INCREMENT 1.85E11 P=1 E=2 T=1 IMP 1. $PHOTON SOURCE
```

*By increasing the relative neutron IMPORTANCE by a factor of 5, the product of **source-strength** and IMPORTANCE is now the same for both INCREMENTS, and roughly equal numbers of neutrons and photons will be generated.*

Below is the COG printout for this problem. Note that the total source strength by particle type is summed for the user's convenience in checking the normalization of the problem. Where TIME or ANGLE description ID#s are listed as zero, default TIME or ANGLE descriptions were used in the problem. Approximately 5000 neutrons and 5000 photons were generated, as desired.

COMPLETE SOURCE DESCRIPTION

TOTAL NUMBER OF SOURCE PARTICLES TO BE GENERATED = 10000

INCREMENT	SOURCE STRENGTH		DESCRIPTION OF SEPARABLE PARTS	RELATIVE IMPORTANCE
			POSITION ENERGY TIME ANGLE	
1	3.7000E+10	N	1 1 1 0	5.0000E+00
2	1.8500E+11	P	1 2 1 0	1.0000E+00
TOTALS				
	3.7000E+10	N		
	1.8500E+11	P		

SUMMARY OF RANDOM-WALK EVENTS FOR 10000 SOURCE PARTICLES
REMOVAL AND ADDITION WEIGHTS AND DEPOSITED ENERGY ARE NORMALIZED
TO ONE SOURCE PARTICLE

REGION NUMBER 1		# OF	REMOVAL	ADDITION	DEPOS
EVENT	PARTICLE	EVENTS	WEIGHT	WEIGHT	ENERGY (MEV)
-----	-----	-----	-----	-----	-----
FIXED SOURCE	NEUTRON	4977	.0000E+00	3.6831E+10	.0000E+00
	PHOTON	5023	.0000E+00	1.8584E+11	.0000E+00

Each ENERGY, TIME, and/or ANGLE dependence which you specify for a SOURCE may also have an IMPORTANCE field. In this field you can specify the relative IMPORTANCE of each interval of ENERGY, TIME or ANGLE. The purpose of using non-uniform SOURCE IMPORTANCES in a problem is to cause COG to create and follow more particles from those intervals (in ENERGY, TIME, or ANGLE) which contribute more to the result at your detector (i.e., have greater IMPORTANCE) than those which contribute relatively little. For example, if one energy interval is assigned twice the importance of another, it will be sampled twice as often. As for INCREMENT biasing, biasing of an ENERGY-, TIME-, and/or ANGLE-dependent SOURCE means that for a fixed number of COG particles run, you will get a more precise answer than if you had used the default condition of uniform SOURCE IMPORTANCES. Note that importance biasing does not change

the result of a run, unless this biasing causes volumes of phase space which contribute significantly to your answer go under sampled, in which case you can get the wrong answer.

9.5 Source POSITION Dependence

The POSITION dependent SOURCE specification is started with:

$$\text{DEFINE } \left[\begin{array}{c} \text{POSITION} \\ \mathbf{P} \end{array} \right] = pd\#$$

where *pd#* is any positive integer chosen by the user to identify this definition.

This statement is then followed by the data for one of the POSITION definitions listed below. You will need a DEFINE POSITION definition for each different position-dependence in your problem.

There are four classes of POSITION dependent sources:

POINT sources;

LINE sources, with a uniform source strength per unit length;

SURFACE sources, with uniform source strengths per unit area;

VOLUME sources, with uniform source strengths per unit volume.

The coordinates used to describe position dependence are in the default units of centimeters, unless specified otherwise in the BASIC Data Block.

Default: None; a POSITION dependence must be specified.

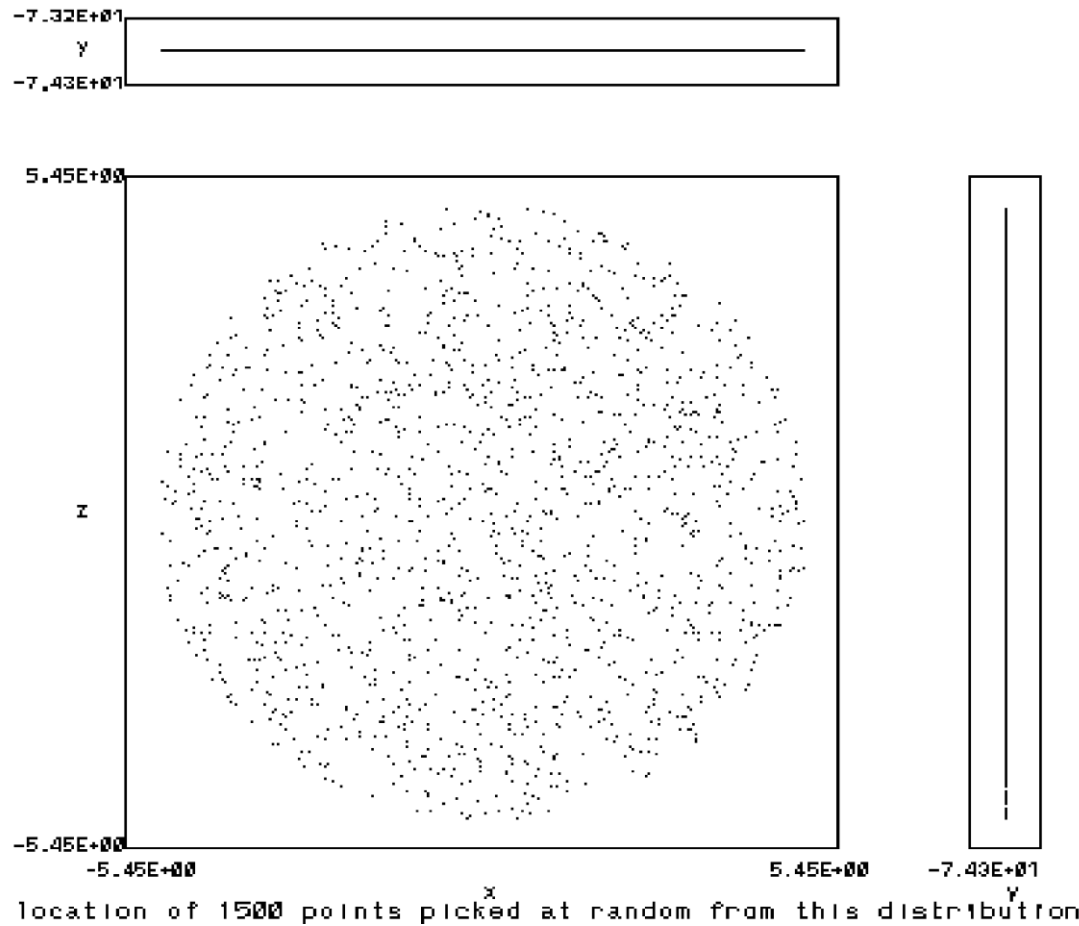
9.5.1 Source POSITION Graphics

COG aids the user in debugging his source POSITION dependence specification by plotting the positions of several thousand points randomly chosen from the SOURCE definitions. This COG graphical output is written to the**ps** file. The user should look at these SOURCE pictures to assure that the source has been properly defined and positioned. The user can also supply an ANALYSIS Data Block to make a picture of SOURCE particles scattering in the user's geometry.

Example of a SOURCE POSITION graphical output showing a DISK surface source at -73.72 mm with a 5 mm radius (mm units set in BASIC Data Block).

```
DEFINE POSITION = 2
SS-DISK 0. -73.72 0. 0. -74.72 0. 5.
```

SOURCE--POSITION. Definition of description 2
 the unit of length is one millimeter
 SURFACE of DISK with center (.0000E+00,-7.3720E+01, .0000E+00)
 point on normal through center (.0000E+00,-7.4720E+01, .0000E+00)
 radius 5.0000E+00



9.5.2 POINT Source

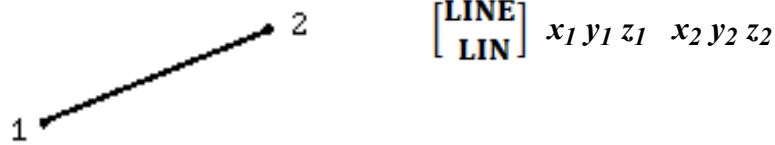
$$\mathbf{P}_1 \cdot \begin{bmatrix} \text{POINT} \\ \text{PT} \end{bmatrix} x \ y \ z$$

Point $\mathbf{P}_1(x,y,z)$ specifies the location of the point source.

Example: Define a *POINT* source at $(-1.3, 2.2, 0.)$.

```
DEFINE POSITION 1  
POINT -1.3 2.2 0. LINE SOURCES
```

Straight LINE Source

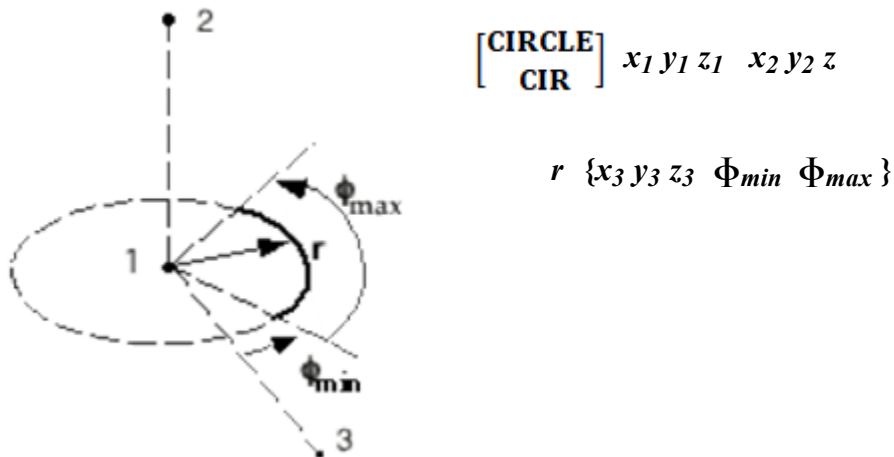


Points $P_1(x_1, y_1, z_1)$ and $P_2(x_2, y_2, z_2)$ form the endpoints of the line.

Example: Define a LINE source between (0.,0.,0.) and (0.,4.,0.).

```
DEFINE P 3
LINE 0. 0. 0. 0. 4. 0.
```

CIRCLE Line Source



Point $P_1(x_1, y_1, z_1)$ is the center of the circle and point $P_2(x_2, y_2, z_2)$ lies on the normal to the circle plane through P_1 .

r is the circle radius.

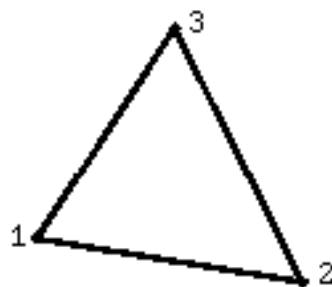
A circular arc source can be defined by specifying the point $P_3(x_3, y_3, z_3)$ and an azimuthal angle range ϕ_{min}, ϕ_{max} (in degrees). P_1P_3 forms a reference line from which the azimuth is measured. P_1P_3 must be normal to P_1P_2 .

Example: Define a CIRCLE line source centered at (0.,0.,0.), lying in the xz plane, and with a radius of 3.5.

```
DEFINE POSITION 2
CIRCLE 0. 0. 0. 0. 1. 0. 3.5
```

9.5.3 Surface Sources

SS-TRIANGLE Surface Source



$$\begin{bmatrix} \text{SS} - \text{TRIANGLE} \\ \text{SS} - \text{TRI} \\ \text{SS} - \text{T} \end{bmatrix} \begin{matrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 \\ & & & x_3 & y_3 & z_3 \end{matrix}$$

Points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$, and $P_3(x_3, y_3, z_3)$ lie at the three vertices of the triangle.

Example: Define a TRIANGLE surface source with vertices at $(0., 0., 0.)$, $(1., 0., 0.)$, and $(0., 1., 0.)$.

```
DEFINE P 1
SS-TRIANGLE 0. 0. 0. 1. 0. 0. 0. 1. 0.
```

SS-PARALLELOGRAM Surface Source



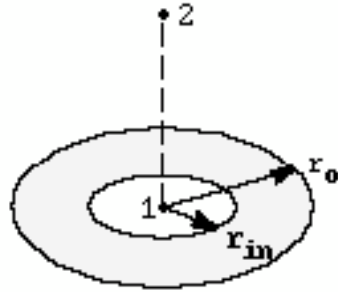
$$\begin{bmatrix} \text{SS} - \text{PARALLELOGRAM} \\ \text{SS} - \text{PAR} \\ \text{SS} - \text{P} \end{bmatrix} \begin{matrix} x_1 & y_1 & z_1 \\ & x_2 & y_2 & z_2 & x_3 & y_3 & z_3 \end{matrix}$$

Points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$, and $P_3(x_3, y_3, z_3)$ are the "top-left", "bottom-left", and "bottom-right" corners of the parallelogram, as shown.

Example: Define a PARALLELOGRAM surface source with corners at $(-1., 1., 0.)$, $(0., 0., 0.)$, and $(2., 0., 0.)$.

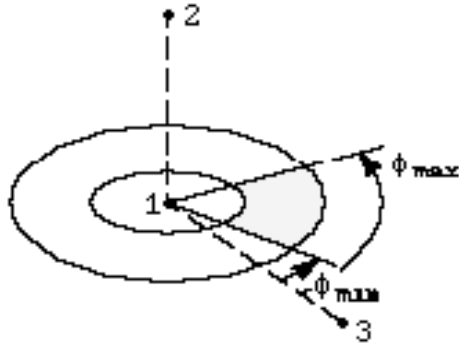
```
DEFINE POSITION 3
SS-PAR -1. 1. 0. 0. 0. 0. 2. 0. 0.
```

SS-DISK Surface Source



$$\begin{bmatrix} \text{SS} - \text{DISK} \\ \text{SS} - \text{DIS} \\ \text{SS} - \text{D} \end{bmatrix} \quad x_1 \ y_1 \ z_1 \quad x_2 \ y_2 \ z_2 \quad r_o$$

$$\{r_{in}\} \quad \{x_3 \ y_3 \ z_3 \quad \phi_{min} \ \phi_{max}\}$$



Point $\mathbf{P}_1(x_1, y_1, z_1)$ is the center of the disk and point $\mathbf{P}_2(x_2, y_2, z_2)$ lies on the disk normal through $\mathbf{P}_1$.

r_o is the outer radius.

r_{in} is the (optional) inner radius, which restricts the surface to an annulus.

The surface can be limited in azimuthal extent by specifying the point $\mathbf{P}_3(x_3, y_3, z_3)$ and

an azimuthal range ϕ_{min}, ϕ_{max} (in degrees). $\mathbf{P}_1\mathbf{P}_3$ forms a reference line from which the azimuth is measured. $\mathbf{P}_1\mathbf{P}_3$ must be normal to $\mathbf{P}_1\mathbf{P}_2$.

Example: Define an annular DISK surface source centered at (0.,0.,0.), lying in the xy plane, and with an outer radius of 3. and an inner radius of 1.5.

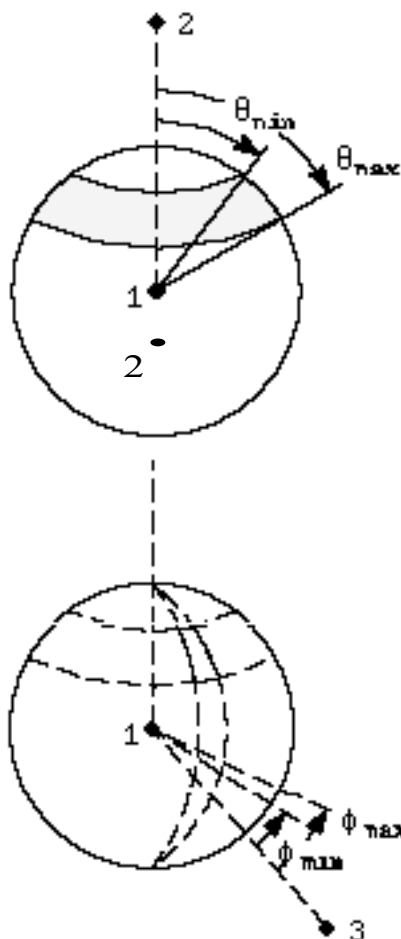
```
DEFINE POSITION 2
SS-DISK 0. 0. 0. 0. 0. 1. 3. 1.5
```

SS-SPHERE Surface Source

$$\begin{bmatrix} \text{SS} - \text{SPHERE} \\ \text{SS} - \text{SPH} \\ \text{SS} - \text{S} \end{bmatrix} \quad x_1 \ y_1 \ z_1 \quad r$$

$$\{x_2 \ y_2 \ z_2 \quad \theta_{min} \ \theta_{max} \}$$

$$\{x_3 \ y_3 \ z_3 \quad \phi_{min} \ \phi_{max} \}$$



Point $P_1(x_1, y_1, z_1)$ is the center of the sphere.

r is the radius.

The surface can be limited in polar angle by specifying a point $P_2(x_2, y_2, z_2)$ on the polar axis and the angle range $\theta_{min}, \theta_{max}$ (in degrees).

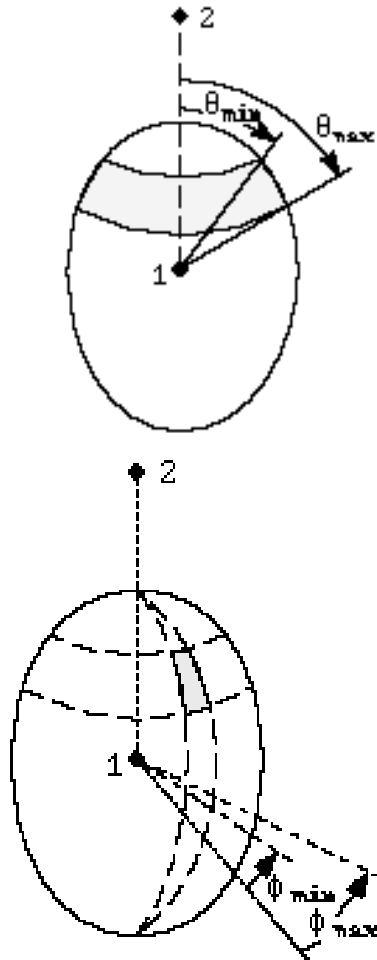
The surface can be further limited in azimuthal extent by specifying the point $P_3(x_3, y_3, z_3)$ and an azimuthal range ϕ_{min}, ϕ_{max} (in degrees).

P_1P_3 forms a reference line from which the azimuth is measured. P_1P_3 must be normal to P_1P_2 .

Example: Define a hemi-spherical SPHERE surface source centered at $(0, 0, 0)$, with a radius of 3., and lying in the $+x$ hemisphere.

```
DEFINE P 1
SS-SPHERE 0. 0. 0. 3. 1. 0. 0. 0. 90.
```

SS-ELLIPSOID Surface Source



[SS - ELLIP] [PRO] a b x₁ y₁ z₁
 [SS - ELL] [OBL]

x₂ y₂ z₂ {θ_{min} θ_{max}}
 {x₃ y₃ z₃ φ_{min} φ_{max}}

The ellipsoidal surface is formed by revolving an ellipse about either its major axis, yielding a prolate spheroid, or its minor axis, forming an oblate spheroid.

PRO denotes a prolate spheroid;

OBL denotes an oblate spheroid;

a is the major axis length;

b is the minor axis length;

Point **P₁(x₁,y₁,z₁)** is the center of the ellipsoid;

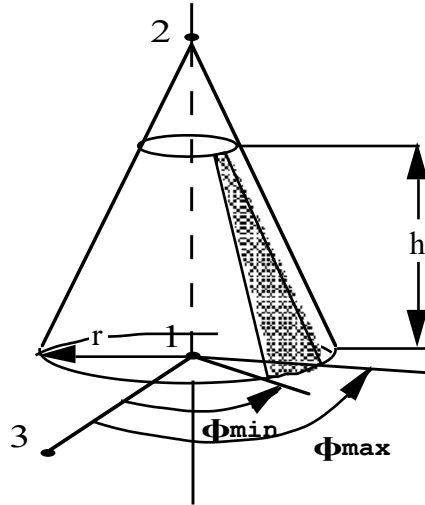
Point **P₂(x₂,y₂,z₂)** lies on the axis of revolution of the figure.

The surface can be limited in polar angle by specifying the polar angle range *θ_{min}*, *θ_{max}* (in degrees) as measured from the **P₁P₂** axis. The surface can be further limited in extent by specifying the point **P₃(x₃,y₃,z₃)** and an azimuthal angle range *φ_{min}*, *φ_{max}* (in degrees). **P₁P₃** forms a reference line from which the azimuth is measured. **P₁P₃** must be normal to **P₁P₂**.

Example: Define a prolate ELLIPSOID surface source with major axis 2. and minor axis 1., centered at (0.,0.,0.), rotated about a line from the center to (0.,0.,1.), and limited in polar extent between 30° and 60°.

```
DEFINE POSITION 1
SS-ELLIPS PRO 2. 1. 0. 0. 0. 0. 0. 1. 30. 60.
```

SS-CONE Surface Source



$$\begin{bmatrix} \text{SS} - \text{CONE} \\ \text{SS} - \text{CON} \\ \text{SS} - \text{K} \end{bmatrix} \begin{matrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & r \\ \{h\} & \{x_3 & y_3 & z_3 & \Phi_{\min} & \Phi_{\max}\} \end{matrix}$$

Point $P_1(x_1, y_1, z_1)$ lies on the cone axis in the base plane.

Point $P_2(x_2, y_2, z_2)$ defines the cone apex.

r is the cone radius in the base plane.

The surface can be limited in axial extent by specifying a height h , measured from the base.

The surface can be limited in azimuthal extent by specifying the point $P_3(x_3, y_3, z_3)$ and an azimuthal angle range $\Phi_{\min}, \Phi_{\max}$ (in degrees).

P_1P_3 forms a reference line from which the azimuth is measured.

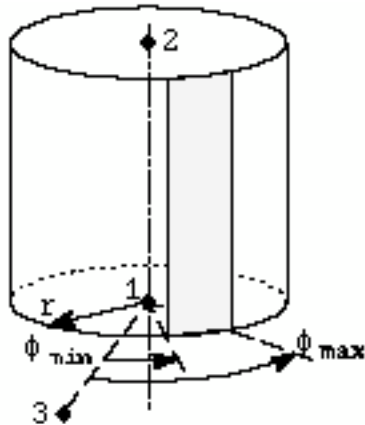
P_1P_3 must be normal to P_1P_2 .

Note: The surface of the cone does not include the base or top planes.

Example: Define a CONE surface source with base center at (0.,0.,0.), apex at (0.,0.,3.), base radius of 1., height of 2., and limited in extent between 0° to 90°, measured from the line formed by the base center and the point (1.,0.,0.).

```
DEFINE POSITION 1
SS-CONE 0. 0. 0. 0. 0. 3. 1 2. 1. 0. 0. 0. 90.
```

SS-CYLINDER Surface Source



$$\begin{array}{l}
 \text{[SS - CYLINDER]} \\
 \text{SS - CYL} \\
 \text{SS - C}
 \end{array}
 \begin{array}{l}
 x_1 \ y_1 \ z_1 \ x_2 \ y_2 \ z_2 \ r \\
 \{x_3 \ y_3 \ z_3 \ \phi_{min} \ \phi_{max}\}
 \end{array}$$

Points $\mathbf{P}_1(x_1, y_1, z_1)$ and $\mathbf{P}_2(x_2, y_2, z_2)$ define the cylinder axis and its extent. The surface is limited by the axis-normal end planes containing $\mathbf{P}_1$ and $\mathbf{P}_2$.

r is cylinder radius.

The surface can be limited in azimuthal extent by specifying the point $\mathbf{P}_3(x_3, y_3, z_3)$ and an angle range ϕ_{min}, ϕ_{max} (in degrees). $\mathbf{P}_1\mathbf{P}_3$ forms a reference line from which the azimuth is measured. $\mathbf{P}_1\mathbf{P}_3$ must be normal to $\mathbf{P}_1\mathbf{P}_2$.

Note: the surface of the cylinder does not include the limiting end planes.

Example: Define a CYLINDER surface source with base center at (0.,0.,0.), top center at (0.,0.,2.), radius of 1., and limited in extent between 30° to 60°, measured from the line formed by the base center and the point (0.,1.,0.).

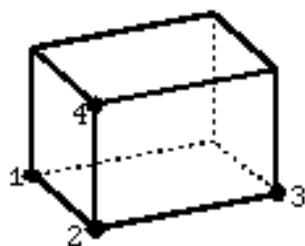
```

DEFINE POSITION 1
SS-CYLINDER 0. 0. 0. 0. 0. 2. 1.
0. 1. 0. 30. 60.

```

9.5.4 Volume Sources

PARALLELEPIPED, BOX Volume Source



$$\begin{array}{c}
 \left[\begin{array}{c} \text{PARALLEL} \\ \text{PAR} \\ \text{BOX} \end{array} \right] \quad \begin{array}{cc} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 \\ & & & x_3 & y_3 & z_3 & x_4 & y_4 & z_4 \end{array} \\
 \left\{ \left[\begin{array}{c} \text{MATERIAL} \\ \text{MAT} \\ \text{REGION} \\ \text{REG} \end{array} \right] = \text{number} \right\}
 \end{array}$$

$P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$, $P_3(x_3, y_3, z_3)$, $P_4(x_4, y_4, z_4)$ represent four corners of the parallelepiped.

Points P_1 , P_2 , and P_3 are three corners of one plane. Point P_4 lies on the remaining corner whose edge connects to P_2 (see figure).

The optional MATERIAL/REGION field specifies either a material or a region number (*mat-ID#* or *reg-ID#*). This constrains the source points to be chosen within the defined volume and also within the specified material or region.

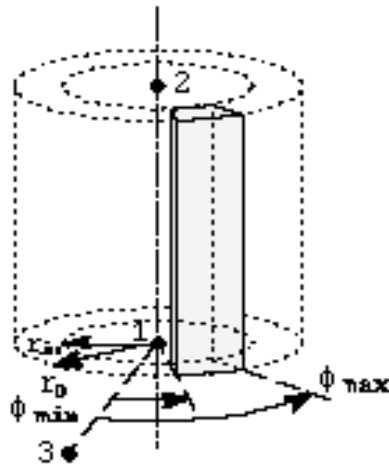
Example: Define a PARALLELEPIPED volume source with four defining corners at $(0., 0., 0.)$, $(3., 0., 0.)$, $(3., 1., 0.)$, and $(3., 0., 1.)$.

```

DEFINE POSITION 1
PARALLEL 0. 0. 0. 3. 0. 0. 3. 1. 0. 3. 0. 1.

```

CYLINDER Volume Source



$$\begin{bmatrix} \text{CYLINDER} \\ \text{CYL} \\ \text{C} \end{bmatrix} \begin{matrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & r_o \\ & & & & & & \{r_{in}\} & (x_3 & y_3 & z_3 & \phi_{min} & \phi_{max}) \end{matrix}$$

$$\left\{ \begin{bmatrix} \text{MATERIAL} \\ \text{MAT} \\ \text{REGION} \\ \text{REG} \end{bmatrix} = \text{number} \right\}$$

Points $\mathbf{P}_1(x_1, y_1, z_1)$ and $\mathbf{P}_2(x_2, y_2, z_2)$ define the cylinder axis and its extent. The volume is limited by axis-normal end planes at $\mathbf{P}_1$ and $\mathbf{P}_2$.

r_o is the cylinder outer radius.

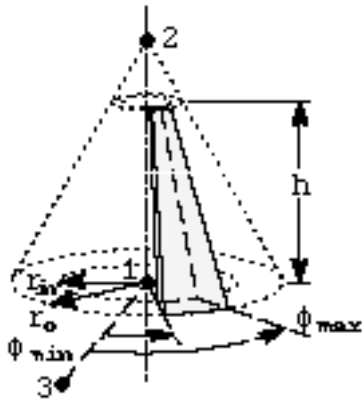
The volume can be limited in radial extent by specifying an inner radius r_{in} . The volume can be limited in azimuthal extent by specifying the point $\mathbf{P}_3(x_3, y_3, z_3)$ and an angle range ϕ_{min}, ϕ_{max} (in degrees). $\mathbf{P}_1\mathbf{P}_3$ forms a reference line from which the azimuth is measured. $\mathbf{P}_1\mathbf{P}_3$ must be normal to $\mathbf{P}_1\mathbf{P}_2$.

The optional MATERIAL/REGION field specifies either a material or a region number (*mat-ID#* or *reg-ID#*). This constrains the source points to be chosen within the defined volume and also within the specified material or region.

Example: Define a CYLINDER volume source with base centered at (0.,0.,0.), top centered at (0.,0.,2.), radius of 1.2, and limited in extent between 0° to 90°, measured from the line formed by the base center and the point (1.,0.,0.).

```
DEFINE POSITION 1
CYLINDER 0. 0. 0. 0. 0. 2. 1.2 1. 0. 0. 0. 90.
```

CONE Volume Source



CONE Volume Source

$$\begin{aligned}
 & \begin{bmatrix} \text{CONE} \\ \text{CON} \\ \text{K} \end{bmatrix} \quad x_1 \ y_1 \ z_1 \quad x_2 \ y_2 \ z_2 \quad r_o \\
 & \{r_{in}\} \quad \left\{ \begin{bmatrix} \text{HEIGHT} \\ \text{H} \end{bmatrix} = h \right\} \\
 & \{x_3 \ y_3 \ z_3 \quad \phi_{min} \ \phi_{max}\} \\
 & \left\{ \begin{bmatrix} \text{MATERIAL} \\ \text{MAT} \\ \text{REGION} \\ \text{REG} \end{bmatrix} = \text{number} \right\}
 \end{aligned}$$

Point $\mathbf{P}_1(x_1, y_1, z_1)$ lies on the cone axis and in the base plane.

Point $\mathbf{P}_2(x_2, y_2, z_2)$ defines the cone apex.

r_o is the cone outer radius in the base plane.

The volume can be limited to a conical shell by specifying r_{in} , the cone inner radius in the base plane. The resulting volume lies between two parallel cones which have the same centerline but different radii in the base plane.

The volume can be limited in axial extent by specifying a height h , measured from the base.

The volume can be limited in azimuthal extent by specifying the point $\mathbf{P}_3(x_3, y_3, z_3)$ and an azimuthal angle range ϕ_{min} , ϕ_{max} (in degrees). $\mathbf{P}_1\mathbf{P}_3$ forms a reference line from which the azimuth is measured. $\mathbf{P}_1\mathbf{P}_3$ must be normal to $\mathbf{P}_1\mathbf{P}_2$.

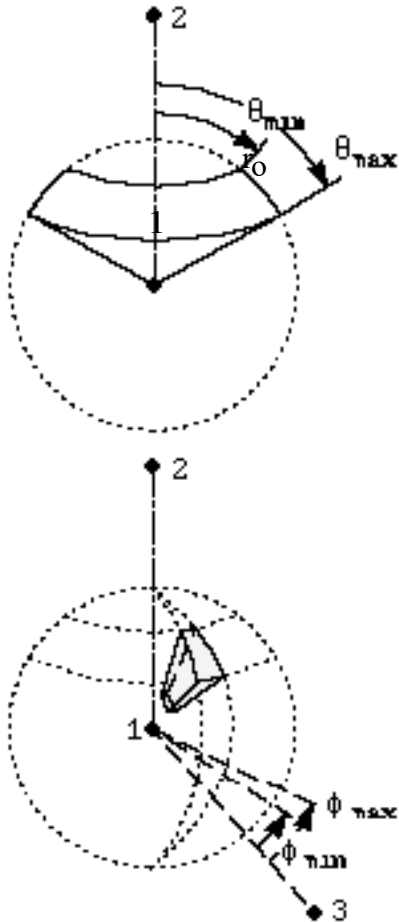
The optional MATERIAL/REGION field specifies either a material or a region number (*mat-ID#* or *reg-ID#*). This constrains the source points to be chosen within the defined volume and also within the specified material or region.

Example: Define a CONE volume source with base centered at (0.,0.,0.), apex at (0.,0.,3.), outer radius of 2., inner radius 0., height of 1.5, and limited in extent between 0° to 180°, measured from the line formed by the base center and the point (2.,1.,0.).

```
DEFINE POSITION 1
```

```
CONE 0. 0. 0. 0. 0. 3. 2. 0. 1.5 2. 1. 0. 0. 180.
```

SPHERE Volume Source



$$\begin{bmatrix} \text{SPHERE} \\ \text{SPH} \\ \text{S} \end{bmatrix} \quad x_1 \ y_1 \ z_1 \ r_o \ \{r_{in}\}$$

$$\{x_2 \ y_2 \ z_2 \ \theta_{min} \ \theta_{max}\}$$

$$\{x_3 \ y_3 \ z_3 \ \phi_{min} \ \phi_{max}\}$$

$$\left\{ \begin{bmatrix} \text{MATERIAL} \\ \text{MAT} \\ \text{REGION} \\ \text{REG} \end{bmatrix} = \text{number} \right\}$$

Point $P_1(x_1, y_1, z_1)$ lies at the sphere center, r_o is the sphere outer radius. The sphere may be restricted to a spherical shell by specifying the inner radius, r_{in} .

The volume can be limited in polar angle by specifying the point $P_2(x_2, y_2, z_2)$ and an polar angle range $\theta_{min}, \theta_{max}$ (in degrees).

The volume can be limited in azimuthal extent by specifying the point $P_3(x_3, y_3, z_3)$ and an azimuthal angle range

ϕ_{min}, ϕ_{max} (in degrees). P_1P_3 forms a reference line from which the azimuth is measured. P_1P_3 must be normal to P_1P_2 .

The optional MATERIAL/REGION field specifies a material or a region number (*mat-ID#* or *reg-ID#*). This constrains the SOURCE points to be chosen within the defined volume and also within the specified material or region.

Example: Define a SPHERE volume source with center at (0.,9.,0.), outer radius of 5., inner radius 1., and limited in extent between polar angles 45° to 90°, measured from the line formed by the center and the point (1.,1.,0.).

```
DEFINE P 4
SPHERE 0. 9. 0.    5.  1.    1. 1. 0.  45. 90.
```

Arbitrary Volume Source

An arbitrary volume source can be specified by utilizing the MATERIAL/REGION option in combination with one of the Volume Sources described above.

Example: The following SPHERE Volume Source with the REGION Option restricts source points to the volume which lies both within the sphere and within the user's REGION 3.

```
DEFINE P 4
SPHERE 0. 9. 0. 5. REGION = 3
```

Since a REGION is a user-defined set of one or more sectors, and sectors may have any desired complex shape, this method allows the construction of a very general source volume.

Each Volume Source may be combined with the MATERIAL/REGION Option. The data for this Option immediately follow the Volume Source specification, and has this form

$$\left\{ \begin{array}{c} \text{MATERIAL} \\ \text{MAT} \\ \text{REGION} \\ \text{REG} \end{array} \right\} = \text{number}$$

(See above-listed Volume Source(s) for details of specifying these sources.)

The optional field specifies either a material or a region number (*mat-ID#* or *reg-ID#*). This constrains the source points to be chosen within the defined volume and also within the specified material or region.

COG will choose source points randomly and uniformly inside the specified Volume Source volume, and then discard those which lie outside the specified MATERIAL or REGION. For efficient point generation, the Volume Source should be roughly the same size and shape as the REGION.

9.6 Source ENERGY Dependence

The definition of an ENERGY-dependent SOURCE specification is started with:

$$\text{DEFINE } \begin{bmatrix} \text{ENERGY} \\ \text{E} \end{bmatrix} = ed\# \begin{bmatrix} \text{NEUTRON} \\ \text{PHOTON} \\ \text{GAMMA} \\ \text{PROTON} \\ \text{ELECTRON} \end{bmatrix}$$

where:

ed# is any positive integer chosen by the user to identify this ENERGY definition;
and

$\begin{bmatrix} \text{NEUTRON} \\ \text{PHOTON} \\ \text{GAMMA} \\ \text{PROTON} \\ \text{ELECTRON} \end{bmatrix}$ specifies the source particle type. Each ENERGY definition is

limited to only one type of particle. Include DEUTERON and ALPHA in the particle type list.

The DEFINE statement is then followed by the data for one of the six types of ENERGY specifications given below. You will need a DEFINE ENERGY definition for each different energy-dependence in your problem.

Default: None; an ENERGY-dependent source specification must be given in each COG problem.

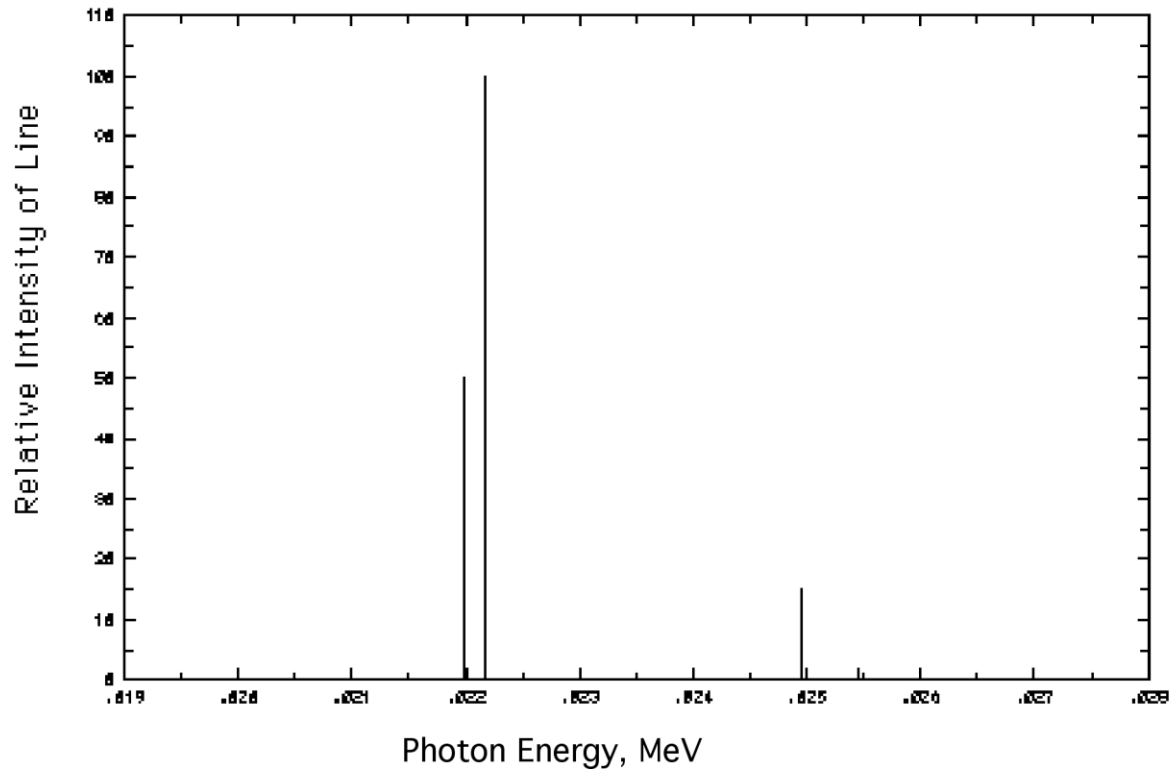
The units used to describe the ENERGY dependence are defaulted to MeV, unless specified otherwise in the BASIC Data Block.

9.6.1 Source ENERGY Graphics

As for the POSITION-dependent source specification, COG makes a plot of the specified ENERGY-dependent source. In some cases, on the top portion of this figure are plotted a set of randomly chosen energies picked from the ENERGY-dependent source distribution. Note that the chosen points include any IMPORTANCE specification.

SOURCE--ENERGY: Definition of description 2
the unit of energy is mev
photon source
LINE

energy	intensity
2.216E-02	1.000E+02
2.199E-02	5.000E+01
2.494E-02	1.500E+01
2.545E-02	2.000E+00



9.6.2 LINE Energy Source

Many sources consist of one or more discrete energy lines. Each line has associated with it a relative intensity. A LINE source is given by

$$\begin{array}{l} \left[\begin{array}{l} \text{LINE} \\ \text{LINES} \end{array} \right] e_1 s_1 (e_2 s_2) \dots \\ \left\{ \left[\begin{array}{l} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\} \end{array}$$

where s_1 is the relative **strength** (intensity) of the line at energy e_1 etc.

The optional **importance-spec** is a distribution which specifies the relative IMPORTANCE of the source over a set of energies. It has two forms:

$$\begin{array}{ll} e_1 i_1 & e_2 i_2 & e_3 i_3 & \dots & e_n i_n & \text{(Linear Dist.)} \\ \text{or } e_1 i_1 & e_2 i_2 & e_3 i_3 & \dots & e_n i_n & e_{n+1} & \text{(Histogram Dist.)} \end{array}$$

In the first form, **Linearly-interpolated Distribution**, IMPORTANCE is given as i_1 at energy e_1 and varies **linearly** to i_2 at energy e_2 , and reaches a value of i_n at energy e_n .

In the second form, **Histogram Distribution**, the IMPORTANCE is constant at a value of i_1 between energies e_1 and e_2 , at i_2 between e_2 and e_3 , etc., and at i_n between e_n and e_{n+1} . Note that the Histogram form requires one more value of e than the Linearly-interpolated form.

See **Source IMPORTANCES** for more discussion on the purpose and use of IMPORTANCES.

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a LINE ENERGY-dependent source corresponding to the fluorescent silver x-ray lines. Sample the kb2 line 50 times as frequently as its natural occurrence.

```
DEFINE ENERGY = 2 PHOTON
LINE
    .022162      100  $AG KA1
    .021988       50  $KA2
    .024942       15  $KB1
    .025452        2  $KB2
IMP
    .022162        1  $AG KA1
    .021988        1  $KA2
    .024942        1  $KB1
    .025452       50  $KB2
```

9.6.3 CASCADE Line Energy Source

COG can simulate simple gamma cascades. This is specified by:

particle-type CASCADE e_1 e_2 w_2 a_2 a_4

where:

particle-type must be **PHOTON** for this option;

e_1 is the energy of the first member of the CASCADE;

e_2 is the energy of the second member;

w_2 is the ratio of the relative intensities of gamma2 to gamma1;

a_2 and a_4 are coefficients which describe the angular correlation between gamma1 and gamma2 in the equation:

$$1 + a_2 * P_2(w) + a_4 * P_4(w),$$

where P_2 and P_4 are Legendre polynomials, and w is the cosine of the angle between gamma1 and gamma2.

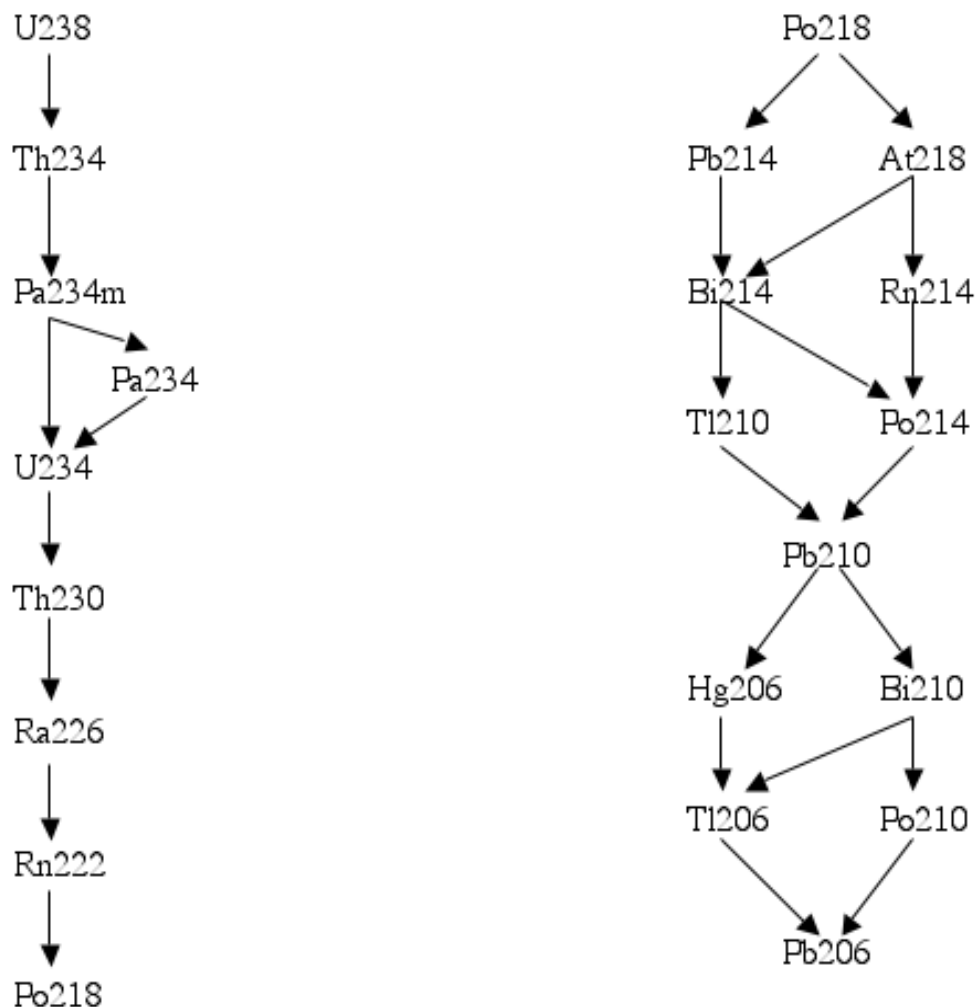
Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a CASCADE source corresponding to gamma emission from Co^{60} .

```
DEFINE E 1
PHOTON CASCADE 1.173 1.332 1. 0.102 0.009
```

9.6.4 RADSRC Energy Source

Many applications require the calculation of an ideal theoretical radiation spectrum resulting from the natural decay of radioactive elements. The **RADSRC** option is designed to fill that need. The figure below shows decay paths for U238 and its daughter isotopes. There are 30 possible paths from U238 to Pb206. COG computes the concentrations of decay products given an initial concentration and age, and photon radiation due to the continuing decay of those products.



COG can simulate a spectrum from a radiation source. This is specified by:

particle-type RADSRC *iso1 f1 iso2 f2 ... AGE yrs {AMPLIMIT amp}*
{ELO elo} {EHI ehi}

where:

particle-type must be **PHOTON** or **GAMMA** for this option;

iso1 is the first component of the radioactive material;

f1 is the fraction of first component in the radioactive material;

iso2 is the second component of the radioactive material;

f2 is the fraction of second component in the radioactive material;

... can have up to 10 component-fraction pairs;

AGE is a keyword ending the component-fraction pairs and preceding the age;

yrs is the age in years at which the radiation spectrum is to be simulated;

AMPLIMIT amp ignores lines with amplitudes less than *amp* times the total amplitude;

ELO elo ignores lines with energies less than *elo*;

EHI ehi ignores lines with energies greater than *ehi*;

Example: Define a RADSRC source for an originating source comprised of 95% U238 and 5% U235 at age 10 years.

```
DEFINE E 1  
PHOTON RADSRC U238 0.95 U235 0.05 AGE 10.
```

NOTE: The fractions should sum to 1 and represent 1 gm of material.

Default: Energies are MeV unless otherwise specified in the BASIC Data Block.

9.6.5 GAUSSIAN Energy Source

A GAUSSIAN energy distribution is given by

$$\text{GAUSSIAN } e_0 \ w \ \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where e_0 is the center energy and w is the full width of the distribution measured at half the maximum value. Internal to the code, this distribution is simulated by a 201 point distribution evaluated from a lower energy of

$e_0 - 2w$ or zero, whichever is greater, to $e_0 + 2w$.

(See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description.)

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a GAUSSIAN source of neutrons with mean energy of 14.1 MeV and full width at half maximum of 2.2 MeV.

```
DEFINE E 1 NEUTRON
      GAUSSIAN 14.1 2.2
```

9.6.6 WATT Fission Neutron Energy Source

A WATT fission neutron energy distribution is given by

$$\text{WATT } a \quad b \quad \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where a and b are the constants in the Watt expression for the fission neutron spectrum:

$$f(E) = \exp(-aE) \sinh \sqrt{bE}$$

For many fissionable isotopes, approximate values are $a=1$, $b=2$.

(See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description.)

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a WATT source of neutrons.

```
DEFINE E 1 NEUTRON
WATT 1. 2.
```


9.6.7 MAXWELLIan Energy Source

A Maxwellian energy distribution is given by

$$\text{MAXWELL } a \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where a is the parameter specifying the Maxwellian distribution:

$$f(E) = \sqrt{E} * \exp(-E/a) .$$

The code will generate 101 points of this distribution, ranging from $E = 0$ to $E = 8a$.

(See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description.)

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a Maxwellian source of neutrons.

```
DEFINE E 1 NEUTRON
MAXWELL 1.2
```

9.6.8 ONE/E Energy Source

A one/e energy distribution is given by

$$\text{ONE/E} \quad e1 \ e2 \quad \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where *e1* and *e2* are the energy bounds.

$$f(E)=1/E$$

The code will generate 101 points of this distribution, ranging from $E = e1$ to $E = e2$

(See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description.)

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.

Example: Define a 1/e source of neutrons between 0.01 and 2. MeV.

```
DEFINE E 1 NEUTRON
ON/E 0.01 2.
```

9.6.9 Point-Wise Energy DISTRIBUTION

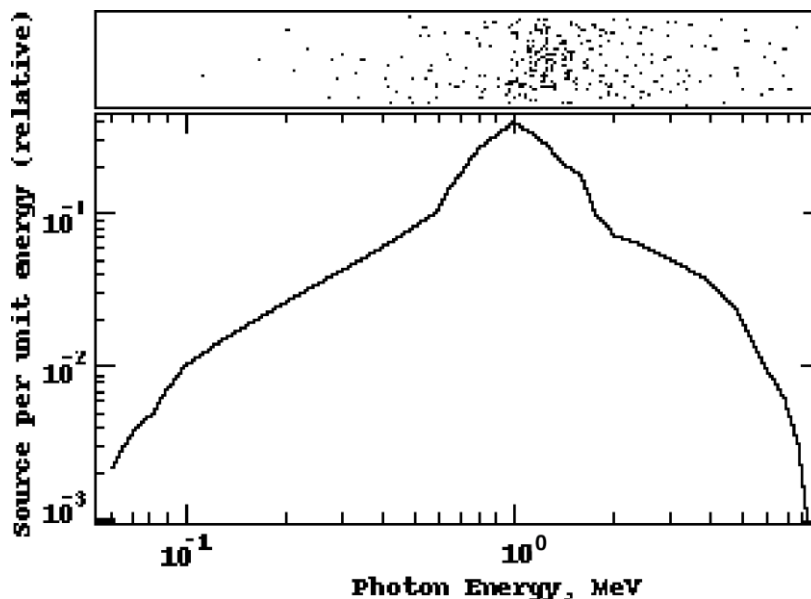
A complete Point-Wise ENERGY DISTRIBUTION may be given by specifying

$$\text{DISTRIBUTION } e_1 \ s_1 \ e_2 \ s_2 \ e_3 \ s_3 \ \dots$$

$$\left\{ \begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right\} \text{importance} - \text{spec} \}$$

where the s_i values represent the relative source intensity *per unit energy* at the preceding energy point. It is assumed that a linear fit between adjacent energy points accurately describes the distribution. The energies may be in either ascending or descending order. Outside of the specified energy range, the source is zero. An example of the plot of such an input is shown below. (See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description).

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.



***Example:** Define a neutron point-wise energy DISTRIBUTION with specified IMPORTANCEs. The number of neutrons per unit energy is 1 at 2 MeV, 2 at 8 MeV, 3 at 14 MeV, and zero everywhere else. The importance is unity at 0 MeV, and decreases to zero at 20 MeV.*

```
DEFINE ENERGY 2 NEUTRON
DISTRIBUTION 2. 1. 8. 2. 14. 3.
IMPORTANCE 0. 1. 20. 0.
```

9.6.10 BIN Energy Source

A BIN source known only by averages within a given energy bin (i.e. a histogram) may be specified by

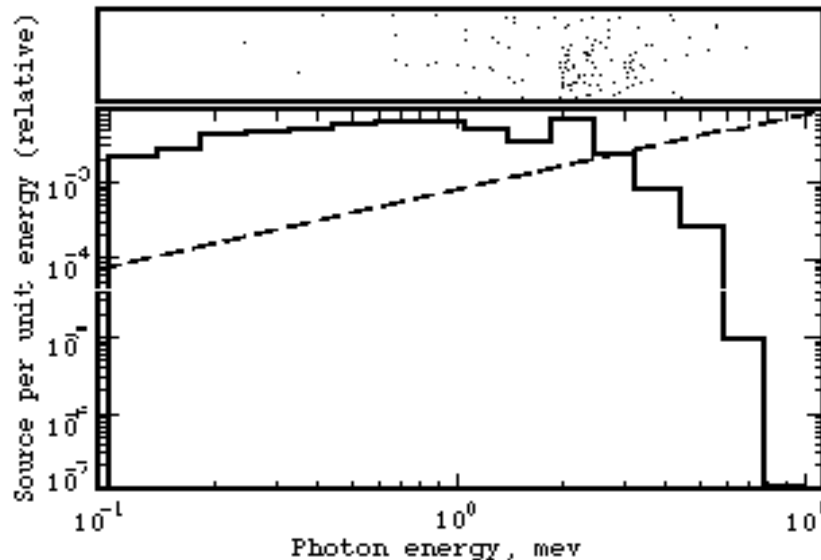
$$\text{BIN } e_1 \ s_1 \ e_2 \ s_2 \ e_3 \ s_3 \ \dots \ e_n \ s_n \ e_{n+1} \\ \left\{ \begin{array}{c} \text{[IMPORTANCE]} \\ \text{IMP} \\ \text{I} \end{array} \right\} \text{importance - spec} \}$$

where s_i represents the average source intensity *per unit energy* between energies e_i and e_{i+1} . The energies may be in either ascending or descending order. Note the presence of the final extra energy value, e_{n+1} . COG first chooses an energy group and then chooses a specific energy by uniformly sampling between the bin boundaries. Outside of the specified bins, the source is zero.

The plot shown below is an energy graphic of BIN data. IMPORTANCE is drawn as a dotted line; the plotted points are the source samples chosen from the IMPORTANCE-weighted energy spectrum.

(See Source **LINE ENERGY** Specification for ENERGY *importance-spec* description).

Default: Energy units are MeV unless otherwise specified in the BASIC Data Block.



***Example** of a neutron BIN source. The average number of neutrons per unit energy is 1 between 2 and 8 MeV, 2 between 8 and 14 MeV, and 0 everywhere else.*

```
DEFINE ENERGY 1  
NEUTRON BIN 2. 1. 8. 2. 14.
```

9.6.11 Spontaneous Fission Energy Source (SFS)

Energy source from spontaneous fission.

particle-type SFS za

where *particle-type* is the particle type, neutron for the SFS option, **SFS** is the key word for the Spontaneous Fission Source, and **za** is the ZA identifier for the fissioning isotope.

The first source call invokes the FREYA fission event which generates an array of energies and directions for the possible fission neutrons. The last member of the array is taken as the source neutron and the size of the array is decreased by one. Succeeding source calls repeat this procedure until the array is empty, then another call to the FREYA fission event restarts the process until the desired number of source particles have been run. This source is similar to the FissSrc below in that the FREYA fission event is used to generate an array of fission particles, in this case only neutrons are generated and they are processed individually and are thus uncorrelated.

Example: Define a Cf-252 spontaneous fission source of neutrons.

```
neutron sfs 98252
```

Note: Only available with U-238, Pu-238, Pu-240, Cm-244, and Cf-252 in the FREYA mode.

Note: The source intensity for SFS specification is in the Source INCREMENTS Section. See Source Examples Section Example 4.

9.6.12 FissSrc: A New Source Energy Option

The **FissSrc** option is defined as....

particle-type FissSrc za

where *particle-type* may be neutron, photon or neut-pho. **FissSrc** is the key word for the full Fission Source, and **za** is ZA identifier the fissioning isotope.

Each source call invokes the FREYA fission event which generates an array of energies and directions for the possible fission neutrons and photons. When the particle-type is **neutron**, only the neutrons are kept in the array. When the particle-type is **photon** (or **gamma**), only the photons are kept in the array. When the particle-type is **neut-pho** (or **neut-gam**), both the neutrons and photons are kept in the array. The first member of the array is taken as a

source particle, the remaining members of the array are placed in the secondary particle bank. After the initial source particle and all its secondary particles are processed, then the remaining source particles are taken from the secondary bank and processed, one after the other until the all the original members of the fission array have been processed. This is repeated until the desired numbers of fission source events have been run. This source is similar to the SFS above in that the FREYA fission event is used to generate an array of fission particles, in this case both neutrons and photons may be generated and they are processed as a batch and are thus correlated in time and space.

***Example:** Define a Cf-252 full Fission Source.*

neut-pho fisssrc 98252

***Note:** The source intensity for **FissSrc** specification is in the Source INCREMENTS Section. See Source Examples Section Example 4.*

9.7 Source TIME Dependence

The definition of a TIME-dependent source specification is started with:

DEFINE $\begin{bmatrix} \text{TIME} \\ \text{T} \end{bmatrix} = td\#$

where:

td# is any positive integer chosen by the user to identify this TIME definition.

Unlike the POSITION-dependent source and the ENERGY-dependent source, a TIME-dependent source DEFINITION is *optional*.

You will need a DEFINE TIME definition for each different TIME-dependence in your problem.

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Default Dependence: If not specified, the TIME dependence will be a delta function at time = 0.

Note that negative times are perfectly legal, since the zero point for time is entirely arbitrary.

Time-Independent Sources: The question sometimes arises as to the proper specification of a source for a time-independent problem. Because COG is a time-dependent transport code, each particle has a time associated with every event in its transport history, from particle birth at the source, to particle scoring at the detector. However, you can choose to suppress the time dimension in the problem output if you wish.

If you are just interested in computing the total probability that a source particle will cause a score in your detector, independent of time, then you can omit the time-dependence from your INCREMENT statements. COG will use the default delta function at time = 0 for all source particles. All detector results will be normalized to results *per source particle* emitted.

If your problem is time-independent because your sources are steady, then you might want to use the STEADY STATE option. All detector results will be normalized to results *per unit time*.

9.7.1 Source TIME Graphics

As for the POSITION-dependent source specification, COG makes a plot of the specified TIME-dependent source. Note that the chosen points include any IMPORTANCE specification.

9.7.2 DELTA Function Time Source

A delta function time distribution causes all particles to be born at one specified time. The input specification is

DELTA t_0

where t_0 is the time of "birth" for all particles.

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a TIME-dependent source corresponding to a DELTA FUNCTION at -1.2×10^{-8} seconds.

```
DEFINE TIME 1  
DELTA -1.2E-8
```

9.7.3 PULSE Time Source

For the case where the TIME-dependent source can be represented as a series of one or more pulses, use:

$$\text{PULSE } t_1 t_2 \dots t_n s_1 s_2 \dots s_n \left\{ \begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right\} \text{importance - spec}$$

where a pulse of relative intensity s_1 occurs at t_1 , etc.

Note that all the time values occur first followed by their respective relative intensities. (This allows the use of the shortcut 'repeat' or 'interpolate' options). The IMPORTANCE, if any, is entered as usual.

The optional *importance-spec* is a distribution which specifies the relative importance of the source over a set of times. It has two forms:

$$t_1 i_1 \quad t_2 i_2 \quad t_3 i_3 \quad \dots \quad t_n i_n$$

or $t_1 i_1 \quad t_2 i_2 \quad t_3 i_3 \quad \dots \quad t_n i_n \quad t_{n+1}$

In the first form, *Linearly-interpolated Distribution*, importance is given as i_1 at time t_1 and varies *linearly* to i_2 at time t_2 , etc., and reaches i_n at time t_n .

In the second form, *Histogram Distribution*, the importance is constant at a value of i_1 between energies t_1 and t_2 , at i_2 between t_2 and t_3 , and at i_n between t_n and t_{n+1} .

Note that the Histogram form requires one more value of t than the Linearly-interpolated form.

See: Source IMPORTANCES for more discussion on the purpose and use of IMPORTANCES.

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a TIME-dependent PULSE SOURCE corresponding to 100 equal pulses starting at 0 seconds and ending at 1 second. In this example, we make use of the interpolation notation and the repeat notation. See *Input Notation Shortcuts* in the *Introduction*.

```
DEFINE TIME 3
PULSE 100 [0. I 1.] 100 [1.]
```

9.7.4 GAUSSIAN Time Source

A GAUSSIAN time distribution can be simulated by specifying:

$$\text{GAUSSIAN } t_0 \ w \ \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where t_0 is the time of maximum source strength and w is the full width at half the maximum amplitude. This is simulated by the code by internally generating a 201 point distribution extending in time from $t_0 - 2w$ to $t_0 + 2w$.

(See Source TIME-PULSE specification for TIME *importance-spec* description).

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a GAUSSIAN TIME-dependent source centered at 0 with a FWHM of 7×10^{-9} seconds.

```
DEFINE T 2
GAUSSIAN 0.      7.0E-9
```

9.7.5 Point-Wise Time DISTRIBUTION

A complete point-wise TIME DISTRIBUTION may be given by specifying

DISTRIBUTION $t_1 s_1 t_2 s_2 t_3 s_3 \dots$

$$\left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance - spec} \right\}$$

where the s values represent the relative source *per unit time*. It is assumed that a linear fit between adjacent time points accurately describes the distribution. The times may be in either ascending or descending order. Outside of the specified time range, the source is zero.

(See Source TIME-PULSE specification for TIME *importance-spec* description).

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a point-wise TIME-dependent source DISTRIBUTION which is 0 below -1×10^{-8} , rises linearly to 1 at 0, and decreases linearly back to 0 at 1×10^{-8} .

```
DEFINE TIME 1
DISTRIBUTION -1.E-8 0. 0. 1. 1.E-8 0.
```

9.7.6 BIN Time Source

A BIN TIME source known only by averages within a given time bin (i.e. a histogram) may be specified by

$$\text{BIN } t_1 s_1 \quad t_2 s_2 \quad t_3 s_3 \quad \dots \quad t_n s_n \quad t_{n+1} \quad \left\{ \begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right\} \text{importance - spec} \}$$

where the s values are the *average* source *per unit time* within each time bin. The times may be either in ascending or descending order. Note the presence of the final extra time value, t_{n+1} . COG first chooses a time bin and then chooses a specific time by uniformly sampling between the bin boundaries. Outside of the specified bins, the source is zero.

(See Source TIME-PULSE specification for TIME *importance-spec* description).

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a BIN TIME-dependent source distribution which is 0 below -5×10^{-9} , 1 between -5×10^{-9} and 5×10^{-9} , and 0 above 5×10^{-9} .

```
DEFINE TIME 3
BIN -5.E-9 1. 5E-9
```

9.7.7 STEADY State Time Source

A steady-state source, one with a constant source strength per unit time over all time, is specified by:

STEADY

Note: If STEADY is chosen for *any* source INCREMENT, then it must be used for *all* source INCREMENTS.

Note: STEADY causes detector results to be reported in terms of results per *unit time*, rather than in terms of *results per source particle*, which is the default calculation.

Default Units: Time units are seconds unless specified otherwise in the BASIC Data Block.

Example: Define a STEADY-state TIME-dependent source:

```
DEFINE TIME 3  
STEADY
```


9.8 Source ANGLE Dependence

The ANGLE dependence of the source determines the emission directions of the source particles. The ANGLE-dependent source specifies a **direction** (vector), whereas the POSITION-dependent source specifies a source **location** in three-space.

The ANGLE dependence of a source means the relative number of particles to be emitted, as a function of the angle between the **emission** direction and user-specified **reference** direction(s).

The **polar reference** direction is given in terms of its **direction cosines**:

$$(u_{pr}, v_{pr}, w_{pr})$$

where:

u_{pr} = Cosine of the angle between a vector parallel to the **+x direction** and the **polar reference** direction;

v_{pr} = Cosine of the angle between a vector parallel to the **+y direction** and the **polar reference** direction; and

w_{pr} = Cosine of the angle between a vector parallel to the **+z direction** and the **polar reference** direction.

The direction cosines satisfy the normalization condition:

$$u_{pr}^2 + v_{pr}^2 + w_{pr}^2 = 1$$

The **polar** angle θ between the **emission** direction of a source particle (given by the particle's direction cosines: (u, v, w)) and the **polar reference** direction is then:

$$\mu = \cos(\theta) = u * u_{pr} + v * v_{pr} + w * w_{pr}$$

In the typical (and simplest) case, the **polar** angle completely characterizes the ANGLE-dependent SOURCE. For more complex cases, it may be necessary to also specify an **azimuthal** ANGLE dependence in the plane normal to the **polar reference** direction.

The **azimuthal reference** direction is given, as for the **polar reference** direction, in terms of its **direction cosines**: (u_{ar}, v_{ar}, w_{ar}) .

The DEFINITION of an ANGLE-dependent SOURCE specification is started with:

$$\text{DEFINE } \begin{bmatrix} \text{ANGLE} \\ \text{A} \end{bmatrix} = \text{ad\#} \left[\begin{array}{l} u_{pr} \ v_{pr} \ w_{pr} \ \{u_{ar} \ v_{ar} \ w_{ar}\} \\ \text{NORMAL} \ \{u_{ar} \ v_{ar} \ w_{ar}\} \\ \text{POINT } x \ y \ z \\ \text{(none)} \end{array} \right]$$

where:

ad# is any positive integer chosen by the user to identify this definition;

The *reference-spec* is the bracketed fields following the *ad#*. It specifies the **polar** (and perhaps **azimuthal**) **reference** directions for this ANGLE-dependent source definition.

reference-spec OPTION 1: $u_{pr} \ v_{pr} \ w_{pr} \ \{u_{ar} \ v_{ar} \ w_{ar}\}$

$u_{pr} \ v_{pr} \ w_{pr}$ are the **direction cosines** of the **polar reference** direction;

$u_{ar} \ v_{ar} \ w_{ar}$ are the **direction cosines** of the **azimuthal reference** direction, if needed.

reference-spec OPTION 2: **NORMAL** $\{u_{ar} \ v_{ar} \ w_{ar}\}$

NORMAL specifies that the **polar reference** direction is taken to be along the **outward normal** to the surface source, at the point of emission. This is only allowed for **Surface-type** POSITION-dependent SOURCES.

$u_{ar} \ v_{ar} \ w_{ar}$ are the **direction cosines** of the **azimuthal reference** direction, if needed. This is only allowed for **planar** Surface-type POSITION-dependent SOURCES.

reference-spec OPTION 3: **POINT** $x \ y \ z$

POINT specifies that the **polar reference** direction is taken to be from the point of emission towards the fixed point in space ($x \ y \ z$).

reference-spec OPTION 4: (none specified)

If none of the preceding options for specifying reference directions appears following the *ad#*, COG chooses (0,0,1), the z-direction, as the **polar reference** direction.

These options are followed by a user-specified distribution in $\cos\theta$, and azimuthal angle ϕ , if needed.

Unlike the POSITION-dependent source and the ENERGY-dependent source, an ANGLE-dependent source DEFINITION is *optional*.

Default: ISOTROPIC emission is chosen if no ANGLE specification is given.

This statement is followed by the data for one of the six ANGLE-dependence types listed below. You will need a DEFINE ANGLE definition for each different angle-dependence in your problem.

9.8.1 Source ANGLE Graphics

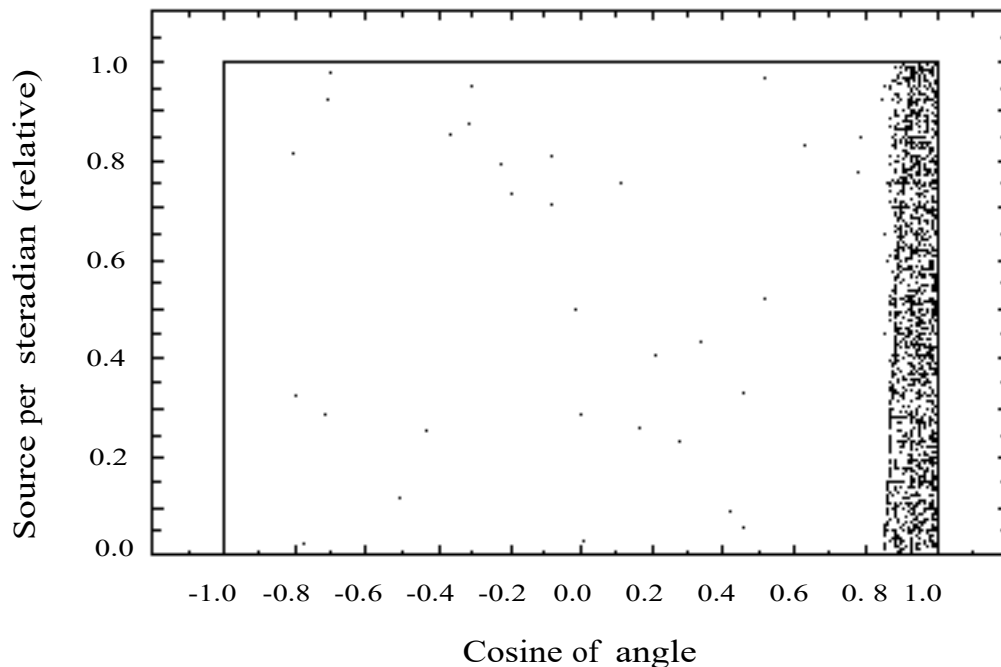
COG generates plots of each problem's ANGLE dependence. An example is shown below.

Example: We begin by defining an ISOTROPIC ANGLE-dependent source. Then by IMPORTANCE-weighting, the source becomes a cone-shaped beam in the +y direction, of half-angle 22.5 degrees. Note that we ask for some samples in the back-angle direction, but we most heavily sample the forward cone, which we expect to provide all the scoring particles for this problem. The requested source angular IMPORTANCE is 0.001 from 180 to 31.8 degrees, rises linearly to 1.0 at 22.5 degrees, and stays at 1.0 to 0.0 degrees.

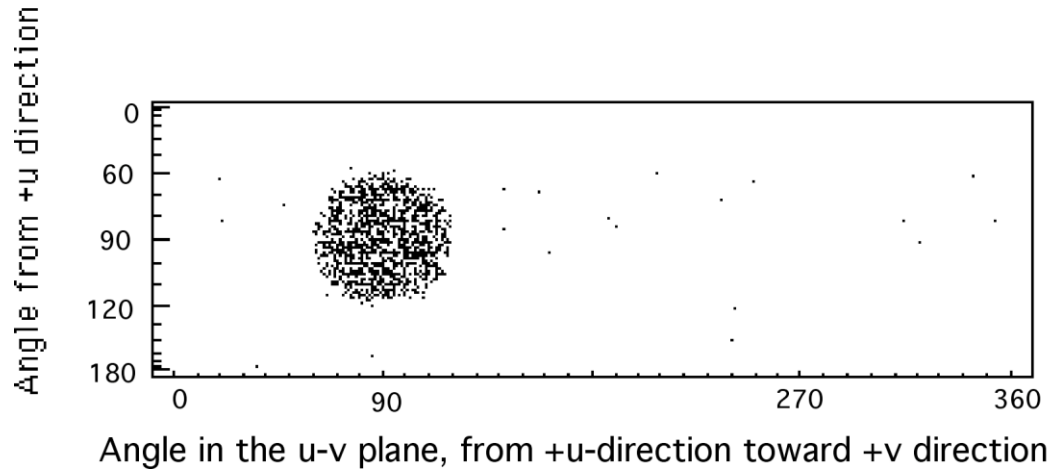
```

DEFINE ANGLE = 2 $ CALL THIS ANGLE-DEPENDENT SOURCE #2
0. 1. 0.          $ POLAR REFERENCE DIRECTION IS +Y AXIS
ISOTROPIC          $ BASIC ANGULAR DEPENDENCE IS ISOTROPIC
IMPORTANCE -1. 0.001 0.85 0.001 .9239 1. 1. 1.
$ SAMPLE ALL ANGULAR SPACE, BUT MOST DENSELY SAMPLE A 22.5
$ DEG CONE [COS(22.5) = 0.9239]

```



Display of points chosen randomly from angle-dependent source #2



9.8.2 FIXED Direction Angle Source

All particles may be emitted along the **polar reference** direction by specifying:

FIXED

If the first *reference-spec* option is used (*upr vpr wpr*), all particles are emitted in this **polar reference** direction. This is sometimes referred to as a *beam* source.

If the second *reference-spec* option is used (**NORMAL**), each particle will start its trajectory along the **outward-directed normal** to the surface specified by the POSITION-dependent SOURCE.

If the third *reference-spec* option is used (**POINT x y z**), each particle will be emitted toward the specified point in space.

An IMPORTANCE function is not needed with this ANGLE-dependent source type since all particles start in the same direction.

Note: Depending on the POSITION-dependent source used with this ANGLE-dependent source, DETECTORs of the POINT type may not receive the uncollided contribution from the source. The user should check the COG output file to be sure that the uncollided contribution has been calculated.

All angles are specified in terms of the angle's cosine.

Example: Define an ANGLE-dependent source which is fixed in the +z direction.

```
DEFINE ANGLE 1  
0. 0. 1.    FIXED
```

9.8.3 ISOTROPIC Angle Source

An ISOTROPIC distribution is probably the most commonly used of all the source angular distributions. It is specified by:

$$\text{ISOTROPIC} \quad \left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance - spec} \right\}$$

If no IMPORTANCE is given, the *reference-spec* has no effect on this distribution.

The optional angle *importance-spec* is a distribution which specifies the relative importance of the angular source over a set of angles. It has two forms:

$$\mu_1 \ i_1 \quad \mu_2 \ i_2 \quad \mu_3 \ i_3 \quad \dots \quad \mu_n \ i_n$$

$$\text{or } \mu_1 \ i_1 \quad \mu_2 \ i_2 \quad \mu_3 \ i_3 \quad \dots \quad \mu_n \ i_n \quad \mu_{n+1} \ i_{n+1}$$

In the first form, **Linearly-interpolated Distribution**, importance is given as i_1 at cosine of angle μ_1 and varies **linearly** to i_2 at cosine of angle μ_2 etc., and reaches i_n at cosine of angle μ_n .

In the second form, **Histogram Distribution**, the importance is constant at a value of i_1 between cosine of angle μ_1 and μ_2 at i_2 between cosine of angle μ_2 and μ_3 , etc., and at i_n between cosine of angle μ_n and μ_{n+1} . Note that the Histogram form requires one more value of μ than the Linearly-interpolated form. The reference direction for the IMPORTANCE specification is the same as that given for the distribution.

Example: Define an *ANGLE-* dependent source which is ISOTROPIC but confined to the +x half-space by the IMPORTANCE specification.

```

DEFINE ANGLE 2 $ LABEL THIS ANGLE DEFINITION #2
      1. 0. 0.    $ POLAR REFERENCE DIRECTION IS IN +X
ISOTROPIC      $ BASIC ANGULAR DEPENDENCE IS
                $ISOTROPIC
IMP  0. 1.  1. 1. $ IMPORTANCE IS UNITY FROM 90 TO 0
                $DEGREES, AND ZERO ELSEWHERE

```

9.8.4 Point-Wise DISTRIBUTION in Polar Angle μ

A complete point-wise DISTRIBUTION in μ , the cosine of the angle between the particle emission direction and the **polar reference** direction, may be given by specifying:

$$\text{DISTRIBUTION } \mu_1 \ s_1 \ \mu_2 \ s_2 \ \mu_3 \ s_3 \ \dots \ \mu_n \ s_n$$

$$\left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \text{importance} - \text{spec} \right\}$$

where the s values represent the relative source intensity *per steradian* at the preceding value of μ . It is assumed that a linear fit between adjacent points accurately describes the distribution. The μ values may be in either increasing or decreasing order. Outside of the specified μ range, the source is zero.

(See ISOTROPIC specification for ANGLE *importance-spec* description).

All angles are specified by the cosine of the angle.

Example: Define an ANGLE dependence which is uniform in a cone between 0° and 26° centered on the +Y axis. [$\cos 0^\circ = 1$, $\cos 26^\circ = .8944$]

```
DEFINE A 3 $ LABEL THIS ANGLE DEPENDENCE #3
0. 1. 0. $ POLAR REFERENCE IS THE +Y DIRECTION
DISTRIBUTION 1. 1. 0.8944 1. $ SPECIFY CONE OF HALF-
$ ANGLE 26 DEGREES
```


9.8.5 BIN Angle Source in Polar Angle μ

A BIN distribution of the angular source strength, which is known only by average values within a specific bin structure, may be specified by

$$\begin{bmatrix} \text{BINS} \\ \text{BIN} \end{bmatrix} \mu_1 s_1 \mu_2 s_2 \mu_3 s_3 \dots \mu_n s_n \mu_{n+1}$$

$$\left\{ \begin{bmatrix} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{bmatrix} \text{importance} - \text{spec} \right\}$$

The s values are the average relative values of the source intensity *per steradian* within each bin and the bins may be specified as in either ascending or descending order of μ . Note the presence of the final extra angle value, μ_{n+1} . When choosing a cosine from this distribution, COG first randomly chooses a bin and then picks a value of μ uniformly within the bin boundaries. The source is zero outside of the specified bin structure.

(See ISOTROPIC specification for ANGLE *importance-spec* description).

All angles are specified by the cosine of the angle.

Example: Define an ANGLE-dependent SOURCE which is uniform in a cone of half-angle 26° , centered about the direction halfway between the $+y$ and $+z$ axes.

```

DEFINE ANGLE 1 $ LABEL THIS ANGLE DEPENDENCE #1
0. 0.707 0.707 $ POLAR REFERENCE IS BETWEEN +Y, +Z
$ DIRECTION
BIN 1. 1. 0.8988 $INTENSITY IS UNITY BETWEEN 0 AND 26
$ DEGREES, AND ZERO ELSEWHERE

```

9.8.6 GENERAL Angle Bins in μ and Φ

For a complete description of an ANGLE source in both the cosine of the polar angle μ , and in the azimuthal angle Φ , specify the following table:

GENERAL	$\min-\mu_1$	$\max-\mu_1$	$\min-\Phi_1$	$\max-\Phi_1$	s_1
	$\min-\mu_2$	$\max-\mu_2$	$\min-\Phi_2$	$\max-\Phi_2$	s_2
	...	...	...	...	...
	$\min-\mu_n$	$\max-\mu_n$	$\min-\Phi_n$	$\max-\Phi_n$	s_n
	$\left\{ \begin{array}{c} \text{[IMPORTANCE]} \\ \text{IMP} \\ \text{I} \end{array} \right\} i_1 \ i_2 \ i_3 \ ... \ i_n$				

The s values are the average relative values of the source *per steradian* within the specified bins, while the minimum and maximum angular values define the bin. Azimuthal angle Φ is specified in degrees. Bins must not overlap, but the set of all specified bins need not cover all of angular space. Those unspecified regions will be considered to have zero source strength. COG makes a random choice of a BIN, based on source strength and importance, then chooses the values of μ and Φ by picking uniformly within the BIN boundaries.

Note that the IMPORTANCE is specified by giving the *average* value within each BIN.

Since the *reference-spec* **OPTION 3, POINT $x \ y \ z$** , does not allow for a azimuthal reference direction, it can not be used for GENERAL BINS.

Example: Define a GENERAL BINS ANGLE-dependent source with user-specified bins. The reference direction for μ is the $+x$ axis, that for Φ is the $+y$ axis.

```

DEFINE A 3 $ LABEL THIS ANGLE DEPENDENCE #3
1. 0. 0. $ POLAR REFERENCE IS THE +X DIRECTION
0. 1. 0. $ AZIMUTHAL REFERENCE IS THE +Y DIRECTION
GENERAL
-1. 0. 0. 90. 9. $ min- $\mu_1$  max- $\mu_1$  min- $\Phi_1$  max- $\Phi_1$   $s_1$ 
-1. 0. 90. 180. 5.
-1. 0. 180. 270. 3.
-1. 0. 270. 360. 1.
0. 1. 0. 90. 90.
0. 1. 90. 180. 50.
0. 1. 180. 270. 30.
0. 1. 270. 360. 10.

```

9.8.7 COG Standard Double-Angle Bin Structure

Needless to say, double-angle bins can be difficult to specify just because of the mass of data required. To simplify this problem, COG has established a standard bin structure which generates double-angle bins with equal solid angles. These can be used both here in the SOURCE angle definitions and also in the DETECTOR specifications. Thus, the results of a COG problem (the DETECTOR scores in angle bins) can more easily be input to a subsequent COG problem as a source bin structure. To specify this form use:

$$\text{GENERAL } N = n \quad s_1 \ s_2 \ s_3 \ \dots \ s_m$$

$$\left\{ \left[\begin{array}{c} \text{IMPORTANCE} \\ \text{IMP} \\ \text{I} \end{array} \right] \quad i_1 \ i_2 \ i_3 \ \dots \ i_m \right\}$$

The value of n determines the complexity of the bin structure. There will be a total of $m = 4n(n+1)$ BINS.

As above, the s_j are the user-supplied source strengths for these BINS. The output from a sample problem, shown below, illustrates the structure for $n = 2$ and shows the COG ANGLE-dependent source picture for this case. When you want to run a problem using this feature, we advise that you try a quick run and print out the boundaries for cases that might interest you. It is easy to get so many BINS that they are almost unusable.

Example: Define a GENERAL BINS ANGLE-dependent source using the COG-supplied bin structure of order 2 (this generates 24 bins). The reference direction for μ is the +z axis, that for ϕ is the +x axis.

```

DEFINE A 3
0. 0. 1. $ POLAR REFERENCE IS THE +Z DIRECTION
1. 0. 0. $ AZIMUTHAL REFERENCE IS THE +x DIRECTION
GENERAL N = 2 $ ASK FOR THE COG BIN STRUCTURE OF ORDER 2
4. 2. 4. 10. 6. 4. 2. 4. 6. 8. 10. 9.
4. 6. 8. 10. 9. 6. 4. 2. 10. 4. 2. 4.

```

This definition results in the following bin structure, as listed in the COG output file:

```

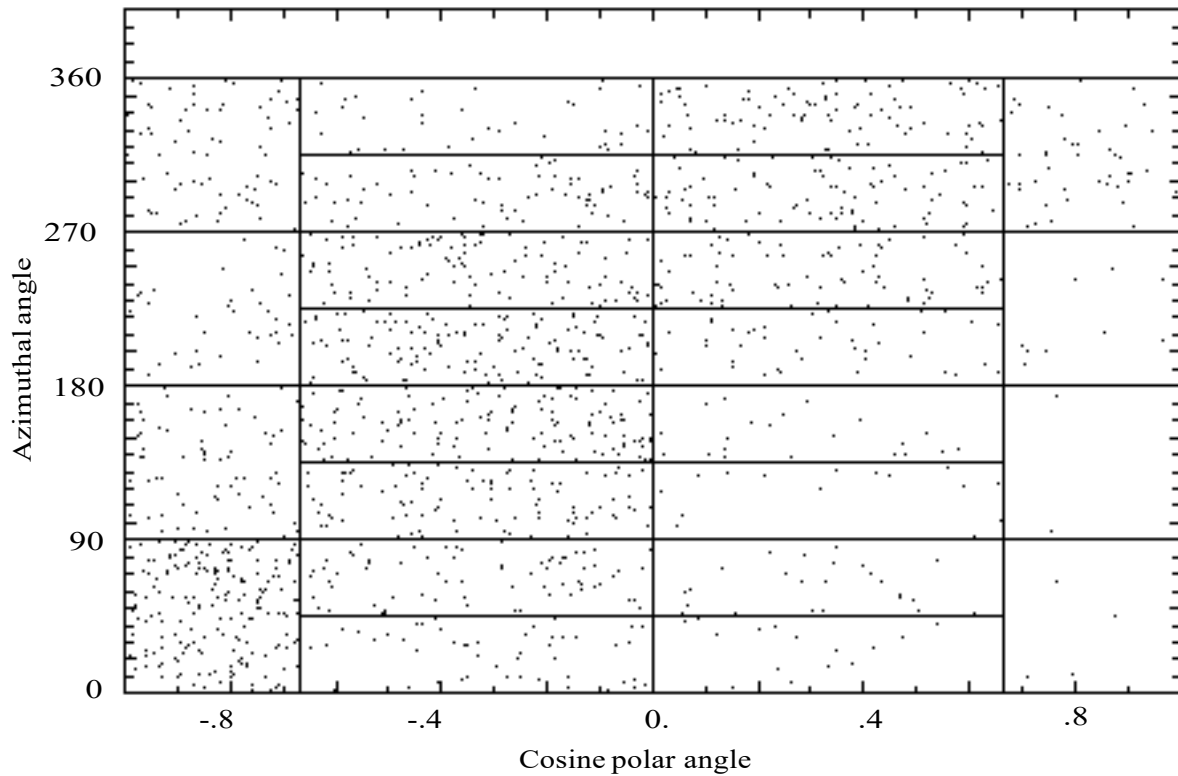
SOURCE--ANGLE:  DEFINITION OF DESCRIPTION      3
REFERENCE DIRECTION FOR POLAR ANGLE IS GIVEN BY
U =   .0000E+00  V =   .0000E+00  W =   1.0000E+00
REFERENCE DIRECTION FOR AZIMUTHAL ANGLE IS GIVEN BY
U =   1.0000E+00  V =   .0000E+00  W =   .0000E+00
GENERAL  N =   2

```

MU-MIN	MU-MAX	PHI-MIN	PHI-MAX	SOURCE	IMPORTANCE
1.0000E+00	6.6667E-01	.0000E+00	9.0000E+01	4.0000E+00	1.0000E+00
1.0000E+00	6.6667E-01	9.0000E+01	1.8000E+02	2.0000E+00	1.0000E+00
1.0000E+00	6.6667E-01	1.8000E+02	2.7000E+02	4.0000E+00	1.0000E+00
1.0000E+00	6.6667E-01	2.7000E+02	3.6000E+02	1.0000E+01	1.0000E+00
6.6667E-01	.0000E+00	.0000E+00	4.5000E+01	6.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	4.5000E+01	9.0000E+01	4.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	9.0000E+01	1.3500E+02	2.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	1.3500E+02	1.8000E+02	4.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	1.8000E+02	2.2500E+02	6.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	2.2500E+02	2.7000E+02	8.0000E+00	1.0000E+00
6.6667E-01	.0000E+00	2.7000E+02	3.1500E+02	1.0000E+01	1.0000E+00
6.6667E-01	.0000E+00	3.1500E+02	3.6000E+02	9.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	.0000E+00	4.5000E+01	4.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	4.5000E+01	9.0000E+01	6.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	9.0000E+01	1.3500E+02	8.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	1.3500E+02	1.8000E+02	1.0000E+01	1.0000E+00
.0000E+00	-6.6667E-01	1.8000E+02	2.2500E+02	9.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	2.2500E+02	2.7000E+02	6.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	2.7000E+02	3.1500E+02	4.0000E+00	1.0000E+00
.0000E+00	-6.6667E-01	3.1500E+02	3.6000E+02	2.0000E+00	1.0000E+00
-6.6667E-01	-1.0000E+00	.0000E+00	9.0000E+01	1.0000E+01	1.0000E+00
-6.6667E-01	-1.0000E+00	9.0000E+01	1.8000E+02	4.0000E+00	1.0000E+00
-6.6667E-01	-1.0000E+00	1.8000E+02	2.7000E+02	2.0000E+00	1.0000E+00
-6.6667E-01	-1.0000E+00	2.7000E+02	3.6000E+02	4.0000E+00	1.0000E+00

This graphic depicts the source bins for the GENERAL n=2 bin structure.

```
SOURCE--ANGLE. Definition of description      3
reference direction for polar angle is given by
u = .0000E+00  v = .0000E+00  w = 1.0000E+00
reference direction for azimuthal angle is given by
u = 1.0000E+00  v = .0000E+00  w = .0000E+00
GENERAL n = 2
```

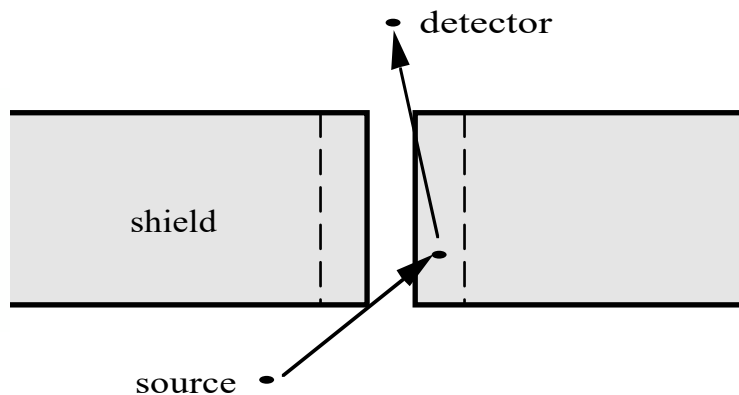


9.9 SOURCE-PATH Option

In some problems, the particle flux at the detector comes mostly from particles which have scattered into the detector from certain limited regions of the geometry. These scattering regions may be so small or transparent to radiation that few source particles interact there. Thus the number of scoring particles is very small, and the problem would have to be run a very long time to get a reasonably good answer.

The SOURCE-PATH option is a method for forcing source particles to collide in the important scattering regions, and thereby increase the number of scoring particles. COG will calculate the uncollided particle flux from the source at randomly-chosen points within the specified scattering volumes. COG forces collisions to occur at these points, and the particles emerging from the collision sites form the source which will be used for the transport random walk.

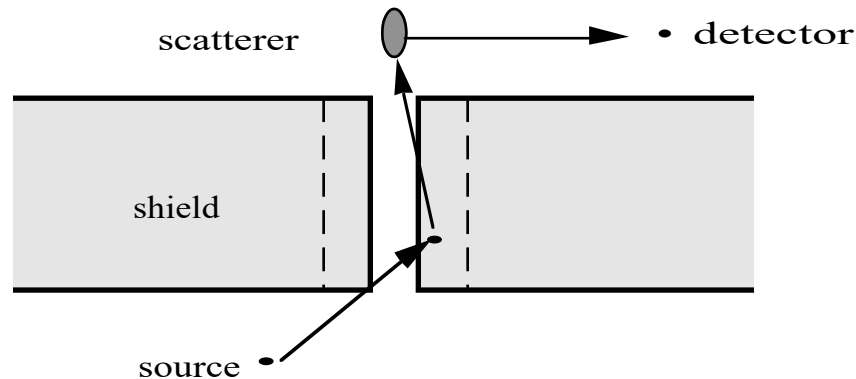
We illustrate this method with the following example, a simple case where this method can be used to advantage. We have a slab shield with a small cylindrical hole running through it, and a detector at the back side of the hole. Our source is on the front side of the slab but not in line with the hole.



If the shield is thick, essentially all of the particles which arrive at the detector will first scatter one or more times in the shield. The major detector flux contribution comes from source particles which first collide in a small cylindrical annulus around the hole (shown by dashed lines), then scatter into the detector. In the SOURCE-PATH option, we specify the cylindrical annulus as a ***first-collision volume***, and COG will use the forced first-collision sites in this volume as the source for the problem's random walk.

The SOURCE-PATH method can handle more complicated cases where the major detector contribution comes from second-collision or third-collision sources. We can complicate our first example by moving the detector away from the hole, and placing

an ellipsoidal scatterer in its place. In this SOURCE-PATH problem we need to specify the cylindrical annulus as a first-collision volume, and the ellipsoidal scatterer as a *second-collision volume*.



In this case COG proceeds as before in computing first-collision sites in the annulus. It then computes collision sites in the ellipsoidal scatterer for particles emerging from the forced first collisions in the annulus. The random walk transport will start with the particles emerging from the ellipsoidal *second-collision volume*.

Examples involving a *third-collision volume* can be imagined.

When SOURCE-PATH is used, the **only** scores in your detector come from the collision volumes you specify. Pieces of your geometry not specified to COG under this option will not be sampled. Therefore you must be careful to specify all the important paths by which particles can reach your detector, and all the collision volumes. Each path may have zero, one, two, or three collision volumes. The physical source must also be specified. If you omit a significant path, you will get a result which is too small. If you define two or more paths which contain identical overlapping pieces, you will get double-counting and a result which is too high. If there are too many paths to easily define them all, you should probably not use this method. To activate the SOURCE-PATH option, insert this block of data:

SOURCE-PATH

PATH *pid#* {V1 = *cv1#*} {V2 = *cv2#*} {V3 = *cv3#*}

{I = *i*}

PATH *pid#* ...

...

DEFINE VOLUME *vid#*

BOX
CYLINDER
CONE
SPHERE

DEFINE VOLUME *vid#* ...

DEFINE VOLUME *vid#* ...

...

Where:

pid# is an integer assigned by the user to a given path. Its only purpose is to help the user to keep track of the paths;

V1 is a keyword indicating that a *collision volume number* follows for the *first-collision volume* in the path; *cv1#* is the user-assigned *collision volume number* which identifies the VOLUME DEFINITION to be used as the *first-collision volume* in this path. If no *second-* or *third-collision volumes* are specified, COG will start the random walk from this *first-collision* source.

Default: If no collision volumes are specified (i.e., if the **PATH** statement has no **Vn** = data), then the source for this PATH will be the normal source as specified by the SOURCE Data Block.

V2 is a keyword indicating that a *collision volume number* follows for the *second-collision volume* in the path. *cv2#* is the user-assigned *collision volume number* which identifies the VOLUME DEFINITION to be used as the *second-collision volume* in this path. If no *third-collision volume* is specified, COG will start the random walk from this *second-collision* source. **Note:** There must be a *first-collision volume* defined for this path.

V3 is a keyword indicating that a *collision volume number* follows for the *third-collision volume* in the path. **cv3#** is the user-assigned *collision volume number* which identifies the VOLUME DEFINITION to be used as the *third-collision volume* in this path. COG will start the random walk from this *third-collision* source.

Note: There must be a *first-collision volume* and a *second-collision volume* defined for this path.

i is the IMPORTANCE of this PATH relative to the other defined paths. The frequency with which COG samples any of the various defined PATHS is proportional to the PATH IMPORTANCE.

Default: *i* = 1.0.

vid# is a positive integer used to identify the following *collision volume* DEFINITION. **vid#** must be one of the **cvn#** used in the PATH specifications. The entire set of **DEFINE VOLUME vid#**s must contain all the **cvn#**s mentioned in all the PATH statements.

BOX . . ., **CYLINDER . . .**, **CONE . . .**, and **SPHERE . . .** are the keywords to specify the *collision volumes* and are identical to those given for volume sources in the section Position Dependence of the SOURCE. Alternate names, such as **SPH** and **S** for the sphere, are allowed. One can also restrict the *collision volume* to a given material or region number, as described in the section Arbitrary Volume Source.

Example: A simple SOURCE-PATH with one path and one *first-collision volume* in the path. The volume is a cylinder of radius 3, with axis along the +x axis, and bounded by planes at $x = 234$ and 240.

```
SOURCE-PATH
PATH 1   V1=5
DEFINE VOLUME 5
CYLINDER 234. 0. 0. 240. 0. 0. 3.
```

Example: A more complex SOURCE-PATH specification with two paths. Path 1 is defaulted to the normal COG source. Path 2 has **first- and second-collision volume sources**. Volume 5 is a cylinder of radius 3, with axis along the +x axis, and bounded by planes at $x = 234$ and 240 . Volume 10 is a sphere of radius 0.1, located at (237.,0.,10.).

```
SOURCE-PATH
PATH 1  IMP = 1.  $ USE NORMAL COG SOURCE
PATH 2  V1=5  V2=10  IMP = 2.  $ PATH WITH A FIRST-
                                $ COLLISION AND A SECOND-COLLISION
                                $ VOLUME SOURCE

DEFINE VOLUME 5
CYLINDER 234. 0. 0. 240. 0. 0. 3.
DEFINE VOLUME 10
SPHERE 237. 0. 10. 0.1
```

9.10 USRSOR – User-Defined Source Option

The USRSOR feature allows the user to generate a special source via a user-written subroutine or set of subroutines. This feature allows the use of much more complicated sources than that permitted by the standard COG source options. For example, one could generate an energy-angle correlated source, or read in a source created by some other code.

Like the USRDET user-specified detector, the user's USRSOR Fortran or C routines must be separately compiled and placed in a runtime (shared) library (named "COGUserlib.sl" on the HP and "COGUserlib.so" on all other platforms), which will be loaded by COG at problem run time. When a USRSOR is specified, all source particles will come from the user's source routine, and none from the standard COG source types.

The USRSOR line must immediately follow the NPART line in the SOURCE Block. It has the form:

USRSOR *subname* { *arg1 arg2 ...* } { FILE *filename* }

where:

subname is the name of the Fortran or C subroutine you are providing to generate the source;

arg1 arg2 ... are the (optional) arguments you need to specify the source parameters to your routine (limit of 20);

FILE *filename* optionally specifies a filename to your source routine, perhaps to identify an input or output file needed by your routine.

Example: This COG Source Block specifies a user-defined source named *IsoP*, a simple isotropic point source.

```

SOURCE
  NPART = 10000
$      NAME TYPE      STRENGTH  X  Y    Z  E  T
  USRSOR ISOP PHOTON          1.  0.  0.  0. 10. 0.
      FILE FILE1

```

Here IsoP is the name of the user subroutine, and the arguments in this case are:

photon : Sets particle type;

1. : Specifies source (increment) strength;

0. 0. 0. : Position coordinates of point source;

10. : Energy of each source photon (MeV);

0. : Age of particle at time of emission.

FILE file1 : optionally specifies the file 'file1' for use by IsoP.

COG does not interpret the arguments but merely gathers them up into two arrays of ASCII values (character*8) and their corresponding real (real*8) values, and passes them into your routine (here IsoP).

See the USRSOR directory in the COG distribution for instructions on preparing a USRSOR. A working example is provided.

9.11 CENSUS Source File Option

A census file can be produced for any Boundary-Crossing detector in a problem by having the keyword CENSUS in the detector definition:

BOUNDARY {COUNTS} CENSUS *reg-ID#₁ reg-ID#₂ area*

(See the **Detector Data Block** for a complete description of detector definitions).

This produces a file named:

inputfile.detname.census

A CENSUS file contains the particle parameters for each scoring particle, including position, direction, age, weight, energy, cos(theta) wrt boundary normal, particle type, source particle number, and # of collisions. The particles are subject to all detector masks specified. The file is written in text mode so that it can be edited if desired.

This feature permits the user to run a complex shielding calculation in stages, using a different set of biasing options for each stage. For example, in the first calculation particles might be directed towards a scattering center and those reaching the center saved into a CENSUS file. In a subsequent calculation, those CENSUS particles could be used as a secondary source of particles which are biased towards a remote detector.

Another use of the CENSUS file is to permit post-processing of scoring particles.

To use a CENSUS file as a source for a subsequent COG run, use this line after the NPART line in the SOURCE Data Block:

CENSUS *source-strength* {STEADY} FILE *filename*

where:

source-strength is the source intensity to be used (same as the INCREMENT source-strength);

STEADY indicates that the source is a steady-state source.

filename is the name of the previously-written detector CENSUS file.

The number of particles run will be the number of particles in the file, or NPART, whichever is less.

All other normal source descriptions in the Data Block will be ignored.

***Example:** This COG Source Block specifies a CENSUS source file named IN1.DET1.CENSUS.*

SOURCE

NPART = 10000

CENSUS 1.0 FILE IN1.DET1.CENSUS

9.12 Source Examples

This section contains a series of examples. Each example represents a complete SOURCE description. COG input is basically form-free, but you are advised to adopt conventions which are consistent and easy for you to read and understand. The examples presented below are in "standard" example style (all capitals) used throughout this manual.

Example 1: *In this two-part example, we show alternate methods of constructing an angle-limited source. This also serves to illustrate the power of IMPORTANCEs for biasing a source (to increase running efficiency) without biasing detector scores.*

In this problem, we model an isotropically-emitting source. We assume we know that only particles emitted in a 22.5 degree cone towards our detector will score; all others are absorbed by shielding or leave the problem without interacting. So in order to run the problem more efficiently, we will restrict source emission to this scoring cone. In both cases, we emit from a spherical volume of radius 0.25, centered at (0,73.720,0), with a triangular spectrum peaking at 0.025 MeV. The center of the scoring cone is along the -y direction, which we designate as the POLAR REFERENCE direction for the problem. The units of the problem are set to millimeters in the BASIC Data Block (not shown); default energy units of MeV and time units of seconds are assumed. No TIME-dependence of the source is specified—so the default delta-function at $t=0$ is used.

Example 1.1: *In this first part of the example, we will simply restrict emission to the scoring cone, by specifying a source angle DISTRIBUTION in polar angle, measured from the POLAR REFERENCE direction (the -y direction). The angular DISTRIBUTION is 0 from 180 to 22.5 degrees, then jumps to 1 and stays at 1 to 0 degrees.*

-----Input File for Example 1.1-----

```

SOURCE
  NPARTICLES = 1000000
  DEFINE POSITION = 1
    SPHERE 0. 73.720 0.      .25 $ SPHERE VOLUME SOURCE
  DEFINE ANGLE = 1
    0. -1. 0.$ POLAR REFERENCE DIRECTION IS -Y
  DISTRIBUTION -1. 0. 0.9238 0. 0.9239 1. 1. 1.$ EMIT INTO
    $ 22.5 DEGREE CONE
  DEFINE ENERGY = 1 PHOTON      $ PHOTON ENERGY SOURCE
  DISTRIBUTION 0. 0.      .025 1.0  .06 0. $ TRIANGULAR
    $ SPECTRUM, PEAKING AT 0.025 MEV, AND
    $ ZERO BEYOND 0.06 MEV

  INCREMENT 1.0  P=1 E=1 A=1

```

-----Output Listing for Example 1.1-----

```

AVERAG SOURCE
      MINIMUM      MAXIMUM      AVERAGE
PHOTON
  X  -2.4953E-01    2.4921E-01    1.2681E-04 MM
  Y   7.3470E+01    7.3970E+01    7.3720E+01 MM
  Z  -2.4899E-01    2.4961E-01   -1.9763E-04 MM
  U  -3.8250E-01    3.8272E-01   -4.9314E-05
  V  -1.0000E+00   -9.2380E-01   -9.6187E-01
  W  -3.8259E-01    3.8267E-01   -7.4482E-04
  T   .0000E+00    .0000E+00    .0000E+00 SEC
  E   9.0694E-05    5.9912E-02    2.8347E-02 MEV
  WT  1.0000E+00    1.0000E+00    1.0000E+00

```

Note that the average source position of emitted particles is close to (0.,73.720,0.), the source center, and no points are outside of the specified source. The direction cosines (u, v, w) show that the emitted particles are confined to the scoring cone, as specified.

Note: Detector scores will have to be hand-corrected by the user to compensate for the anisotropic bias of this source. The scores are not what you would get if an isotropically-emitting source had been used.

The source energy has an average of .028 MeV—about as expected for the triangular energy distribution of the source. The particle weights are uniform.

Example 1.2: In this second part of the example, we use IMPORTANCES to bias the ISOTROPIC source to emit predominately into the scoring cone (oriented in the -y

direction from the source- the specified POLAR REFERENCE direction). The IMPORTANCE is specified to be 0.001 from 180 to 31.8 degrees, rises linearly to 1.0 at 22.5 degrees, and stays at 1.0 to 0.0 degrees.

-----Input File for Example 1.2-----

```

SOURCE
  NPARTICLES = 1000000
  DEFINE POSITION = 1
    SPHERE 0. 73.720 0. .25 $ SPHERE VOLUME SOURCE
  DEFINE ANGLE = 1
    0. -1. 0. $ POLAR REFERENCE DIRECTION IS -Y

ISOTROPIC $ BASIC ANGULAR SOURCE IS ISOTROPIC
  IMP -1.0 .001 0.85 .001 .9238 1. 1.0 1.0 $ IMPORTANCE:
    $ SAMPLE DENSELY WITHIN 22.5 DEGREE CONE
  DEFINE ENERGY = 1 PHOTON $ PHOTON ENERGY SOURCE
  DISTRIBUTION 0. 0. .025 1.0 .06 0. $ TRIANGULAR
    $ SPECTRUM, PEAKING AT 0.025 MEV, AND
    $ ZERO BEYOND 0.06 MEV

INCREMENT 1.0 P=1 E=1 A=1

```

-----Output Listing for Example 1.2-----

AVERAGE SOURCE PARAMETERS

	minimum	maximum	average	
photon				
x	-2.4987E-01	2.4956E-01	-1.0805E-03	mm
y	7.3470E+01	7.3969E+01	7.3719E+01	mm
z	-2.4979E-01	2.4987E-01	2.3880E-04	mm
u	-9.9992E-01	9.9976E-01	5.1699E-03	
v	-1.0000E+00	9.9991E-01	-2.6929E-03	
w	-9.9999E-01	9.9996E-01	-4.1166E-03	
t	.0000E+00	.0000E+00	.0000E+00	sec
E	2.1143E-05	5.9937E-02	2.8185E-02	MeV
wt	5.7493E-02	5.7493E+01	9.8766E-01	

The direction cosines of the emitted particles each vary from -1 to +1, with an average of about zero. Note in particular that the average v (direction cosine in the y-direction) is nearly zero, even though the great majority of particles were emitted into the 22.5 degree scoring cone. The code has properly modified the weights of the emitted particles so that the weighted source averages seen here are correct for the isotropic source, and the detector scores will be correct also.

Example 2: A three INCREMENT problem with three POSITION dependences (three spherical shells) and two ENERGY dependences (two LINE spectra). The source strengths for the three shells are 4.67×10^8 , 8.46×10^8 , and 1.74×10^6 respectively. The INCREMENTS are IMPORTANCE-weighted in the ratio of 1, 10, and 100 respectively. For this problem, NPART = 900000.

The expected number of particles to be emitted by each of the three INCREMENTS is obtained from the rule:

$$\langle N_j \rangle = \text{NPART} \left[\frac{S_j I_j}{\sum S_j I_j} \right]$$

This gives: 46182, 836611, and 17207 respectively. The ANGLE dependence defaults to isotropic and the TIME dependence defaults to a delta function at $t=0$.

```

SOURCE
NPART 900000
DEFINE POSITION 1
SPHERE 0. 0. 0. 4.75 1.58
DEFINE POSITION 2
SPHERE 0. 0. 0. 10.99 10.45
DEFINE POSITION 3
SPHERE 0. 0. 0. 13.85 13.59
DEFINE ENERGY 1
PHOTON LINE
143.786 8396.
163.379 3758.
185.739 42380.
205.333 3758.
345.921 30.39
387.874 30.39
390.32 31.99
766.412 1.6475
1001. 5.170
DEFINE ENERGY 2
PHOTON LINE
742.817 9.378
766.412 32.95
1001. 103.4
1737.8 2.259
1831.7 1.782
INC 4.67E8 P 1 E 1 I 1.

```

```

INC 8.46E8  P 2  E 1  I 10.
INC 1.74E6  P 3  E 2  I 100.

```

Example 3: Here we handle a single physical source with a complex ENERGY dependence by breaking it into two INCREMENTS, each with a simple energy dependence. The source POSITION dependence for both INCREMENTS is that part of a sphere, centered at (0.,0.,0.) with radius 0.0375, which contains material 6. We specify two ENERGY dependences: first is the Co⁵⁷ photon cascade, 0.1219 to 0.0144; second is the Co⁵⁷ line at 0.1363. These are placed into separate INCREMENTS which have source-strengths in the ratio of 0.89 to 0.11. Together these form the desired total COG source. (The ANGLE and TIME dependences are defaulted). Energy units default to MeV.

```

SOURCE
NPART 10000  $ EMIT 10000 PARTICLES
DEFINE POSITION 1
SPHERE 0. 0. 0.  0.0375  MAT 6 $ EMIT FROM SPHERE;
                                $ MATERIAL #6 ONLY.

DEFINE ENERGY 1
PHOTON CASCADE 0.1219 0.0144 1. 0. 0.
INC 0.89  P 1  E 1  $ INCREMENT 1 USES CASCADE SOURCE
DEFINE ENERGY 2
PHOTON LINE 0.1363 1.
INC 0.11  P 1  E 2  $ INCREMENT 2 USES LINE SOURCE

```

Example 4: This is a one-INCREMENT problem. The POSITION dependence is a SURFACE-type source in the shape of a parallelogram. The ENERGY dependence is the 662 keV line source from Cs¹³⁷ (The energy units have been set to keV in the BASIC Data Block). The TIME dependence is STEADY STATE. The ANGLE dependence is FIXED NORMAL to the plane of the parallelogram. The source strength is 3.7×10^{10} photons per second.

```

SOURCE
NPART 100000
DEFINE POSITION 1
SS-PARALLELOGRAM
-0.3535 -0.4999 0.3535
 0.3535 -0.4999 -0.3535
 0.3535 0.4999 -0.3535
DEFINE ENERGY 1
PHOTON LINE 662. 1.
DEFINE TIME 1
STEADY
DEFINE ANGLE 1
NORMAL FIXED
INC 3.7E10 P 1 E 1 T 1 A 1

```

Example 5: This is a one-INCREMENT problem with a nominal source strength of 1. The POSITION dependence is a POINT at (0.,0.,0.). The ENERGY dependence is represented as binned neutron data with binned IMPORTANCES. The TIME dependence is a GAUSSIAN centered at 0. with a width of 7×10^{-9} . The ANGLE dependence is defaulted.

```

SOURCE
NPART 20000
DEFINE POSITION 1
POINT 0. 0. 0.
DEFINE ENERGY 1
NEUTRON BIN 14.68 0.0819 13.54 0.0145 12.5 0.0091
              11.55 0.0084 9.67 0.0116 7.91 0.0244
              5.37 0.0605 4.7 0.1959 2.53 0.6304
              1.18 1.32 0.633
IMPORTANCE 14.68 1. 13.54 0.9 12.5 0.4
              11.55 0.3 9.67 0.2 7.91 0.1
              5.37 0.1 4.7 0.1 2.53 0.05
              1.18 0.05 0.633

```

```

DEFINE TIME 1
GAUSSIAN 0. 7.E-9
INC 1.  P 1  E 1  T 1

```

Example 6: *There is a single increment with a nominal source strength of 1. The POSITION dependence is the surface of a disk centered at (0.,0.,200.) with a radius of 0.5; the normal is along the -Z axis. The ENERGY dependence is a Cf-252 spontaneous fission source of neutrons (in FREYA mode). The ANGLE dependence is uniform from 0° to 20° about the -Z axis; IMPORTANCES weight the angles about 0° more heavily. The TIME dependence is defaulted.*

```

SOURCE
NPART 5000000
DEFINE POSITION 1
SS-DISK 0. 0. 200.  0. 0. 0.  0.5
DEFINE ENERGY 1
NEUTRON SFS 98252
DEFINE ANGLE 1
0. 0. -1.  BIN 1. 1.  0.94
IMP 1. 10.  0.99 8.  0.98 5.  0.97 3.  0.96 2.
      0.94
INC 1.  P 1  E 1

```

Example 7: *A description of the differential energy distribution from a D,T neutron source with an incident deuteron beam of 300 keV. This is an example of a source which has a correlated energy-angle dependence, i.e. the energy and angle dependencies are **not** separable. We approximate this correlation with a series of 20 ANGLE bins; in each ANGLE bin the ENERGY dependence is given as a bin with the appropriate range. The POSITION dependence is a DISK centered at (0.,0.,0.), normal to the +X axis, and with a radius of 0.3. There are 20 ANGLE bins and 20 corresponding ENERGY bins. The 20 INCREMENTS combine the corresponding ANGLE and ENERGY bins with appropriate source strengths.*

```

SOURCE
NPART 40000
DEFINE POSITION 1
SS-DISK 0. 0. 0.  1. 0. 0.  0.3
DEFINE ANGLE 1
1. 0. 0.  BIN 1. 1.  0.9877
DEFINE ANGLE 2
1. 0. 0.  BIN 0.9877 1.  0.9511
DEFINE ANGLE 3
1. 0. 0.  BIN 0.9511 1.  0.891

```

```
DEFINE ANGLE 4
1. 0. 0.  BIN 0.891 1.  0.809
DEFINE ANGLE 5
1. 0. 0.  BIN 0.809 1.  0.7071
DEFINE ANGLE 6
1. 0. 0.  BIN 0.7071 1.  0.5878
DEFINE ANGLE 7
1. 0. 0.  BIN 0.5878 1.  0.454
DEFINE ANGLE 8
1. 0. 0.  BIN 0.454 1.  0.309
DEFINE ANGLE 9
1. 0. 0.  BIN 0.309 1.  0.1564
DEFINE ANGLE 10
1. 0. 0.  BIN 0.1564 1.  0.
DEFINE ANGLE 11
1. 0. 0.  BIN 0. 1.  -0.1564
DEFINE ANGLE 12
1. 0. 0.  BIN -0.1564 1.  -0.309
DEFINE ANGLE 13
1. 0. 0.  BIN -0.309 1.  -0.454
DEFINE ANGLE 14
1. 0. 0.  BIN -0.454 1.  -0.5878
DEFINE ANGLE 15
1. 0. 0.  BIN -0.5878 1.  -0.7071
DEFINE ANGLE 16
1. 0. 0.  BIN -0.7071 1.  -0.809
DEFINE ANGLE 17
1. 0. 0.  BIN -0.809 1.  -0.891
DEFINE ANGLE 18
1. 0. 0.  BIN -0.891 1.  -0.9511
DEFINE ANGLE 19
1. 0. 0.  BIN -0.9511 1.  -0.9877
DEFINE ANGLE 20
1. 0. 0.  BIN -0.9877 1.  -1.
DEFINE ENERGY 1
NEUTRON BIN 15.37048 1.  15.35482
DEFINE ENERGY 2
NEUTRON BIN 15.35482 1.  15.30831
DEFINE ENERGY 3
NEUTRON BIN 15.30831 1.  15.23236
DEFINE ENERGY 4
```

```
NEUTRON BIN 15.23236 1. 15.12927
DEFINE ENERGY 5
NEUTRON BIN 15.12927 1. 15.0021
DEFINE ENERGY 6
NEUTRON BIN 15.0021 1. 14.85455
DEFINE ENERGY 7
NEUTRON BIN 14.85455 1. 14.69083
DEFINE ENERGY 8
NEUTRON BIN 14.69083 1. 14.51545
DEFINE ENERGY 9
NEUTRON BIN 14.51545 1. 14.33314
DEFINE ENERGY 10
NEUTRON BIN 14.33314 1. 14.14861
DEFINE ENERGY 11
NEUTRON BIN 14.14861 1. 13.96647
DEFINE ENERGY 12
NEUTRON BIN 13.96647 1. 13.7911
DEFINE ENERGY 13
NEUTRON BIN 13.7911 1. 13.62655
DEFINE ENERGY 14
NEUTRON BIN 13.62655 1. 13.47645
DEFINE ENERGY 15
NEUTRON BIN 13.47645 1. 13.34401
DEFINE ENERGY 16
NEUTRON BIN 13.34401 1. 13.23195
DEFINE ENERGY 17
NEUTRON BIN 13.23195 1. 13.1425
DEFINE ENERGY 18
NEUTRON BIN 13.1425 1. 13.07737

DEFINE ENERGY 19
NEUTRON BIN 13.07737 1. 13.03781
DEFINE ENERGY 20
NEUTRON BIN 13.03781 1. 13.02454
INC 1.085172 P 1 E 1 A 1
INC 1.084079 P 1 E 2 A 2
INC 1.080834 P 1 E 3 A 3
INC 1.075529 P 1 E 4 A 4
INC 1.068318 P 1 E 5 A 5
INC 1.059405 P 1 E 6 A 6
INC 1.049040 P 1 E 7 A 7
```

—

INC 1.037510	P 1	E 8	A 8
INC 1.025126	P 1	E 9	A 9
INC 1.012213	P 1	E 10	A 10
INC 0.999105	P 1	E 11	A 11
INC 0.986128	P 1	E 12	A 12
INC 0.973597	P 1	E 13	A 13
INC 0.961807	P 1	E 14	A 14
INC 0.951025	P 1	E 15	A 15
INC 0.941490	P 1	E 16	A 16
INC 0.933407	P 1	E 17	A 17
INC 0.926944	P 1	E 18	A 18
INC 0.922233	P 1	E 19	A 19
INC 0.919368	P 1	E 20	A 20

9.13 References

- 1.0 R. Y. Rubinstein, Simulation and the Monte Carlo Method,
(John Wiley & Sons, NY, 1981).

10 The DETECTOR Data Block

10.1 Introduction to COG Detectors

A detector is used to specify what type of information the user wants to gain from the COG calculation. A detector in the physical sense is a “gadget” that can detect and record the number and energy of particles in a given location in the experimental space. A similar terminology is used to describe a detector within COG. Since COG detectors are mathematical entities, they do not suffer from most of the faults of real detectors and can be given any size or response or resolution you desire. COG detectors can be made sensitive to any property of the particles, including properties which most real detectors cannot sense—such as particle age, particle type, and angle of incidence. COG computes DETECTOR output—which can be the particle current across a surface, the particle flux at a point, the number of collisions in a volume, etc.—by analyzing in detail the random walk histories of all problem particles.

Each event in a particle’s random walk—its birth at the source, a boundary crossing, a scattering, a splitting, a Russian roulette outcome, etc.—is recorded in the particle’s Event History record. This particle history includes all secondary particles produced by the original source particle and their subsequent histories. (The printout produced by the RETRACE option is an abridged version of the particle Event History record.)

At the end of each source particle history, COG analyzes the particle’s Event History record to compute incremental detector scores. Four basic detector types have been built into COG. These are the REACTION, BOUNDARY-CROSSING, POINT, and PULSE detector types. In addition, the user can write his own routine for specialized processing; this is termed the USRDET detector type.

The basic quantity scored for any detector type (except PULSE) is particle fluence (or number flux), in units of particles/cm² per source particle, multiplied by source strength. (Exception: if the source specifies a STEADY state TIME-dependence, then the basic detector score is given in units of particles/cm²-sec.)

This basic score can be converted to energy flux, dose, or other quantity by specifying an appropriate detector response function.

The particles which are allowed to score in a detector can be limited by MASKS, which specify allowed ranges of particle ENERGY, TIME, and ANGLE, for example.

Differential results for any detector can be obtained in ENERGY, TIME, and ANGLE space by specifying one or more BIN structures.

It is often a good idea to place multiple detectors, with multiple ENERGY BINs or response functions, at the same location in a problem. These extra detectors take little additional computational time and may give extra insight into the problem.

10.2 Outline of the Detector Data Block

The DETECTOR Data Block specifies all the detectors in a problem. This Data Block has the form:

DETECTOR

NUMBER = *label* { **TITLE** = ". . ." }

detector type and associated data

{*response functions*}

{*masks*}

{*differential bins*}

NUMBER = *label* { **TITLE** = ". . ." }

detector type and associated data

{*response functions*}

{*masks*}

{*differential bins*}

. . .

where:

NUMBER is the keyword indicating the beginning of a new detector definition;

label is an identifying ASCII label (up to 8 characters) for this detector definition;

{ **TITLE** = ". . ." } is an optional title associated with this detector, used for identification only, and is printed along with the calculated results. The quotes *must* be included around the title text;

detector type and associated data specify the detector type (REACTION, BOUNDARY-CROSSING, POINT, PULSE, USRDET) and location, along with any other data necessary to define each detector type (see section **COG Detector Types**);

{*response functions*} are optional ENERGY, TIME, or ANGLE functions which convert the particle flux to the desired output quantity. Responses are available which will convert neutron and/or gamma flux to a biological dose, gamma flux to an electron production rate, etc. If omitted, detector results will be given in terms of particle number flux (particles/cm² per source particle, times source strength). Exception: if the source specifies a STEADY state TIME-dependence, then the basic detector score is given in units of particles/cm²-sec.

Note: The units of a response are **not** changed by any units specified in the BASIC Data Block.

{masks} are optional MASKs which limit detector scoring to particles which fall into specified ranges of ENERGY, TIME, or ANGLE, for example. Masks allow the user to specify important detector properties, such as collimation or energy thresholds;

{differential bins} define optional ENERGY, TIME, and/or ANGLE BINs into which the scoring particles will be sorted. For example, if an ENERGY BIN structure is specified, then the incident energy of each scoring particle is sorted into one of the ENERGY BINs. A similar procedure is followed for TIME and ANGLE BINs.

10.2.1 Detector Output Units

The basic detector response shown in the COG output file is the omnidirectional particle flux, in units of *particles per square centimeter per source particle emitted, multiplied by source strength*. (Exception: if a STEADY state source has been specified, the units are *particles per square centimeter per second*.) If the user has specified a **response function** for the detector, then the flux is converted to the specified response, e.g. energy deposited in the detector per source particle emitted or per unit time.

Differential bins produce scores in an ENERGY, TIME, or ANGLE BIN structure. The units of these BINned scores will be response per unit energy for ENERGY BINs, response per unit time for TIME BINs, and response per steradian for ANGLE BINs. The user can specify desired units of energy and time in the BASIC Data Block; these units will otherwise default to MeV and seconds respectively.

10.2.2 Detector Results

Detector results are written into the standard COG output text file, the....**out** file. An abbreviated form of the detector results is also written into a tab-delimited text file—the**det** file. The**det** file presents detector results in a form convenient for plotting or post-processing. (The meaning of each column in this file can be deduced by comparing with the**out** file.) COG also plots detector results in its graphics file—the**ps** file.

10.3 COG Detector Types

10.3.1 Reaction Detector

The simplest detector type, and often the best, is the REACTION detector. A reaction detector counts the number of collisions (reactions) occurring within a specified volume (REGION) of the user's geometry. This gives an estimate of the **collision density Π** , averaged over the detector's volume. The **particle flux Φ** is obtained from the relationship:

$$\Pi = \Phi \mu$$

where μ is the total material cross section (in units of inverse length). At each collision, a particle's contribution df to the flux score is:

$$\delta\Phi = \frac{w}{\mu V} ,$$

where w is the particle's weight and V is the detector volume.

Properly normalized, the sum of the contributions for all particles is the particle number flux averaged over the volume (average number of particles/cm²-source particle). For thick regions this method works well, but for thin regions, where the expected number of collisions is < 1 , this method yields large errors.

COG uses a variation of this method to improve the accuracy for thin regions. Rather than summing the flux contributions df at each collision site, it uses the following procedure. When a particle enters the detector volume, COG computes the flux contribution from an estimated collision. The probability that the particle will collide at least once in its flight through the detector is:

$$\text{Pr(at least one collision)} = 1 - e^{-\mu d} ,$$

where d = distance to the nearest exit surface of the detector. COG then uses for the flux contribution df :

$$\delta\Phi = \frac{w[1 - e^{-\mu d}]}{\mu V}$$

This quantity tends to the standard formula for μd large, but gives improved results for thin volumes. Each particle produces a usable score to sum, whether it actually reacts in the volume or not. Summing the few real collisions in thin volumes would be much less accurate.

Volumes with zero cross sections are handled by calculating the average path length through the volume. This produces an accurate flux estimate. (Reference 30 provides theoretical details of this and other methods of flux estimation.)

The REACTION detector works best for volumes large enough to have many particles going through them. If the detector volume is small and very few particles transverse it, the statistics will be poor and the results may be very unreliable. This latter problem can be alleviated by modifying the random walk to force particles toward the detector volume. In any case, even for very small volumes, the statistics properly reflect the real events being measured.

The REACTION detector is specified by:

REACTION { LUCKY } *reg-ID# volume-size*

where:

{ LUCKY } is an optional input causing the code to operate in the "Lucky Particle" mode (see the section **Lucky Particle Detection**);

reg-ID# is the REGION number assigned to the detector volume. By default, the ***reg-ID#*** of any specified volume (SECTOR) is the same as the Sector number (***sect-ID#***). This can be changed by an ASSIGN statement;

volume-size is the size of the detector region's volume, in units of length³. The unit of length may be specified in the BASIC Data Block, or allowed to default to cm.

As mentioned above, the detector results, if a response function is not specified, will be given in units of particles/cm², regardless of the problem's unit of length.

Note that the detector region does **not** have to be a single, continuous volume, but may be several, possibly disjoint, volumes. The calculated result, in this case, is the average over all these separate volumes. You can use an ASSIGN statement to assign one REGION number (***reg-ID#***) to many SECTORS.

Example: Define a REACTION detector corresponding to REGION 5 with volume 158.4. If no problem unit of length was specified in the BASIC Data Block, the volume is in units of cm³.

```
REACTION 5 158.4
```

Output Units: REACTION, BOUNDARY-CROSSING and POINT detectors will calculate the number of particles per square **centimeter**, (and per **second** for a STEADY state source)—regardless of the input units of length and time. The standard output for energy is **MeV**. These detector output units are **not** changed by units specified in the BASIC Data Block.

10.3.2 Boundary-Crossing Detector

A BOUNDARY-CROSSING detector calculates the average particle number flux at a boundary between two adjacent REGIONS. Particle flux is defined as the number of particles traversing unit boundary area normal to the trajectory, per unit time. Each particle crossing the boundary, from any direction, makes a positive flux contribution df :

$$\delta\phi = \frac{w}{A |\cos(\phi)|} ,$$

where w is the particle's weight, A is the boundary area, and q is the angle between the particle's velocity vector and the boundary normal.

One way of looking at a boundary-crossing detector is to take the reaction detector and shrink the volume, so that it has zero thickness in one of the dimensions. The resulting score is then the incident particle weight divided by the projected area of the surface being crossed, where the projected area includes the cosine of the angle between the particle direction and the normal to the surface. Unfortunately, the cosine tends to zero for the very few particles that have a grazing intersection with the surface. To prevent this from causing computational problems, any time the cosine is less than 0.01 it is set to 0.005. There is some rationale for doing this, but it is only valid for certain geometries and not in general. Usually few, if any, particles cross the surface with a grazing angle. If you have a problem where grazing crossings may be a problem, we suggest you use a reaction detector.

The boundary-crossing detector is specified by:

[BOUNDARY – CROSSING]
BOUNDARY {LUCKY} {COUNTS} {CURRENT}

{CENSUS} *reg-ID#₁* *reg-ID#₂* *area*

where:

{LUCKY} is an optional input causing the code to operate in the "Lucky Particle" mode (see the section **Lucky Particle Detection**);

{COUNTS} is an optional keyword which causes this detector to score particles instead of particle flux. Use this option if you simply wanted to know how many particles cross this boundary;

{CURRENT} is an optional keyword which causes this detector to score current density instead of particle flux. Current density is defined as the directional flux crossing the boundary surface projected along the normal to the detector boundary, in units of particles per unit area. A particle crossing from *reg-ID#₁* to *reg-ID#₂*

makes a positive contribution to the net current density; a particle crossing in the reverse direction makes a negative contribution.

{**CENSUS**} is an optional keyword which causes the parameters of scoring particles to be written into a text file. (This file can be used as a Source file in a subsequent run. See the **CENSUS Source File Option** in the **SOURCE Data Block** for a complete description of this feature.;

reg-ID#₁ reg-ID#₂ are the REGION numbers of two adjoining regions (the detector BOUNDARY is the surface between them);

area is the detector surface area, in units of length². The unit of length may be specified in the BASIC Data Block, or allowed to default to cm.

The detector results, if a response function is not specified, will be given in units of particles/cm², regardless of the problem's unit of length.

***Example:** Define a BOUNDARY-CROSSING detector which scores **flux** on the surface between REGIONs 1 and 2. The detector area = 12.566. If no problem unit of length was specified in the BASIC Data Block, the area is in units of cm².*

```
BOUNDARY 1 2 12.566
```

***Example:** Define a BOUNDARY-CROSSING detector which counts **particles** crossing the surface between REGIONs 1 and 2. The detector area = 12.566.*

```
BOUNDARY COUNTS 1 2 12.566
```

Output Units: REACTION, BOUNDARY-CROSSING and POINT detectors will calculate the number of particles per square **centimeter**, (and per **second** for a STEADY state source)—regardless of the input units of length and time. The standard output for energy is **MeV**. These detector output units are **not** changed by units specified in the BASIC Data Block.

10.3.3 Point Detector

General Considerations

The REACTION detector measures the average flux within some specified volume, while the BOUNDARY-CROSSING detector measures the flux averaged over some specified surface area. To obtain an answer at a point in space—i.e. one that is not averaged over some volume or surface area, we have to use a much more complicated procedure known in COG as a POINT detector. This has been called a next-flight estimator or a last-flight estimator at various times in the literature.¹ This method estimates the flux at a given point by summing the contributions from each source point and from each scattering event. In effect, we calculate the probability that a particle of the desired type emerges from the source/collision point and flies in a straight line to the point detector, without ever colliding in the intervening geometry. Each differential contribution is of the form

$$d\Phi = S(\mathbf{r}') e^{-mfp}/|\mathbf{r} - \mathbf{r}'|^2$$

where $S(\mathbf{r}')$ is the flux at the source/collision point $\mathbf{r}'$ in space, directed toward the detector position $\mathbf{r}$. For a true particle source, $S(\mathbf{r}')$ is normally easy to compute. For a scattering source, $S(\mathbf{r}')$ involves the probability of scattering through an angle defined by the initial direction of the particle and the direction toward the point detector.

There is, of course, an energy associated with this scattering source. mfp is the number of mean free paths that the scattered or source particle must pass through in order to travel from the source/collision point to the detector. $|\mathbf{r} - \mathbf{r}'|$ is the distance between the source/collision point and the detector.

The energy-dependent flux at the point detector is the integral, over all phase space, of $d\Phi$. This is a form of the integral Boltzmann equation. This is a six-fold integral and can best be evaluated by the Monte Carlo integration method. Thus a Monte Carlo particle transport simulation is combined with a Monte Carlo integral calculation to compute results at a point detector. The random walk scattering points are the sample points taken to evaluate the integral. The normal process of simulating particle transport by Monte Carlo is, in part, short circuited. Actual transport of a particle from the source/collision point to the detector is replaced by a calculation of the potential contribution of these "sources" to the detector's response. Tracking between source/collision point and point detector may be thought of as via "pseudoparticles" that undergo no further collisions, but whose "weight" is reduced by the solid angle and transmission probabilities.

In addition to the usual random walk calculation, several other calculations are needed to solve this equation. Each of these calculations involves going to the cross section data sets and evaluating the probability of scattering in a given direction,

analyzing the geometry to find the distance through each material between two points, and then calculating the transmission mean free path. These additional calculations are time consuming—so much so that the ordinary random walk computing time can almost be neglected by comparison.

However, this additional computing time delivers two very attractive advantages. The first is that we really do get an estimate of the flux at a point, regardless of where it is located. The second is that we get the maximum amount of data out of each collision and each source particle generation. For some shielding problems, the point detector is the most effective and efficient method of obtaining a result. Indeed for deep penetration problems, this is one of the few methods that can provide a meaningful answer.

The point detector technique also has some limitations of which the user should be aware. The main limitation is that the variance of this estimate is infinite, which leads to a slower convergence of the average score to the expected value as the number of histories n is increased. (It has been shown that the rate of convergence varies as $n^{-1/3}$, compared to the $n^{-1/2}$ convergence rate of the Reaction and Boundary detector types, to which the classical Central Limit Theorem applies). This does not mean that the estimate is unreliable; the expectation of the point flux estimator is the true flux at the point. But the distribution of point detector scores is usually far from a normal distribution. Scores below the expected value are more likely than scores above. This means that the point detector tends to underestimate the true flux. The expected (true) value is obtained in the long run by adding in the contributions from a few large scores. The occasional large contribution comes from a particle which scatters in the vicinity of the point detector, where the

$1/|\mathbf{r} - \mathbf{r}'|^2$ factor is large.

Because of this limitation and the fact that this hybrid calculation is only a simulation of a true random walk problem, point detectors should only be considered if an answer cannot be obtained by other, more conventional methods.

Warning: In particularly difficult problems, the point detector technique is often combined with source biasing and random walk biasing in order to get an answer in a reasonable amount of computer time. But deviating from analog Monte Carlo transport (no biasing) increases the risk of errors in the point detector results. *The POINT DETECTOR method can yield results that are orders of magnitude away from a correct value if improperly used; the stated statistical error can also be grossly wrong* (see the section **COG Statistical Considerations: The Problem of Under Sampling**). These anomalous results nearly always arise from failure to sample the problem's phase space adequately. For example, a user may beam all the source particles to a particular spot in the problem, believing this is the most likely

path to get signals from source to detector. All of the point detector scores obtained with this "biased" source may be very uniform, giving a result with a small standard deviation. But it could happen that one or more unconsidered transport paths would have provided much more detector flux than did the chosen path, thus rendering the calculation useless (and misleading).

Point detector results will often have statistical errors that are very large. This does not necessarily invalidate the answer. In many cases, the result is reasonable and the statistical errors have to be accepted. Sometimes knowing the correct order of magnitude is all that is required. Nearly all the large statistical errors we have seen have been the result of under sampling rather than collisions close to the detector. If a particular particle or a few particles are contributing nearly all the answer, these particles' paths should be investigated (using RETRACE) to see how you might possibly get more particles into these high scoring locations.

Point Detector

A POINT detector (point-flux estimator) is specified by:

POINT $x y z$

where:

$x y z$ is the detector position. Units are in centimeters unless specified otherwise in the BASIC Data Block.

As we discussed above, there are at least two major concerns about the use of the POINT detector. One is that some (or many) parts of phase space may not be sampled enough or at all. Another concern is that collisions close to the detector can produce large single particle contributions to the result. Both of these problems may, in turn, produce a very large statistical error.

To provide additional information to the user about both these potential problems, COG lists the percentage of the total POINT detector answer that came from scattering in every problem REGION. In effect, this breaks the integral response into spatial parts and tells you what is going on in each part (region). One of these regions will include the POINT detector and these results will indicate the effect of close-by scattering. Regions with few, or no, scattering will warn you of problems in under sampling phase space. We show below a COG scattering-analysis table, taken from a sample problem. The term fsd is the **fractional standard deviation** associated with the response. This is one standard deviation divided by the response.

TOTAL DETECTOR RESPONSE LISTED BY THE STATISTICAL
REGION IN WHICH SCATTERINGS OCCURRED

region	particles	source + collisions	%response	fsd
4	5000	7998	56.261099	.008
5	5000	338	18.311108	.215
2	5000	822	16.452030	.134
1	5000	27875	7.488245	.026
6	5000	213	1.110742	.206
3	5000	221	0.270175	.307

You can use this REGIONal breakdown to improve the efficiency of subsequent runs. It frequently occurs, as in this example, that some regions contribute almost nothing to the point detector results, while some regions have not been sampled enough to know what they contribute.

In this case, the 7998 collisions in REGION 4, which included the fixed source, yielded ~56% of the answer. This represents ~21% (7998/37467) of the total collisions and, hence, ~21% of the time spent calculating the POINT detector results. The error associated with this calculation is very small, and one would almost always be satisfied with it. The scattering in REGION 1, however, used ~74% of the calculational effort and produced only ~7% of the result. It would have been more productive to have sampled this region much less often. REGIONs 5 and 2 had only ~3% of the collisions (effort) and produced ~35% of the result—these regions were under sampled. To run more efficiently, we would like to reduce the number of point-flux estimates made from REGION 1, and concentrate on the other regions.

The following optional POINT detector specification will LIMIT the number of point flux estimates to be made from any specified region.

LIMIT $\left[\begin{array}{c} \text{REGION} \\ \text{REG} \\ \text{R} \end{array} \right] \left[\begin{array}{c} \text{ALL or A} \\ \text{reg\_ID\#} \\ \text{reg\_ID\#1 TO reg\_ID\#2} \end{array} \right] n$
 REGION . . .
 . . .

Where:

ALL or A indicates all REGIONS are to be limited;

reg-ID# is the number of the single REGION to be limited;

reg-ID#₁ TO reg-ID#₂ indicates REGIONS numbered *reg-ID#₁* up to and including *reg-ID#₂* will be limited;

n is the number limit which to be applied to the selected region(s). These specified regions will have point flux estimates made for only the first *n* source particles.

If a region is specified more than once, the last specified *n* value overrides any previous value. This method has the effect of limiting the point flux estimations within a specified region to the first *n* source particles generated. COG will scale the results from each region so that an unbiased total result is still obtained. Needless to say, *n* must be equal to or less than the total source particles to be run in the problem. If no limit is input for a region, *n* is set to the total number of source particles. A value of *n* equal to zero means that no contribution will be calculated from that region and the total result will not include the contribution from that region. ***The use of any LIMIT will not allow the COG run to be terminated by the calculational time limits (see BASIC Data Block).***

We emphasize that the above technique is meant to optimize the calculation for the **total** POINT detector results. If you are interested in, say, TIME-dependent results and specify TIME BINs for the POINT detector, this LIMIT technique could work against you. For example, a particular TIME BIN of interest to you might represent only a very small fraction of the total response. If you altered the problem with the LIMIT method, you could make the result in this TIME BIN worse. To insure that you are doing the correct thing you may want to run the POINT detector with a TIME MASK that gives results only for this one TIME BIN. The COG listing could then guide you towards a more optimum solution for that TIME BIN.

As an aid in understanding the relative contributions of various regions, you can restrict a POINT detector to score only those particles which scatter/originate in designated regions. See Section: **Masks That Limit What a Detector Can See.**

Example: Define a POINT detector at (0.,150.,0.).

```
POINT 0. 150. 0.
```

Example: Define a POINT detector at (202.2303,0. 18.5463). LIMIT the calculations in REGION 1 to the first 5000 source particles, exclude all contributions from REGIONS 2, 3, and 4.

```
POINT 202.2303 0. 18.5463
LIMIT REGION 1 5000
REGION 2 TO 4 0
```

Output Units: REACTION, BOUNDARY-CROSSING and POINT detectors will calculate the number of particles per square **centimeter**, (and per **second** for a STEADY state source)—regardless of the input units of length and time. The standard output for energy is **MeV**. These detector output units are **not** changed by units specified in the BASIC Data Block.

10.3.4 Volume Point Detector

A VOLUME POINT detector is a generalization of the POINT detector to a finite volume. In a standard POINT detector, the scoring location is a point fixed in space. In the generalized VOLUME POINT detector, the scoring point location is randomly chosen within the detector volume, whenever a point-flux estimate is made. The net result of this process is the same as establishing many standard POINT detectors in a volume, then averaging the results.

To use a VOLUME POINT detector, specify:

POINT VOLUME *sect-ID#*

where:

sect-ID# is the number of the SECTOR which is the detector volume.

The specified sector must be a BOX, CYLINDER, or SPHERE.

10.3.5 Pulse Detector

A PULSE detector simulates a counting experiment which uses a detector and a multichannel analyzer. A crystal (or solid state detector or ion chamber or proportional counter) with gamma rays incident on it can emit pulses of light (or electrical pulses) proportional to the amount of energy deposited in the crystal. In a real experiment, the pulse is analyzed to find its peak amplitude (voltage). If this amplitude is greater than the threshold of channel A_n and less than that of channel A_{n+1} , it is counted in channel A_n . If the count is below the lowest channel threshold, no count is recorded.

COG simulates a PULSE detector by analyzing the energy deposited in a specified detector region by each particle, and all of its daughter particles. If the total energy deposited is greater than the detector's threshold energy, the PULSE detector's score is incremented by the particle weight. If the user has specified an ENERGY BIN structure for the detector, then a BIN is selected, based on the total energy deposited, and the BIN value incremented by the particle weight.

Note: The response of this DETECTOR is different from other COG detectors in that it records the *sum of all the energy* deposited in a detector region by a primary particle and its progeny, even if the energy is deposited in several separate events (or even if deposited by several particles, if they are daughters of the primary particle). Other detectors record the energy **per collision** and do not do this summing; consequently their response is different from the PULSE detector response. For example, if one used a REACTION detector instead of a PULSE detector, one would see many more low-energy "counts" that would have been summed into a single higher-energy count by a PULSE detector.

The primary particle may be a neutron or a photon. The PULSE detector is insensitive to particle type. That is, if the total energy deposited in the detector by any primary particle and its secondaries (whether neutrons or photons) exceeds the specified threshold, then a count is tabulated in the energy bin containing the total energy deposition. A few examples may make this concept clear:

- In a photon-only problem, a photon of weight 1 and energy 2 MeV deposits all its energy in the pulse detector. It is scored as a count of 1 in the energy bin at 2 MeV.
- In a photon-only problem, a photon of weight 1 and energy 2 MeV has a pair-production event in the detector. Both annihilation photons escape the detector without losing any energy. It is scored as a count of 1 in the energy bin at $0.978(=2 - 2*0.511)$ MeV.
- In a neutron-only problem, a neutron of weight 1 and energy 5 MeV has several elastic-scattering events in the detector, followed by an (n,g) event (Q for this event is

+6.1 MeV). It is scored as a count of 1 in the energy bin at $11.1 (= 5 + Q)$ MeV. (Photons deposit their energy immediately in a neutron-only problem.)

- In a neutron/photon problem, a neutron of weight 1 and energy 5 MeV has several elastic-scattering events in the detector, followed by an (n,g) event (Q for this event is +6.1 MeV). The photon escapes the detector without losing any energy. It is scored as a count of 1 in the energy bin at 5 MeV.

Note: Only a unity response function (see below) should be used with this detector type.

The PULSE detector is specified by:

PULSE [reg-ID#, threshold]

where:

The square brackets [] are required;

reg-ID# specifies a REGION number corresponding to the physical volume of the detector;

threshold is the energy threshold for the detector.

A count will be registered in the detector if the total energy deposited in the detector by a photon and its daughter particles exceeds **threshold**. The default units of **threshold** will be in MeV unless some other energy unit was specified in the BASIC Data Block.

Experimentalists in the real world often use more complicated pulse detection schemes. Beside the base PULSE counter, they can add one or more counters to provide particle identification or background rejection through counter logic. A count in the base counter will only be accepted if some other events have occurred—for example, if another detector also records an event (coincidence).

COG provides three options to simulate counter logic:

a AND b

where both counters **a** and **b** must have signals to generate a count in the base counter;

c OR d

where either counter **c** or **d** must have a signal to generate a count in the base counter;

e NOT f

where a count will be generated in the base counter only if a signal occurs in counter **e** but not in counter **f**.

In these more complicated cases, the **base counter** is always the **first counter** mentioned following the PULSE keyword.

***Example:** Define a PULSE detector as REGION 5, with a threshold of 0.5. This detector will register one count for every photon that deposits more than 0.5 MeV in REGION 5.*

```
PULSE [5, 0.5]
```

***Example:** Define a PULSE detector as REGION 5 with a threshold of 0.5, and REGION 7 with a threshold of 0.2. This detector will register one count for every photon that deposits more than 0.5 MeV in REGION 5, the base counter, **and** more than 0.2 MeV in REGION 7.*

```
PULSE [5, 0.5] AND [7, 0.2]
```

Sets of counters can be grouped together to form a composite counter by enclosing them in square brackets.

***Example:** Define a PULSE detector as REGION 5 and **not** REGIONs 7, 8, or 9. This detector will register one count for every photon that deposits more than 0.5 MeV in REGION 5, the base counter, while, at the same time, **not** depositing more than 0.2 MeV in REGIONs 7, 8, or 9.*

```
PULSE [5, 0.5] NOT [[7, 0.2] OR [8, 0.2] OR [9, 0.2]]
```

The energy associated with the PULSE detector is the energy deposited in the base counter. Hence, the ENERGY-dependent differential output is a simple pulse height representation of the results.

***Example:** Define a PULSE detector as REGION 5 with a threshold of 0.5, and REGION 7 with a threshold of 0.2. This detector will register one count for every photon that deposits more than 0.5 MeV in REGION 5, the base counter, and more than 0.2 MeV in REGION 7. Sort the counts into ENERGY BINs that range from 0 to 5 MeV. The energy to be sorted on is the energy deposited into the base counter (REGION 5).*

```
PULSE [5, 0.5] AND [7, 0.2]
BIN ENERGY = PHOTON 0. 1. 2. 3. 4. 5.
```

The BIN results of this example will be counts vs. energy-deposited-in-base-counter.

With the proper choice of regions and options, we can easily simulate simple coincidence, anti-coincidence, etc., experiments. The CASCADE source was

specifically designed to be used with a pulse detector to simulate correlated gamma counting experiments.

Do ***not*** use walk modifications with the pulse detector if they would modify the particle weights after a collision in any of the pulse counters. Source biasing is safe. Other forms of biasing can create errors, depending on the pulse-detector setup. For example, particle splitting could result in one split particle scoring in two coincidence detectors.

10.3.6 Imaging Detector

FORMAT: The following is a generalized template for a j X k pixel JH Image detector (j pixels down X k pixels across) as it might appear in the DETECTOR block of a COG input deck:

```

NUMBER = number
      {TITLE = "title"}
      IMAGING
      MODE dsmode
      REGIONS r1 r2
DETBNDRY xtl ytl ztl
xbl ybl zbl
xbr ybr zbr
MESHSIZE nrows ncols
PARTICLE iptype
{ E-MASK iptype emargv1 emargv2 }
{ T-MASK tmargv1 tmargv2 }
{ A-MASK amargv1 amargv2 }
{ C-MASK cmargv1 cmargv2 }

```

where the required entries are:

```

NUMBER    = detector ID string keyword
'number'  = identification string (≤ 8 char)
IMAGING  = JH Image specification keyword
REGION    = detector region keyword
'r1'      = detector region number 1
'r2'      = detector region number 2
MODE      = scoring mode keyword
'dsmode'  = detector scoring mode:
            'ptcnts' (particle counts [#])
            'sccnts' (scaled counts (USRDRF))
            'ptflux' (particle flux [#/cm^2])
            'scflux' (scaled flux (USRDRF))
            'ptenrg' (particle energy [MeV])
            'enflux' (energy flux [MeV/cm^2])
            'dose91' (radiation dose [mrem])
            'r.vs.e' (response vs energy [response/cm^2])
DETBNDRY  = detector boundary keyword
'xtl, ..' = corner points (tl, bl, br)
MESHSIZE  = detector mesh keyword
'nrows,..' = detector dimensions (j, k)
PARTICLE  = particle type keyword
'iptype'  = incident particle type

```

and the optional entries {} are:

TITLE = detector title keyword
'title' = title string (≤ 80 char)

the following optional quantities are read in the DetMask routines
and checked in the InMask routine before this routine is called

E-MASK = energy mask keyword
'iptype' = incident particle type
'emargv1' = lower (upper) bound
'emargv2' = upper (lower) bound
T-MASK = time (age) mask keyword
'tmargv1' = lower (upper) bound
'tmargv2' = upper (lower) bound
A-MASK = angle (μ) mask keyword
'amargv1' = lower (upper) bound
'amargv2' = upper (lower) bound
C-MASK = collisions mask keyword
'cmargv1' = lower (upper) bound
'cmargv2' = upper (lower) bound

NOTE: All length, energy and time arguments for the USRDET may be entered in user units (the code will handle all conversions).

NOTE: The scoring modes 'ptflux', 'scflux', 'enflux' and 'dose91' are NOT intended to provide "field" estimates of their respective quantities (e.g. 'ptflux' will NOT provide an estimate of the particle flux at a given point in space). These scoring modes record particles crossing through individual pixels in the detector without regard to their angle of incidence (e.g. 'ptflux' records the ACTUAL flux across a pixel [$\#/\text{cm}^2$] - if the pixel is oriented at 90° to the "field" flux, then it will record 0 (just like a real imaging detector would)).

NOTE: The optional detector masks may be entered in any order the user likes; however, if multiple masks of the same type are specified (presumably by accident), the last instance will prevail (no warning of this action will be provided to the user). If no masks are specified, then none will be applied (amazing!).

NOTE: In order to use the angle masking option (A-MASK) reliably, the USRDET imaging detector (defined in the SURFACES and GEOMETRY blocks of the input deck) should consist of a thin box split into two regions by a directed plane (the positive normal of the plane then defines the positive normal of the detector)

EXAMPLE: The following code defines a 16 X 16 pixel USRDET based on a 4" X 4" rectangle oriented parallel to the yz plane and centered on the x axis at -39.3701" from the problem origin. The detector will tally photons (iptype = 7) with energies between 900 and 1100 keV that have undergone ≤ 5 collisions prior to crossing the boundary between regions 301 & 302. The final detector score will be reported in terms of particle counts.

```
number = pbcd01
title = "Pbcd3 USRDET (16 X 16 pixels) (ptcnts)"
  imaging
mode ptcnts $ particle counts [#]
detbndry -39.3701 -2.0000 +2.0000
          -39.3701 -2.0000 -2.0000
          -39.3701 +2.0000 -2.0000
meshsize 16 16 $ (j X k)
particle photon $ photons (only)
  e-mask 900.000 1100.000
  c-mask 0 5
```

OUTPUT: Two output files will be generated upon completion of the run. The first file ('infile'.uout.01) will contain summary information on the detector and the second ('infile'.udet.01) will contain the tab-delimited image data. The detector tab-delimited image data can be viewed and annotated by open source software ImageJ as 'Text Image'. The ImageJ software is available at url <https://imagej.nih.gov/ij/download.html>.

10.3.7 USRDET — User-Defined Detector Subroutine

The standard COG detectors are very versatile, but occasions arise when the user may want something special. For example, suppose you want to score only single-scattered particles, or those which have passed through a particular region of your geometry. Or suppose you would like to use a detector response which is spatially dependent within the detector region, or which has a complicated reaction dependence. The USRDET feature allows you to write your own detector subroutine to do such custom processing.

The shared runtime library capability in UNIX allows the linking-in of library routines at runtime. We exploit this capability to link user-written detector subroutines into the executing COG code. User-written detector subroutines allow the user to define new types of detector response functions and do special processing of scoring particles. Linking at run time keeps the user routines isolated from standard COG. The user never compiles or loads the standard COG routines, but all of the COG routines may be called as needed. In addition to providing the user with a custom detector capability, the USRDET can function more generally as an interface to post-processing applications.

The user must write and compile one or more subroutines and store them in a runtime (shared) library named “libCOGUser.so”. The name of the entry subroutine is specified to COG on a USRDET line in the COG input file. The user's detector routine will be called by COG after the completion of each source particle history, so that the routine can do its special scoring.

The user's detector routine may call other user-written subroutines, and any COG routines as desired. The user may have more than one instance of a given user-specified detector in a problem, by assigning a different detector number to each USRDET instances. More than one named user detector may be used in a problem.

Specifying the USRDET to COG

Each USRDET must be specified in the DETECTOR Data Block of the COG input file. The USRDET specification has the form:

```
USRDET { subname } REGION reg-ID#1 { REGION reg-ID#2 }
      { PASSPVM } { arg1 arg2 arg3 . . . } { FILE filename }
```

where:

subname is the name of the user's detector subroutine. If omitted, it defaults to “usrdet”;

REGION *reg-ID#₁* { **REGION** *reg-ID#₂* } specifies the one (or two) REGION(s) needed to locate the detector (one *reg-ID#* for a Volume detector, two *reg-ID#s* for a Boundary-crossing detector);

arg1 arg2 arg3 . . . are optional values which will be passed on without processing to the USRDET subroutine;

PASSPVM is a control parameter used in parallel computations under PVM or MPI. It signals the COG Master process to expect scoring tallies to be sent back from each Slave process. The default is that no tallies are returned to the Master process. There is a PComm parallel communications package loaded into COG, which can be used to send results from each Slave usrdet, and receive them in the Master usrdet. See the /usrdet directory in the COG distribution for more information.

File *filename* optionally specifies a 32-character or less file name to be passed to the USRDET, perhaps for storing detector results.

Note: unlike most COG input, *filenames* are case sensitive, i.e. they must be typed *exactly* as they appear on the computer. No embedded blanks are allowed. And unlike a string used in a COG title, the filename is *not* surrounded by ".

As with other detector types, the USRDET line is preceded by the line:

NUMBER = det# {TITLE = " . . ."

where:

det# is an integer;

TITLE= specifies an optional ASCII title.

Example: *Specify a USRDET DETECTOR in REGION 5. Let the user subroutine be named myuser1, and the arguments be the detector volume (22.5), a lower and upper energy limit (0.1 and 10.1), a label (MeV) and a filename to write results into.*

```
USRDET myuser1 REGION 5 22.5 0.1 10.1 MEV FILE
MYOUTFILE
```

COG does not interpret the arguments but merely gathers them up into two arrays of ASCII values (character*8) and their corresponding real (real*8) values, and passes them into your USRDET routine (here myuser1).

See the USRDET directory in the COG distribution for instructions on preparing a USRDET. Examples are provided.

10.3.8 Lucky Particle Detection Mode

It frequently happens that one wants to make a parameter study in some limited region of a problem geometry, by trying out different shielding materials, say, or evaluating different detector schemes. If there is a low probability for source particles to reach the region of interest, then a long computer run will be necessary to generate results with good statistics. If you wish to evaluate many problem permutations, then many long runs will be necessary. However, if the changes you are contemplating are small enough that the particle flux reaching the boundary of the region of interest will not be seriously perturbed by your changes, then you can run your studies much faster by using the Lucky Particle/CORRELATED option.

Using the optional keyword **LUCKY** in either the REACTION or BOUNDARY-CROSSING DETECTOR definition causes the code to operate in the "Lucky Particle" mode. In this mode the code writes into the SOURCE output file (the....**sor** file) the starting random number seeds of **only** those particles which reach the detector. Normally the**sor** file (if enabled via the **WRITESOURCE** option in the SOURCE Data Block) contains a set of SOURCE-description parameters and a list of starting random number seeds for each of the NPART particles making up the particle SOURCE.

If the problem is now rerun in CORRELATED mode with the**sor** file produced by the Lucky Particle run, the only particles which will be followed are those which contributed a score to the Lucky Particle detector. All other particle histories are omitted. In this way, for example, detector parameter studies can be done efficiently by running from a Lucky Particle**sor** file over and over again, with different detectors in the location of the original Lucky Particle detector.

Notes:

- In a CORRELATED run, the Lucky Particle mode must be turned off;
- The results of these runs will be highly correlated, because you are using the same set of "lucky" particles every time;
- If the changes from problem to problem are large enough to significantly perturb the flux field, then your results will not be valid.

Limitations: You may have only one Lucky Particle detector per problem. Because Lucky Particle mode causes a**sor** file to be written, it can not be used with an existing**sor** file (e.g., cannot be used in a CORRELATED run).

Example: *We have a BOUNDARY-CROSSING detector specified as the boundary between REGIONS 1 and 2, with area 12.566. Cause it to function in Lucky Particle mode.*

```
BOUNDARY LUCKY 1 2 12.566
```

MASKs are filters which limit detector scoring to particles which fall into specified ranges of particle properties – such as ENERGY, TIME, ANGLE, number of collisions. In activation problems, a half-life mask can be used. A REGION mask is available for POINT detectors only. Masks allow the user to specify important detector properties, such as collimation or energy thresholds.

An ENERGY MASK is specified by:

where:

e_{lower} and e_{upper} are the lower and upper energy limits of the acceptance range.

The mask is used only for the detector with which it is defined. It results in the detector scoring only particles with energies between the given limits. More than one MASK-E statements can be supplied per detector, one for each particle type.

Default: If no mask is defined, then none is applied.

Example: Limit the detector's neutron response to neutron energies between 1 and 5 MeV, and the photon response to energies below 0.511 MeV.

```

MASK-E NEUTRON 1. 5.
      PHOTON 0. 0.511

```

A TIME MASK is defined by:

Where:

t_{lower} and t_{upper} are the lower and upper particle-age limits of the acceptance range.

The mask is used only for the detector with which it is defined. It results in the detector scoring only particles with AGEs between the given limits. A particle's AGE is equal to its time of birth (at the source or at a collision point) plus the travel time to the detector scoring position.

Default: If no mask is defined, then none is applied.

Example: Limit the detector's time response to incident particles with ages between 1×10^{-9} and 5×10^{-9} seconds.

MASK-T 1.E-9 5.E-9

10.4.3 Single-Angle, MASK-A

An ANGLE MASK to limit particle scoring based on a single (polar) angle is specified by:

$$\text{MASK-A} \quad \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \quad \mu_{lower} \quad \mu_{upper}$$

where:

u_{ref} v_{ref} w_{ref} are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

μ_{lower} and μ_{upper} are the lower and upper limits of the acceptance range, in terms of the cosine of the polar angle between the scoring particle's direction and the reference direction.

Default: If no mask is defined, then none is applied.

Example: Limit the detector's angle response to angles between 0° and 10° from the normal. Assume we have a boundary detector.

MASK-A NORMAL 0.9848 1.

10.4.4 Double-Angle, MASK-A*

An ANGLE mask which limits particle scoring based on a polar angle and an azimuthal angle is specified by:

$$\text{MASK-A}^* \quad \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \quad u_a \quad v_a \quad w_a$$

$$\mu_{lower} \quad \mu_{upper} \quad \Phi_{lower} \quad \Phi_{upper}$$

where:

u_{ref} v_{ref} w_{ref} are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

u_a v_a w_a are the direction cosines defining the azimuthal reference direction;

μ_{lower} and μ_{upper} are the lower and upper limits of the acceptance range for the cosine of the polar angle between the particle's direction and the polar reference direction;

Φ_{lower} and Φ_{upper} are the lower and upper limits of the acceptance range for the azimuthal angle (in degrees) between the particle's direction and the azimuthal reference direction.

The two reference directions must be at right angles to each other. Default: If no mask is defined, then none is applied.

Example: Limit the detector's angle response to angles in the +Z, +X quadrant.

```
MASK-A* 0. 0. 1. 1. 0. 0. 0. 1. 0. 90.
```

You cannot use both the MASK-A and the MASK-A* specifications with the same detector.

10.4.5 Isotope, MASK-ISO

An ISOTOPE MASK is specified by:

MASK-ISO *n isotope1 isotope2 ...*

where:

n is the number of isotopes in the list;

isotope1 isotope2 ... are the masking isotopes - given as ZA numbers, e.g., 13027 for Al27. The mask is used only for the detector with which it is defined. It results in the detector scoring only particles whose last event before scoring was in one of the isotopes in the list.

Default: If no mask is defined, then none is applied.

Example: Limit the detector's response to particles whose last event before scoring was in Al27.

```
MASK-ISO 1 13027
```

10.4.6 Reaction, MASK-REA

A REACTION MASK is specified by:

MASK-REA *n reaction1 reaction2 ...*

where:

n is the number of reactions in the list;

reaction1 reaction2 ... are the masking reactions - given as reaction numbers, e.g., 12 for (n,2n), see Reaction Number Listing section for a complete list of reaction numbers. The mask is used only for the detector with which it is defined. It results in the detector scoring only particles whose last event before scoring was one of the reactions in the list.

Default: If no mask is defined, then none is applied.

Example: *Limit the detector's response to particles whose last event before scoring was (n,2n).*

MASK-REA 1 12

10.4.7 Collision, MASK-C

This mask restricts scoring to only those particles which have undergone a specified number of collisions.

A COLLISION mask is specified by:

MASK-C *n1 n2*

where:

n1 and *n2* are the minimum and maximum particle collision numbers allowed to pass the mask. Each particle carries a collision-number counter, which is initially set to zero for source particles and to one for secondary particles born out of a collision. At every subsequent collision suffered by the particle, the collision number is incremented by one.

10.4.8 Region, MASK-REG (Point Detectors only)

This mask restricts POINT detector scoring to only those particles whose source event or last scatter event occurred in a set of specified REGIONS.

A REGION mask is defined by:

MASK-REG *list_spec*

where:

list_spec specifies the REGIONS to be allowed. The list can be specified in various ways, as the following examples illustrate:

MASK-REG ALL

MASK-REG ALL EXCEPT 4 14 98

MASK-REG 30 to 50

MASK-REG 30 to 50 EXCEPT 33 40

MASK-REG 14 22 98 403

10.4.9 Half-Life, MASK-HL (Activation problems only)

Up to this point, we have assumed that when a neutron reacts to produce a secondary photon, the photon exits the reaction without a time delay. Actually, there is always a small time delay, but it is so small that we can neglect it. We have not considered the case where the neutron excites (activates) the nucleus and decay photons are not emitted until days, weeks, or even years after the activating event. These are problems of activation and are not really different from a standard problem, except that the appropriate decay time must be taken into account. A COG problem involving activation must use a special neutron cross-section library, ACTL, rather than the standard ENDL90 or ENDF/B-V neutron libraries. The ACTL library (Ref. 7 of the Introduction) contains secondary production data which include neutron activation and known decay schemes for most radioactive isotopes. A decay constant or half-life is associated with the emission of every secondary photon. To achieve a correct time-dependent result at a detector, the detector results must be sorted by half-life and then multiplied by a time-dependent factor (which includes the period of original irradiation and the time since irradiation).²

The HALF-LIFE MASK filters the detector results based on a specified emission half-life of secondary photons. It has the form:

MASK-HL *half-life*

where:

half-life is the half-life desired, in the units of the reference time. COG limits this detector's result to particles with emission half-lives ranging from 0.975 times ***half-life*** to 1.025 times ***half-life***.

Default: If no mask is defined, then none is applied.

Example: Limit the detector's half-life response to values about 4.6 seconds (4.485 to 4.715 seconds). It is assumed that this is an ACTIVATION problem and the detector is sensitive to photons.

MASK-HL 4.6

10.5 Detector Response Functions (DRFs)

10.5.1 Energy, DRF-E

By default, the REACTION, BOUNDARY-CROSSING and POINT detectors will calculate number flux at the detector –the number of particles per square centimeter. Sometimes this is what the user wants, but in most cases the quantity desired is the detector's response to this number flux. Almost always, the conversion from number flux to the desired quantity is energy dependent.

An ENERGY-dependent detector response is specified by:

DRF-E *particle-type response*
(*particle-type response*) . . .

where:

particle-type is the particle type (must be one of those specified in the BASIC Data Block);

response represents the desired energy response and has one of the forms given below. More than one DRF-E statement can be supplied per detector, one for each particle type.

Default: All ***specified particles*** in the problem will have a response value of ***unity***. If there is more than one type of particle specified in the problem, and if one type has a **DRF-E** given but the other types do not, the ***unspecified particle types*** will have a default response of ***zero***.

response may be any one of the following:

- **NUMBER-FLUX** Yields a response equal to one for all particle energies. Detector results will be given in units of particles/cm².
- **ENERGY-FLUX** Yields a response equal to the incident particle energy, in MeV, at all energies. Detector results will be given in units of MeV/cm².
- **DOSE77** Produces a result in **REM** (Roentgen Equivalent Man). For neutrons with energies from 0 to 20 MeV, the conversion data (to convert fluence to dose) are taken from the ANSI/ANS standard 6.1.1 (1977). From 20 to 200 MeV, the data are taken from Table 1.3-7 of the *Engineering Compendium of Radiation Shielding* (1968). For photons greater than 10 keV, the conversion factors are from the ANSI/ANS standard 6.1.6 (1977). Photons less than 10 keV will use a zero conversion factor and a warning will be printed.
- **DOSE** or **DOSE91** Produces a result in Sieverts (1Sv = 100REM). The results are interpreted as the dose, in Sv/source particle, which human tissue would receive at the

detector position. Neutron and photon Sievert dose data (which convert fluence to dose) are taken from the ANSI/ANS standard 6.1.1 (1991).

•**DOSE59** Produces a result in REM or RAD from XDC 59-8-179, "Conversion of Neutron or Gamma Ray Flux to Absorbed Dose Rate" by B J Henderson, Aug 14, 1959

•**R-RATE** *mat-ID# reaction#*

asks COG to estimate a reaction rate due to the flux of particles of type *particle-type* on a specified problem material, where:

mat-ID# is the MATERIAL number of one of the materials defined in the MIX Data Block, *which must be an isotope*;

reaction# is a specified reaction on the selected isotope, chosen from the Reaction Number Listing section. Note that data for a particular reaction may not be available for all isotopes. Reaction cross sections are taken from the problem's neutron library (nlib).

Results for this response are in units of reactions per unit volume of the specified isotope, per source particle.

•**IRDF-R-R** *mat-ID# reaction#*

asks COG to estimate a reaction rate due to the flux of particles of type *particle-type* on a specified problem material, where:

mat-ID# is the MATERIAL number of one of the materials defined in the MIX Data Block, *which must be an isotope*;

reaction# is a specified reaction on the selected isotope, chosen from the Reaction Number Listing. Note that data for a particular reaction may not be available for all isotopes. Reaction cross sections are taken from the problem's IRDF (International Reactor Dosimetry and Fusion) library (doslib).

Results for this response are in units of reactions per unit volume of the specified isotope, per source particle.

•**ENERGY-DEPOSITION** *mat-ID#*

where *mat-ID#* is the MATERIAL number of one of the materials defined in the MIX Data Block. This option will produce the energy deposited in that material (which should be the actual material in the detector region) by the particle fluence at the detector. Energy deposition is given in units of MeV/cm³. This calculation is made using data available on the EPDL and ENDL cross section libraries. EPDL is COG's default photon library (COGGXS). *Neutron energy deposition is not available from the ENDF (ENDF/B-V,-VI,-VII,-VIII) cross section libraries.* The required data consist of two parts: 1) for each reaction, the average energy of the residual nucleus versus the incoming particle energy; and 2) for each reaction and for each outgoing particle type,

the average energy of the scattered particle versus incoming particle energy. The local energy deposition is the sum of the average energy of the residual nucleus and the average energy of the scattered particle, for those particle types which are not being transported. The cross-section weighted average of the local energy deposition produces the (energy-dependent) energy deposition response.

• **E-PRODUCTION *mat-ID#***

Specifies the Electron Production response, where *mat-ID#* is the MATERIAL number of one of the materials defined in the MIX Data Block. This causes COG to calculate the number of electrons produced in that material (which should be the actual material in the detector region) by photon interactions. The ENERGY, TIME and ANGLE results for this detector are given for the electrons produced. Note that *no* electrons are transported when using this response and any secondary photons and electrons generated by these electrons will *not* be included. This option provides first estimates for those who wish to calculate detector sensitivities.

• **EE-PRODUCTION *mat-ID#***

Specifies the Electron Energy Production response, where *mat-ID#* is the MATERIAL number of one of the materials defined in the MIX Data Block. This causes COG to calculate the electron energy created in that material (which should be the actual material in the detector region) by photon interactions. The ENERGY, TIME and ANGLE results for this detector are given for the electrons produced. Note that *no* electrons are followed when using this response and any secondary photons and electrons generated by these electrons will *not* be included. This option provides first estimates for those who wish to calculate detector sensitivities.

• Alternatively, the desired response function can be specified as a series of **energy, response** pairs (a distribution):

$$e_1 \text{ response}_1 \ e_2 \text{ response}_2 \ . \ . \ .$$

where:

response₁ is the detector response for incident particle energy *e₁*, etc. The detect or response is assumed to vary linearly between the *e_i*'s and to be zero above and below the specified range.

Example: Define a detector response to be neutron energy flux.

DRF-E NEUTRON ENERGY-FLUX

Example: Define a detector response to be neutron and photon dose.

```
DRF-E NEUTRON DOSE
      PHOTON DOSE
```

Example: Define a photon energy response as a point-wise distribution. The response is 2.1×10^{18} at 1. MeV, changing linearly to $4. \times 10^{18}$ at 2.5 MeV, and again changing linearly to 5.1×10^{17} at 10. MeV. The response is zero below 1. and above 10. MeV.

```
DRF-E PHOTON 1. 2.1E18 2.5 4.E18 10. 5.1E17
```

10.5.2 Secondary Energy Detector Response Function, DRF2-E

The secondary detector energy response is specified by...

DRF2-E *particle-type response*

[*particle-type response...*]

where *particle-type* is the particle type and *response* is one of the following:

- **polynomial c0 c1 c2 ...**

***response* = c0 + c1*energy + c2*energy*energy ...**

limited to **c5**, i.e., 5th order

- **histogram e1 r1 e2 r2 e3 ... e#**

***response* = r1 if e1 < energy < e2**

r2 if e2 < energy < e3

...

limited to **e101**, i.e., 101 energies

- **distribution e1 r1 e2 r2 ... e# r#**

***response* = linear fit to e1 r1 e2 r2 if e1 < energy < e2**

linear fit to **e2 r2 e3 r3** if **e2 < energy < e3**

...

limited to **e101**, i.e., 101 energies

Example: Define a neutron energy response as number-flux. Apply a secondary energy detector response function, polynomial 0. 1., effectively converting the total response to energy-flux.

```
DRF-E NEUTRON NUMBER-FLUX
DRF2-E NEUTRON POLYNOMIAL 0. 1.
```

10.5.3 Time, DRF-T

A detector response as a function of time may be provided by specifying:

DRF-T t_1 *response*₁ t_2 *response*₂ . . .

where:

t_i *response* _{i} are the point-wise **time, response** pairs defining the response. The response is assumed to vary linearly between the t_i 's and to be zero above and below the specified range.

Default: If no **DRF-T** is given the problem will have a time response of **unity**.

Example: Define a time response which is 2.1 at $1.X10^{-9}$ seconds, changing linearly to 4. at $2.5X10^{-9}$ seconds, and again changing linearly to 0.5 at $1.X10^{-8}$ seconds. The response is zero below $1.X10^{-9}$ and above $1.X10^{-8}$ seconds.

DRF-T 1.E-9 2.1 2.5E-9 4. 1.E-8 0.5

10.5.4 Single-Angle, DRF-A

ANGLE-dependent detector response functions are very similar in form to ANGLE-dependent source specifications. For a response dependent upon only one polar angle measured from a polar reference direction, specify:

$$\text{DRF-A} \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \mu_1 \text{ response}_1 \mu_2 \text{ response}_2 \dots$$

where:

$u_{ref} \ v_{ref} \ w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

$\mu_i \ \text{response}_i$ are the point-wise μ , *response* pairs defining the response. μ is the cosine of the angle between the particle's direction and the polar reference direction. The response is assumed to vary linearly between the μ_i 's and to be zero above and below the specified range.

Default: If no **DRF-A** or **DRF-A*** is given the problem will have an angle response of *unity*.

Example: Define an angle response which is constant between 0° and 10° , as measured from the normal (this assumes a boundary detector).

```
DRF-A NORMAL 0.9848 1. 1. 1.
```

You cannot use both the DRF-A and the DRF-A* specifications with the same detector.

10.5.5 Double-Angle, DRF-A*

For a detector response dependent upon polar angle and an azimuthal angle, specify:

$$\text{DRF-A*} \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} u_a \ v_a \ w_a$$

$$\mu_{1-} \ \mu_{1+} \ \phi_{1-} \ \phi_{1+} \ response_1$$

$$\mu_{2-} \ \mu_{2+} \ \phi_{2-} \ \phi_{2+} \ response_2$$

$$\dots$$

where:

$u_{ref} \ v_{ref} \ w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

$u_a \ v_a \ w_a$ are the direction cosines defining the azimuthal reference direction;

$response_i$ is the response in the solid angle bin defined by $\mu_{i-} \ \mu_{i+} \ \phi_{i-} \ \phi_{i+}$. The μ 's are the cosines of the angle between the particle's direction and the polar reference direction, and the ϕ 's are the azimuthal angle (in degrees) between the particle's direction and the azimuthal reference direction.

The two reference directions must be at right angles to each other.

Default: If no **DRF-A*** or **DRF-A** is given the problem will have an angle response of unity.

Example: Define an **ANGLE** response which is uniform in the +Z, +X quadrant and zero elsewhere.

$$\text{DRF-A*} \ 0. \ 0. \ 1. \ 1. \ 0. \ 0. \ 0. \ 1. \ 0. \ 90. \ 1.$$

You cannot use both the DRF-A and the DRF-A* specifications with the same detector.

COG Standard Double-Angle Bin Structure

A double-angle response function can alternatively make use of the COG pre-defined equal-solid-angle BIN structure. This is the same structure which may be used in a COG SOURCE or a detector angle differential response. (See the section **COG Standard Double-Angle Bin Structure**, under **Sources**, for the definition of this bin structure.)

DRF-A*

$$\begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \quad u_a \quad v_a \quad w_a$$

$$N = n \quad response_1 \quad response_2 \quad \dots \quad response_m$$

where:

$u_{ref} \quad v_{ref} \quad w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

$u_a \quad v_a \quad w_a$ are the direction cosines defining the azimuthal reference direction;

N is the keyword indicating that the COG pre-defined equal-solid-angle BIN structure is desired;

n is an integer denoting the order of the BIN structure;

$response_i$ is the response in each of the solid-angle BINs.

The two reference directions must be at right angles to each other.

There will be $m = 4n(n + 1)$ equal-solid-angle BINs

10.6 BINs – Obtaining Differential Detector Results

Frequently users want to see their detector results broken down by ENERGY, TIME, and/or ANGLE. In COG, we refer to these as the **differential results** for each detector. You obtain differential results for a detector by specifying one or more BIN structures in ENERGY, TIME, and/or ANGLE. You can even specify more than one set of ENERGY BINs, say, so that you can have both a coarsely-gridded and a finely-gridded spectrum.

BIN edges can be specified in either increasing or decreasing order, and need not cover the entire parameter range.

10.6.1 ENERGY Bins

ENERGY BINS are specified by:

$$\text{BIN } \begin{bmatrix} \text{ENERGY} \\ \text{E} \end{bmatrix} = [\text{particle\_type}] \ e_1 \ e_2 \ e_3 \ \dots$$

$$\{ \text{LIST E } e_{1a} \ e_{1b} \ e_{2a} \ e_{2b} \ \dots \}$$

where:

particle-type is the particle type (must be one of those specified in the BASIC Data Block);

e_i are the BIN edges, in MeV (or in the ENERGY units specified in the BASIC Data Block);

LIST E is an option indicating that the user wants to LIST particle properties for those scoring particles which fall into the ENERGY ranges specified by the endpoint pairs $e_{1a} \ e_{1b} \ e_{2a} \ e_{2b} \ \dots$. This is a diagnostic tool which can be used to find out where the particles are coming from that score into a particular part of the detected spectrum. See the section **LISTing Properties of Scoring Particles**, below.

Example: Define a neutron ENERGY BIN structure with 10 bins (11 edges) which range from 0. to 10. MeV in steps of 1. MeV. Note the use of the interpolation feature (in the brackets).

```
BIN ENERGY NEUTRON 11 [0. I 10.]
```


10.6.2 TIME Bins

TIME BINs are specified by:

$$\text{BIN} \begin{bmatrix} \text{TIME} \\ \text{T} \end{bmatrix} t_1 t_2 t_3 \dots$$

where:

t_i are the BIN edges, in seconds (or in the units specified for TIME in the BASIC Data Block).

Example: Define a TIME BIN structure with 10 bins (11 edges) running from 0. to 1×10^{-8} seconds in steps of 1×10^{-9} seconds. Note the use of the interpolation feature (in the brackets).

```
BIN T 11 [0. I 1.E-8]
```

10.6.3 ANGLE Bins - Single Angle

ANGLE BINs that are dependent on only one (polar) angle are specified by:

$$\text{BIN} \begin{bmatrix} \text{ANGLE} \\ \text{A} \end{bmatrix} = \begin{bmatrix} u_{ref} v_{ref} w_{ref} \\ \text{NORMAL} \end{bmatrix} \mu_1 \mu_2 \mu_3 \dots$$

where:

$u_{ref} v_{ref} w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

μ_i are the BIN edges, given in cosine of the angle between the polar reference direction and the particle direction.

Example: Define an ANGLE BIN structure which consists of 10 equally-spaced (in cosine) bins running from -1. to 1. in cosine of the angle between the particle direction and the normal to the detector surface. This example assumes a boundary detector.

```
BIN A NORMAL -1. -0.8 -0.6 -0.4 -0.2 0.
               0.2 0.4 0.6 0.8 1.
```

If desired, the reference direction may also be specified by any method used in the path-stretching or angle-biasing techniques described in the section **The Monte Carlo Random Walk**.

Each of the above BIN structures produces a single list of bin values as a function of just one detector output quantity—that is, a one-dimensional, 1D, output.

10.6.4 Multiple Bins

2D Bins

If we wanted to see how the detector results varied in both TIME and ENERGY, we could specify:

$$\text{BIN} \begin{bmatrix} \text{TIME} \\ \text{T} \end{bmatrix} \quad t_1 \quad t_2 \quad t_3 \quad \dots$$

$$\begin{bmatrix} \text{ENERGY} \\ \text{E} \end{bmatrix} = [\text{particle\_type}] \quad e_1 \quad e_2 \quad e_3 \quad \dots$$

We would now get a set of ENERGY BINs for each TIME BIN. There are many more individual bins for this two-dimensional, 2D, representation, and it may be necessary to run your problem longer to achieve meaningful statistics in every bin. Other cases of 2D BINs would be TIME and single-ANGLE BINs, or ENERGY and single-ANGLE BINs.

A double-ANGLE BIN structure also produces a set of 2D BINs. This is specified by:

$$\text{BIN} \begin{bmatrix} \text{ANGLE}^* \\ \text{A}^* \end{bmatrix} = \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \quad u_a \quad v_a \quad w_a$$

$$\mu_{1-} \mu_{1+} \quad \phi_{1-} \phi_{1+}$$

$$\mu_{2-} \mu_{2+} \quad \phi_{2-} \phi_{2+}$$

$$\dots$$

where:

$u_{ref} \ v_{ref} \ w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

$u_a \ v_a \ w_a$ are the direction cosines defining the azimuthal reference direction;

$\mu_{i-} \mu_{i+}$ are the BIN edges in the cosine of the angle between the polar reference direction and the particle direction;

$\phi_{i-} \phi_{i+}$ are the BIN edges of the angle (in degrees) between the azimuthal reference direction and the particle direction.

The two reference directions must be at right angles to each other.

Example: Define a double ANGLE BIN structure which consists of 8 quadrants. The reference and azimuthal directions are the +Z axis and the +X axis respectively.

```

BIN A* 0. 0. 1. 1. 0. 0.
        -1. 0. 0. 90.
        -1. 0. 90. 180.
        -1. 0. 180. 270.
        -1. 0. 270. 360.
        0. 1. 0. 90.
        0. 1. 90. 180.
        0. 1. 180. 270.
        0. 1. 270. 360.
    
```

If desired, the reference direction may also be specified by any method used in the path-stretching or angle-biasing techniques described in the chapter **The Monte Carlo Random Walk**.

COG Standard Double-Angle Bin Structure

A 2D double-ANGLE BIN structure can alternatively make use of the COG pre-defined equal-solid-angle BIN structure. This is the same structure which may be used in a COG SOURCE or a detector double-angle response function. (See Appendix **COG Standard Double-Angle Bin Structure** for the definition of the bin structure.)

$$\text{BIN} \begin{bmatrix} \text{ANGLE} * \\ \text{A} * \end{bmatrix} = \begin{bmatrix} u_{ref} & v_{ref} & w_{ref} \\ \text{NORMAL} \end{bmatrix} \quad u_a \quad v_a \quad w_a$$

$$\text{N} = n$$

where:

$u_{ref} \ v_{ref} \ w_{ref}$ are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

$u_a \ v_a \ w_a$ are the direction cosines defining the azimuthal reference direction;

N is the keyword indicating the COG pre-defined equal-solid-ANGLE BIN structure is desired; and

n is an integer denoting the order of the BIN structure.

There will be $m = 4n(n + 1)$ equal solid ANGLE BINs. *The two reference directions must be at right angles to each other.*

Example: Use the COG pre-defined double-ANGLE BIN structure of order 3. The polar reference and azimuthal reference directions are the +Z axis and the +Y axis respectively.

BIN A* 0. 0. 1. 0. 1. 0. N 3

3D and 4D Bins

Three-dimensional results can be obtained by specifying any of the following:

BINs in TIME, ENERGY, and polar ANGLE:

BIN T = . . .

E = . . . A

= . . .

BINs in TIME, polar ANGLE, and azimuthal ANGLE:

BIN T = . . .

A* = . . .

BINs in ENERGY, polar ANGLE, and azimuthal ANGLE:

BIN E = . . .

A* = . . .

Four-dimensional results are a combination of T, E, and A* specifications.

10.6.5 Output of Differential Detector Results

The COG differential detector results are listed in the out file along with a complete definition of the bin structure. Binned results have units of response per second, or per MeV, or per steradian for the 1D cases and the correspondingly complex units for the 2D, 3D, and 4D cases.

When COG lists BINned detector results, if space allows, COG also lists the BIN standard deviation, the fractional standard deviation, the contribution made to the response from each BIN, and the running integral response—i.e., total response integrated from the lowest BIN to the current BIN. If the BIN structure includes the full range of the differential variables, the final value of the integral response should equal the total response listed for the detector.

Output Units: REACTION, BOUNDARY-CROSSING and POINT detectors will calculate the number of particles per square **centimeter** (and per **second** for a STEADY state source)—regardless of the input units of length and time. The standard output for energy is **MeV**. These output units are **not** changed by units specified in the BASIC Data Block.

10.7 LISTing Properties of Scoring Particles

Occasionally the user may find that a COG detector energy spectrum contains some surprises. Perhaps mysterious peaks or other features appear in the spectrum. Where do the particles come from that make up these features? COG provides the LIST option to help you find out. For one detector in your problem, you can put, following an ENERGY BIN structure, this statement:

```
LIST E e1a e1b e2a e2b . . .
```

where each energy pair such as *e1a e1b* specifies an ENERGY range to be LISTed (maximum of 20 ranges). The units of energy are always MeV. Every time a particle is scored into one of the ENERGY BINs of your detector, COG checks to see if the scored energy also falls into one of the LIST E ranges. If it does, the scoring particle properties are listed in a file named *...list*. Particle properties LISTed include the particle's sequence number, the scored energy, and the index of the ENERGY BIN in which the particle was tallied. You can use these particle sequence numbers as input in a RETRACE statement for a subsequent COG run. (See section **RETRACE Source Particles**). In the RETRACE mode, COG will print out all the events in the history of each specified particle. This will enable you to determine exactly how these particles came to score in that detector in that particular energy range. **Warning:** for problems with a large number of source particles, the *...list* file can get very long.

***Example:** In a prior example we defined a neutron ENERGY BIN structure with 10 bins which ranged from 0. to 10. MeV in steps of 1. MeV. We now follow this BIN description with a LIST statement to list the parameters of all scoring particles which fall in the ranges: [0.0,0.2] MeV, [0.33,0.37] MeV, and [0.85,0.95] MeV.*

```
BIN ENERGY NEUTRON 11 [0. I 10.]
LIST E 0.0 0.2 0.33 0.37 0.85 0.95
```

10.8 Tagged List-Mode Detector

The OUTFILE option creates a secondary output file listing the time, energy, and score of individual scoring events. This option is invoked by including in the DETECTOR description the line *outfile*. For example:

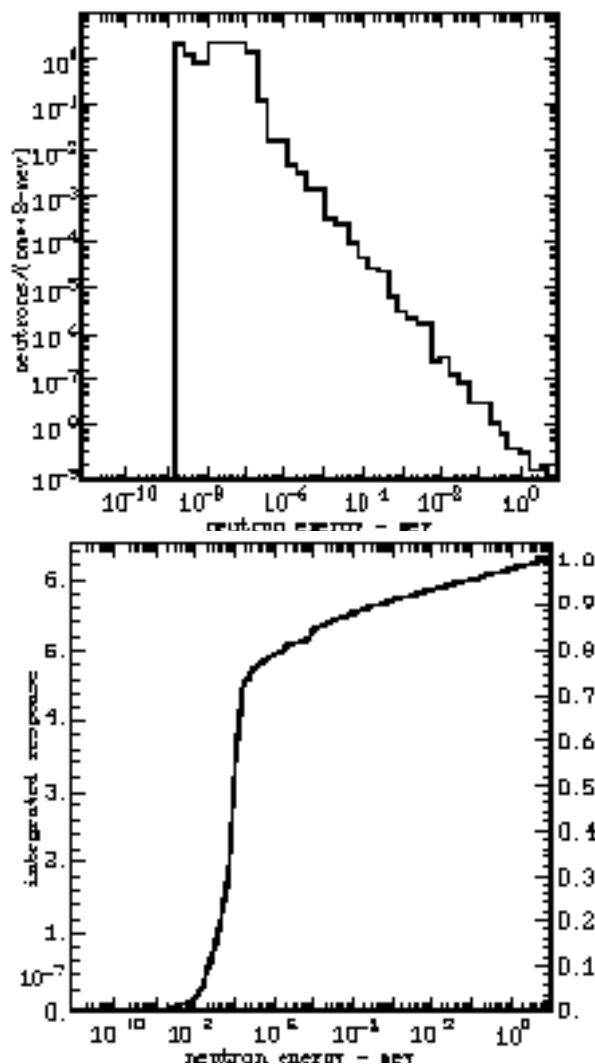
```
DETECTOR
number 1
reaction 2 1.
mask-rea 1 40 $ scores only (n,pγ) reactions
outfile
```

In this example, COG will create an additional output file with the suffix '.outfile' listing the time, score, energy, position (as coordinates x, y, z), direction (as direction cosines u, v, w), and detector# of each (n,pγ) reaction scored by detector 1.

10.9 Plotting Detector Differential Results

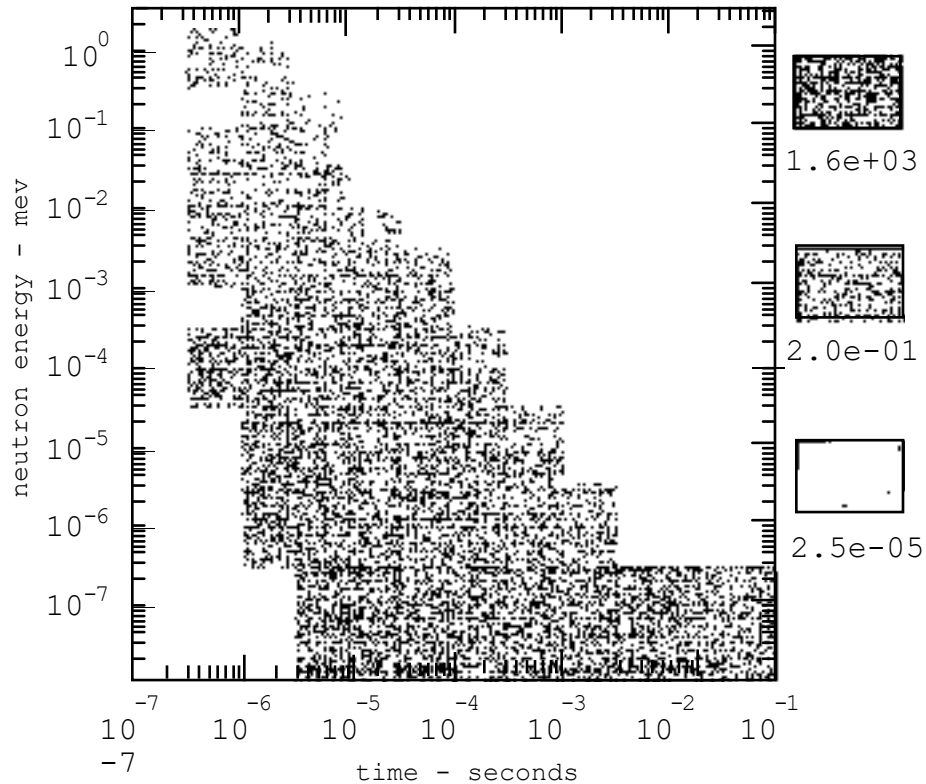
COG has a basic graphics package for plotting detector results, including some user controls over plot scales and labels. Much more flexibility in the data presentation can be achieved by reading the ...**det** file of detector results into your favorite plotting routine.

Each 1-D set of **differential results** (BIN structure) specified for a DETECTOR will generate three plots in the**ps** graphics file. The first plot is the detector response as a function of the differential (BINned) quantity. The second plot shows the same data, with the addition of dotted lines to indicate both plus and minus one standard deviation. The third plot is the running integral response. The first and third plots are illustrated below.



The plotted values should not be considered to be exact, since the calculated values used to make these plots come from a histogram and are not point-wise values.

Each set of 2-D **differential results** (BIN structure) specified for a DETECTOR will generate a plot in the ps graphics file. The plot is a density representation of the results. This may show the histogram nature of your differential BIN structure. An example is given below.



For 3D and 4D differential output, users will have to design their own plotting schemes. For this reason, and to accommodate those who wish to do other things with the output, all detector results are written into an ASCII file, ...det. Viewing this file along with the regular COG output, ...out, will readily identify the quantities contained therein.

10.9.1 Changing Detector Plot Scales/Labels

The COG plots of **differential results** (BIN structures) always use units of **seconds** for time and **MeV** for energy. You may wish to change these units to get a better presentation. You may also wish to change the default limits of the COG detector plots or replace the default plot labels. You do this by specifying, in the lines following a BIN definition, the plot items you wish to change.

To change a plot's **scale factors**, use:

$$XCONV = \textit{scale-factor}$$

where:

X is a letter which designates the physical quantity to be scaled before plotting.

X is one of:

T	Time
E	Energy
P	Cosine of polar angle
A	Azimuthal angle
Y	Differential response (always the plot y-variable)

This *X* convention is used throughout this section.

scale-factor is a number which multiplies the designated values before printing or plotting. It is used primarily to convert units on plots.

Example: We have specified a detector *ENERGY* response, and set up a *TIME BIN* structure. We now want to scale our *TIME BINS* so the results will be plotted in nanoseconds rather than in seconds. We also want our energy results to appear in keV rather than MeV.

```
TCONV 1.0E+09  YCONV 1000.
```

To change a plot's **minimum axis value**, use:

$$XMIN = \textit{min-value}$$

where:

min-value is the desired **minimum axis value**.

To change a plot's **maximum axis value**, use:

$$XMAX = \textit{max-value}$$

where:

max-value is the desired **maximum axis value**.

Example (continued): We want our time axis to run from 0. to 100. ns, and our energy axis to run from 0.1 to 20. keV.

```
TMIN 0.  TMAX 100.  YMIN 0.1  YMAX 20.
```


To use a **logarithmic plot axis**, specify:

XLOG

To use a **linear plot axis**, specify:

XLIN

To use your own plot **label along an axis**, specify:

XLABEL = "...your plot label goes here..."

***Example (continued):** We want our energy axis to be plotted on a log scale and labeled appropriately.*

YLOG YLABEL = "Energy (keV) "

The plot-control data is entered after the last entry defining a BIN structure (if it applies to that structure) or after a BIN definition (if it applies to the response).

The BIN structure can be **titled** by specifying:

TITLE = "...your plot title goes here..."

***Example:** We have a signal and a background detector, both with photon energy-flux responses. The results are specified to be differential in time, differential in energy, and doubly-differential in time and energy.*

```
DETECTOR
NUMBER SIGNAL
TITLE "SIGNAL"
POINT 0. 0. 151.
DRF-E PHOTON ENERGY-FLUX
BIN T 10 [1.E-7 I 1.E-6]
BIN E PHOTON 0. 0.1 0.5 1. 2. 3. 4. 5.
BIN T 10 [1.E-7 I 1.E-6]
      E PHOTON 0. 0.1 0.5 1. 2. 3. 4. 5.

NUMBER BKGRND
TITLE "BACKGROUND"
POINT 5. 0. 151.
DRF-E PHOTON ENERGY-FLUX
BIN T 10 [1.E-7 I 1.E-6]
BIN E PHOTON 0. 0.1 0.5 1. 2. 3. 4. 5.
BIN T 10 [1.E-7 I 1.E-6]
      E PHOTON 0. 0.1 0.5 1. 2. 3. 4. 5.
```

Example: Apply re-scaling and re-labeling options to a differential detector result.

```
DETECTOR
NUMBER 1
TITLE "AVERAGE ENERGY FLUX IN LEAD RING"
REACTION 28 197.3
MASK-T 0. 1.E-6
DRF-E PHOTON ENERGY-FLUX
BIN T 10 [1.E-7 I 1.E-6]
TITLE "AVERAGE ENERGY FLUX VS TIME"
TCONV 1.E6
TMIN 1.E-7 TMAX 1.E-6
TLOG YLIN TLABEL "TIME (MICROSECONDS) "
YLABEL "PHOTONS/CM^2-SEC"
```

Note: The **standard plot output** for time has units of seconds and the **standard plot output** for energy is MeV. These are **not changed** by units specified in the BASIC Data Block. If you want other units on the plots, use the **XCONV** command.

10.10 ANALYSIS Data Block

10.10.1 Plots of Collision Sites

Although the ANALYSIS plot is, strictly speaking, not one of the detector plots, this feature produces plots which can be very helpful in understanding where the scatterings are occurring in your problem which contribute to your detector results. The ANALYSIS plot shows the locations of collisions which occur in a specified slab volume in your geometry. Each picture can be restricted to show only those colliding particles which lie within specified energy, age, and/or weight limits. The ANALYSIS plot helps you see qualitatively what is happening during the random walk process in regions of interest to you. If you modify SOURCE IMPORTANCES, or use a random-walk biasing technique, the ANALYSIS plot can monitor the effects of these changes on the distribution of scattering events in your problem. *ANALYSIS is a separate Data Block, not a part of the DETECTOR Data Block.* An ANALYSIS Data Block is specified as follows:

ANALYSIS

[NEUTRON / PHOTON]

x_{tl} y_{tl} z_{tl} x_{bl} y_{bl} z_{bl} x_{br} y_{br} z_{br}

thickness

{TITLE = ". . ."}

{ENERGY = e_{lower} e_{upper} }

{AGE = age_{lower} age_{upper} }

{WEIGHT = $weight_{lower}$ $weight_{upper}$ }

where:

ANALYSIS begins this Data Block (may be given only once).

For *each* ANALYSIS picture desired, specify the following information:

NEUTRON or **PHOTON** is the particle-type to be plotted;

x_{tl} y_{tl} z_{tl} x_{bl} y_{bl} z_{bl} x_{br} y_{br} z_{br} are the top left, bottom left, and bottom right corners of the desired picture;

thickness is the thickness of the rectangular volume slice in which scatterings will be tallied for this picture. The volume extends from a plane which lies a distance *thickness*/2 below the picture plane, to a plane which lies a distance *thickness*/2 above the picture plane;

TITLE is the optional ANALYSIS picture title;

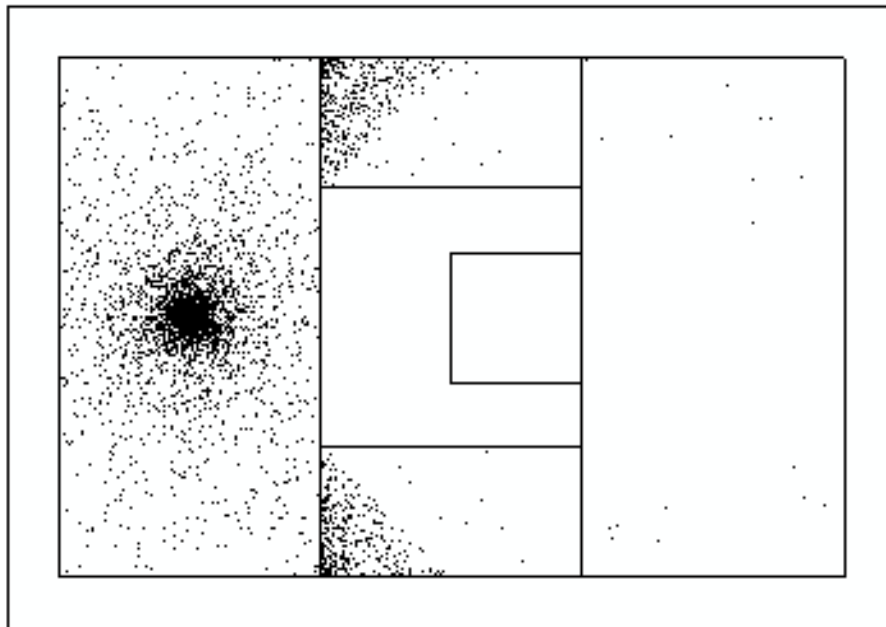
ENERGY, **AGE**, and **WEIGHT** are optional ranges which limit the collision-site plotting to incident particles which have values in the specified range(s).

Example: Specify a neutron ANALYSIS picture to show neutron collisions in a rectangular volume. Restrict plotting to incident neutrons with energies in the range (1.,5.) MeV.

```
ANALYSIS
  NEUTRON      80.  0. 190.$ TOP LEFT CORNER OF PICTURE
  PLANE
                80.  0.  90. $ BOTTOM LEFT CORNER OF PICTURE
  PLANE
                200. 0.  90. $ BOTTOM RIGHT CORNER OF PICTURE
  PLANE
                1.          $ THICKNESS = 1.
  TITLE "NEUTRON SCATTERING"
  ENERGY 1. 5.
```

All energies are in MeV, and distances in cm unless specified otherwise in the BASIC Data Block.

ANALYSIS pictures are written into the COG graphics file, ...ps. Plots are limited to 5000 collisions in the specified volume. It will take some experience to gain a feeling for the information contained in the ANALYSIS pictures. Note that collisions within the volume are projected onto the central plotting plane. An example of an ANALYSIS plot is given below. Several more ANALYSIS pictures are shown in the section **The Monte Carlo Random Walk**.



10.11 Calculated Detector IMPORTANCE

In the description of the SOURCE Data Block, we saw that the source parameters (INCREMENT, ENERGY, TIME, ANGLE, PATH) can be biased by an optional user-specified set of IMPORTANCE values. IMPORTANCE values can also be specified to modify the Monte Carlo random walk process, as described in the section **The Monte Carlo Random Walk**. In every case, the purpose of specifying IMPORTANCES is to make the problem run more efficiently by sampling more frequently those volumes of phase-space which contribute most heavily to the detector result. The problem for the user, of course, lies in knowing what numerical values of importance to specify. Fortunately, COG will optionally list, for each detector in a problem, the relative IMPORTANCES of SOURCES and REGIONs to the detector result. This section describes what these calculated importance values are, how to get them, and how to apply them.

One way to view IMPORTANCE is that it is the ratio of the *result* obtained by a detector specified at $\mathbf{P}'(\mathbf{e}', \vec{\Omega}, t)$, to a *unit particle source* at some point $\mathbf{P}(\vec{r}, \mathbf{e}, \vec{\Omega}, t)$ in the problem's phase space—that is,

$$\text{Importance}(\vec{r}, \mathbf{e}, \vec{\Omega}, t) = \text{Result}(\mathbf{e}', \vec{\Omega}, t) / \text{Source}(\vec{r}, \mathbf{e}, \vec{\Omega}, t) .$$

Volumes in phase space with high values of importance contribute heavily to the result. It is desirable to concentrate the calculational effort on volumes that contribute most of the result. Locations of low importance values contribute only a little to the answer and may be de-emphasized. Once you have determined the relative IMPORTANCES of the SOURCES, REGIONs, and/or PATHs in your problem, you can bias the calculation to run more efficiently.

The COG output file out lists some **default** IMPORTANCE calculations which are performed on every run. For each detector, COG lists the relative IMPORTANCE of the various SOURCE INCREMENTs in the problem for scoring in that detector. For each POINT detector, COG lists the relative IMPORTANCE of the various REGIONs in your problem for scoring in that detector. If you use the SOURCE-PATH option, COG will list for each detector the relative IMPORTANCE of each specified PATH for scoring in that detector.

In addition to these default IMPORTANCE calculations, you can specify **optional** IMPORTANCE calculations for each detector which will list the relative IMPORTANCE of particle SOURCES or REGIONs. You do this by specifying a set of IMPORTANCE bins in particle ENERGY, ANGLE, and/or TIME.

With the **default** IMPORTANCE information, you can make your problem run more efficiently:

- by specifying more particles in the most important SOURCE INCREMENTs;

- by running more particles in the most important PATHs;
- by using WALK-XX biasing to get more particles to scatter in the most important regions.

With the **optional** IMPORTANCE bin values calculated for each detector, you get IMPORTANCE listed by ENERGY, ANGLE, and TIME. You can use this to make the problem run more efficiently:

- by biasing the ENERGY, ANGLE, or TIME-dependences of the particle sources to more heavily sample the most important parts of each parameter range;
- by using the WALK-XX options to more heavily sample those parts of parameter-space (particle ENERGY, ANGLE, TIME, AND REGION) where scattering contributes most to the detector scores.

The main restriction on calculated importances is that particles must get into all parts of the problem before a "correct" set of importances can be calculated. This may force the user to make an educated guess at some form of biasing to get some particles in all regions of the problem. The calculated importances, particularly if split into energy, time and region components, will have poor statistics and often require smoothing to observe the trends in the data.

When you specify an optional IMPORTANCE bin structure to obtain importances by ENERGY, ANGLE, and/or TIME, you must choose whether the bins are to apply to SOURCE or REGIONal importances.

If SOURCE is specified, then the importances are calculated for the particle sources. This means, for example, that if you specify an IMPORTANCE ENERGY bin structure, the energy which will be sorted into the bins is the **initial** energy of the scoring particles, when they were emitted from the SOURCE. Likewise if you ask for TIME or ANGLE bins, the quantity sorted is the **initial** value upon particle emission, for all scoring particles.

If REGION is specified instead, then the energy which will be sorted into ENERGY bins is the **scattered** energy of the scoring particles, as they emerged from collisions in the specified REGIONs. Whenever a particle scores, COG looks back through the Event History record for the particle, and sorts into the ENERGY, AGE, and/or ANGLE bins the values the particle had when it emerged from collisions in the REGIONs.

To get the optional IMPORTANCE calculation performed for a detector, specify:

$$\begin{bmatrix} \text{IMPORTANCE} \\ \text{IMP} \end{bmatrix} \left\{ \begin{bmatrix} \text{REGION} \\ \text{REG} \\ \text{R} \end{bmatrix} \begin{bmatrix} \text{ALL or A} \\ \text{reg-ID\#} \\ \text{reg-ID\#}_1 \text{ TO reg-ID\#}_2 \end{bmatrix} \right\} \\ = \begin{bmatrix} \text{NEUTRON} \\ \text{PHOTON} \end{bmatrix} e_1 e_2 e_3 \dots$$

and/or

$$\begin{bmatrix} \text{ANGLE} \\ \text{A} \end{bmatrix} = \mu_1 \mu_2 \mu_3 \dots$$

and/or

$$\begin{bmatrix} \text{TIME} \\ \text{T} \end{bmatrix} t_1 t_2 t_3 \dots$$

where:

The optional **REGION** specification causes COG to tally importance information for the REGIONs in the problem.

*If the **REGION** specification is omitted, then COG will tally importance information for the **SOURCE**;*

ALL or **A** indicates **all** REGIONs will be tallied;

reg-ID# is the number of the **one** REGION to be tallied;

reg-ID#₁ TO reg-ID#₂ indicates REGIONs numbered **reg-ID#₁** up to and including **reg-ID#₂** will be tallied.

The remaining fields specify the usual COG BIN structure(s) for the particle parameters you are interested in. The optional fields are:

ENERGY or **E** is the keyword denoting the following bins are in ENERGY;

NEUTRON or PHOTON denotes the particle type;

e_i are the BIN edges, in MeV (or in the energy units specified in the BASIC Data Block);

ANGLE or A is the keyword denoting the following bins are in cosine of the polar angle;

u_{ref} v_{ref} w_{ref} are the direction cosines of the polar reference direction;

NORMAL indicates the reference direction is normal to the surface. This can be used only for the special case of a BOUNDARY-CROSSING detector;

m_i are the BIN edges, given in cosine of the angle between the polar reference direction and the particle direction;

TIME or T is the keyword denoting the following bins are in TIME;

t_i are the BIN edges, in seconds (or in the units specified for time in the BASIC Data Block).

COG will list the calculated relative IMPORTANCES of the BINned parameters for scoring in the detector. These values may be used as inputs to the next COG run of this problem by:

- inserting these calculated IMPORTANCES into the optional IMPORTANCE fields of the SOURCE Data Block;
- biasing the random walk with the WALK-XX commands to create more scattering contributions in regions of high importance.

The section *Random Walk Biasing Examples* includes several examples of how detector calculated IMPORTANCES are used and should be referred to for more information.

Note: the calculated DETECTOR IMPORTANCE values can seldom be taken verbatim from the output file and inserted directly into your input file; rather they should be used to guide your choice of importance values for SOURCE or WALK-XX biasing.

Example: In this example we ask COG to calculate IMPORTANCES by REGION and ENERGY, for scoring into a BOUNDARY-CROSSING photon detector.

```
DETECTOR
NUMBER 10
BOUNDARY 10 11 9.534E-4
DRF-E PHOTON ENERGY-FLUX
BIN E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 1 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 2 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
```



```

IMP REGION 3 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 4 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
. . .
IMP REGION 9 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 10 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 11 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.

```

The output for this case shows the calculated IMPORTANCES by REGION and ENERGY BIN. An average IMPORTANCE is also computed for each REGION by summing over all the ENERGY BIN values, i.e. it is the IMPORTANCE one would calculate for a single ENERGY BIN covering the specified energy range.

IMPORTANCE RESULTS BASED ON COLLISIONS IN REGION 1 FOR DETECTOR 10

bin specifications		importance		
photon energy - mev		value	std.dev.	fsd
.000E+00 TO	1.000E-01	.000E+00	.00E+00	.000
1.000E-01 TO	3.000E-01	.000E+00	.00E+00	.000
3.000E-01 TO	5.000E-01	3.389E+02	3.20E+01	.094
5.000E-01 TO	7.000E-01	1.634E+03	2.85E+02	.174
7.000E-01 TO	9.000E-01	2.551E+03	2.89E+02	.113
9.000E-01 TO	1.000E+00	4.081E+03	3.76E+02	.092
average		1.620E+02	1.12E+01	.069

IMPORTANCE RESULTS BASED ON COLLISIONS IN REGION 2 FOR DETECTOR 10

bin specifications		importance		
photon energy - mev		value	std.dev.	fsd
.000E+00 TO	1.000E-01	.000E+00	.00E+00	.000
1.000E-01 TO	3.000E-01	.000E+00	.00E+00	.000
3.000E-01 TO	5.000E-01	1.235E+03	1.21E+02	.098
5.000E-01 TO	7.000E-01	3.171E+03	6.98E+02	.220
7.000E-01 TO	9.000E-01	6.882E+03	9.60E+02	.139
9.000E-01 TO	1.000E+00	1.209E+04	2.24E+03	.185
average		1.548E+02	1.45E+01	.094

IMPORTANCE RESULTS BASED ON COLLISIONS IN REGION 9 FOR DETECTOR 10

bin specifications		importance		
photon energy - mev		value	std.dev.	fsd
.000E+00 TO	1.000E-01	9.586E+03	1.34E+03	.140
1.000E-01 TO	3.000E-01	1.519E+04	1.07E+03	.070
3.000E-01 TO	5.000E-01	1.732E+05	1.57E+04	.090
5.000E-01 TO	7.000E-01	7.611E+05	7.69E+04	.101

7.000E-01 TO	9.000E-01	2.885E+06	5.39E+05	.187
9.000E-01 TO	1.000E+00	1.396E+07	8.76E+06	.627
	average	9.544E+03	4.72E+02	.049

IMPORTANCE RESULTS BASED ON COLLISIONS IN REGION 10 FOR
DETECTOR 10

bin specifications		importance		
photon energy - mev		value	std.dev.	fsd
.000E+00 TO	1.000E-01	3.067E+04	1.95E+03	.064
1.000E-01 TO	3.000E-01	4.717E+04	2.28E+03	.048
3.000E-01 TO	5.000E-01	6.668E+05	6.09E+04	.091
5.000E-01 TO	7.000E-01	2.799E+06	4.24E+05	.151
7.000E-01 TO	9.000E-01	9.447E+06	1.03E+06	.109
9.000E-01 TO	1.000E+00	5.942E+07	2.33E+07	.392
	average	2.660E+04	1.15E+03	.043

IMPORTANCE RESULTS BASED ON COLLISIONS IN REGION 11 FOR
DETECTOR 10

bin specifications		importance		
photon energy - mev		value	std.dev.	fsd
.000E+00 TO	1.000E-01	5.905E+04	3.74E+03	.063
1.000E-01 TO	3.000E-01	1.275E+05	1.48E+04	.116
3.000E-01 TO	5.000E-01	1.620E+06	7.22E+05	.446
5.000E-01 TO	7.000E-01	3.000E+06	4.29E+05	.143
7.000E-01 TO	9.000E-01	.000E+00	.00E+00	.000
9.000E-01 TO	1.000E+00	.000E+00	.00E+00	.000
	average	5.141E+04	5.03E+03	.098

10.12 References

- 1.0 D. K. Trubey and M. B. Emmett, A Comparison of First- and Last-Flight Expectation Values Used in an O5R Monte Carlo Calculation of Neutron Distributions in Water, Oak Ridge National Laboratory, Oak Ridge, TN, ORNL-RSIC-3, (1965).
- 2.0 T. P. Wilcox, "A Method to Calculate Induced Radioactivity Using Existing Transport Codes," in Transactions of the American Nuclear Society, 22, 816, (1975).

11 The Monte Carlo Random Walk

11.1 Analog vs. Biased Monte Carlo

We begin by introducing several concepts; *analog* random walk, particle *weight*, *biasing*, and *figure-of-merit*.

An *analog* random walk is characterized by the absence of random walk modification techniques, i.e. the history of the random walk is determined solely by the probabilities governing the particle's elementary processes. This is the default COG random walk process. You get this unless you specify a WALK-XX modification (see the section **Random Walk Modification Techniques**).

The particle *weight* is a particle property, one of several—particle type, position, direction, age, weight—which describe the state of a particle. Weight is the contribution a particle makes when it is scored by a particle detector. For example, two particles each of weight 1/2 are equivalent in scoring to one particle of weight 1. The weight can be thought of as the number of physical particles associated with each COG particle. Particle weights are affected by biasing or by reaction multiplicities (e.g. the two neutrons exiting the n,2n reaction may be followed as a single neutron having twice the weight of the incoming neutron).

Biasing (or random walk modification) techniques alter the probabilities governing a particle's elementary processes to achieve some desired effect, e.g. to increase the sampling from important regions of phase space. If a problem is biased, each particle's weight must be automatically and appropriately adjusted so as to obtain unbiased results from the now biased problem. Within the same problem, different detectors are generally responsive to different regions of phase space—either the detectors are in different physical locations, are sensitive to different particle types or different energy ranges, etc. Biasing tends to force particles toward one selected point in phase space; therefore a given set of random walk modifications will generally improve results at only one detector while producing poorer results at all others.

The figure-of-merit of a detector result (which has meaning only within the context of a specific problem) is simply the inverse of the product of the detector result variance and the problem running time. Since the variance, or square of the standard deviation, is inversely proportional to the number of particles run, and the running time is directly proportional to the number of particles run, the product should be a constant. The figure-of-merit is intended to monitor the efficiency of the code as various biasing techniques are tried and retried. When the figure-of-merit increases, it means the current biasing has improved the result (i.e., a better variance in the same running time) and more biasing along the same line may be warranted (see the following section **Random Walk—Figure-of-Merit (FOM)**).

To see the need for biasing consider the following hypothetical problem. Particles leave the source and travel a torturous, many-mean-free-path trajectory to reach the detector. For example, assume only one source particle in 10^6 reaches the detector. (Deep penetration problems are characterized by scoring probabilities this small or smaller.) Further assume COG processes an average of 1000 particles in one second. Therefore one can expect an average of one detector score every 1000 seconds. To first order, the uncertainty in an answer goes inversely as the square root of the number of scoring events. Hence to get an answer with 10% uncertainty requires 100 scores or about 28 hours of computing time; a 3% uncertain answer would require almost two weeks! Appropriate biasing techniques may allow this problem to be calculated to 3% uncertainty in a few hours or less. Indeed there are significant numbers of problems that can *only* be calculated in any reasonable amount of time by using biasing techniques.

If they are so powerful, why aren't biasing techniques used all the time? Random walk modification techniques are usually termed "variance reduction techniques". Unfortunately, the very act of biasing introduces a spread in the particle weights and, therefore, may significantly *increase* the variance of the problem answer, for a fixed run time. Biasing is something of a two edged sword; it has the potential for significant computer time savings, but if used incorrectly biasing will increase the running time required to compute an accurate solution. The effect of biasing a problem is to increase the sampling of certain parts of the problem phase space at the expense of other parts. If the increased sampling occurs in regions which contribute most significantly to the problem solution, then the biasing will succeed and running times will decrease. If the converse occurs, then problem solution times will increase over the non-biased case. However, even in the case where bias is erroneously applied, the solution will still be valid if the problem is run long enough. (But note that a valid solution may not be obtained if the user has instructed COG to **never** sample some regions of his problem.) The remainder of this section along with the following examples will provide some general guidelines.

To achieve a fast and accurate problem solution, ideally one might like the transport code to implement one or more random walk biasing techniques automatically. There have been attempts to do this, and some codes have been written with various automatic biasing methods built into them. Most of the automatic techniques were found to work well only with a fairly fixed type of problem—they made some other types of problems run much worse. COG does not include any automatic technique; however, it does produce output that enables the user to look at a short problem run, determine what is happening, and then bias the problem so that it will run more efficiently.

A user should generally make use of only one or two of these random walk modification techniques in the same problem. We allow a great range of choice, to

accommodate different types of problems. While each of the options has occasions where it should or shouldn't be used, the choice is often a matter of personal taste. Thus, splitting and path stretching accomplish the same thing and are mathematically equivalent. In following sections we discuss each of the modification techniques in some detail.

11.1.1 Random Walk IMPORTANCE

In the description of the SOURCE Data Block, we saw that the SOURCE parameters (INCREMENT, ENERGY, TIME, ANGLE, PATH) can be biased by an optional user-specified set of IMPORTANCE values. IMPORTANCE values can also be specified to modify the Monte Carlo random walk process. In every case, the purpose of specifying IMPORTANCES is to make the problem run more efficiently by sampling more frequently those volumes of phase-space which contribute most heavily to the detector result.

One way to view IMPORTANCE is that it is the ratio of the *result* obtained at a specified detector to a *unit particle source* at some point

$(\vec{r}, \mathbf{e}, \vec{\Omega}, t)$ in the problem's phase space—that is,

$$\text{Importance}(\vec{r}, \mathbf{e}, \vec{\Omega}, t) = \text{Result}(\mathbf{e}', \vec{\Omega}', t) / \text{Source}(\vec{r}, \mathbf{e}, \vec{\Omega}, t)$$

Places in phase space with high values of importance contribute heavily to the result. It is desirable to concentrate the calculational effort on areas that contribute most to the result. Places with low importance values contribute only a little to the answer and may be de-emphasized. Once you have determined the relative IMPORTANCES of the SOURCES, REGIONs, and/or PATHs in your problem, you can bias the calculation to run more efficiently.

The problem for the user, of course, lies in knowing what numerical values of IMPORTANCE to specify. Fortunately, COG will optionally calculate for each DETECTOR in a problem, the relative IMPORTANCES of SOURCES and REGIONs to the DETECTOR result. The details are given in the section **Calculated Detector IMPORTANCE**. The examples at the end of this chapter also show some examples of using IMPORTANCES. COG automatically compensates the DETECTOR results for the IMPORTANCE values used, and in a statistical sense the use of IMPORTANCES leads to an unbiased result.

11.1.2 Random Walk Figure-of-Merit (fom)

The figure-of-merit, fom, for a calculation is defined as the inverse of the product of the square of the fsd and the problem running time, i.e.

$$\text{fom} = 1. / (\text{fsd}^2 \times (\text{problem-running-time}))$$

where the fsd is the fractional standard deviation, i.e. the standard deviation divided by the result. As you try different schemes for modifying the random walk, you will want to know how successful you have been *for a given problem*. The standard measure of this is the figure-of-merit. *The more efficient the calculation the higher the fom.*

The figure-of-merit is intended to monitor the efficiency of the code on a given problem as various biasing techniques are tried. The figure-of-merit, strictly speaking, applies only to a given problem on a given machine and should not be used to compare calculations of different problems. When the figure-of-merit increases, it means the current biasing technique has improved the calculation and more biasing along similar lines may be warranted. To first order the fsd is inversely proportional to the square root of n, i.e., $\text{fsd} \sim 1/\sqrt{n}$, where n is the number of particles processed, while the problem-running-time is directly proportional to n. Hence, the figure-of-merit should be approximately independent of n for a well-behaved problem. If a biasing technique is applied and it results in a more efficient calculation, then the fsd will be smaller for a given n, and the fom will be larger.

But note that a small fsd in itself does not necessarily mean that COG has converged to the right answer. In the section **Understanding Your COG Run**, we show how under sampling may yield wrong results that have small fsd's. The user must carefully review COG output files and pictures to be sure that all parts of the problem important to the result have been adequately sampled.

11.2 Random Walk Modification Techniques: The WALK Data Block

SUMMARY OF COG RANDOM WALK MODIFICATIONS

<i>Biasing Technique</i>	<i>Description</i>	<i>Walk-values</i>
WALK-XX	General description of Region and Energy specifications used by all WALK blocks	
WALK-AGE	Termination based on age	t_{co}
WALK-BC	Splitting/Russian roulette at a boundary crossing	I_{val}
WALK-COLLISION	Splitting/Russian roulette at a collision site	β
WALK-ENERGY	Terminate low-energy particles	Ecut
WALK-FC	Forced collisions	none
WALK-FPP	Fast photon physics	level
WALK-ISOTOPE	Force reactions in a specified isotope	ZA
WALK-PRODUCTION	Secondary production control	none
WALK-PS	Path stretching (exponential transform)	q refdir
WALK-REACTION	Force a specified reaction	reactno
WALK-SDB	Scattered direction bias	b refdir
WALK-SURVIVAL	Survival of particle undergoing a collision	none
WALK-WT	Splitting/Russian roulette using statistical weights	minwt maxwt
SOURCE-PATH	Defines a set of particle paths, extending from the SOURCE through selected scattering REGIONS. (see the section SOURCE-PATH Option).	
SOURCE	Source biasing (see the section Source IMPORTANCES).	

<u>Biasing Technique</u>	<u>Walk-values</u>	<u>Definition</u>
AGE	t_{co}	time cutoff,
BC	I_{val}	importance ratio
COLLISION	β	splitting ratio
ENERGY	Ecut	low-energy cutoff
FPP	level	approximation level, $0 < \text{level} < 3$
ISOTOPE	ZA	ZA-ID of isotope (e.g., 13027 for Al)
PS	q	stretching parameter, $-1 < q < 1$
REACTION	refdir	reference direction specification
SDB	reactno	reaction # (e.g., 12, for (n,2n))
	b	strength parameter, $b > 0$
	refdir	reference direction specification
WT	minwt	minimum weight
	maxwt	maximum weight

11.2.1 WALK-XX

How to Specify WALK Parameters

Throughout the following sections, we use WALK-XX as a notational convention to refer to any of the random walk modification blocks, which are all named beginning with "WALK-". All WALK-XX blocks require you to define the volumes of phase-space to which they apply, in terms of particle TYPE, REGION, and ENERGY.

For each particle type, defining the phase-space means specifying a two-dimensional grid that lists REGION number against particle ENERGY.

Each WALK-XX Data Block has this form:

WALK-XX

$$\begin{array}{l}
 \textit{particle-type} \left[\begin{array}{c} \text{REGION} \\ \text{REG} \\ \text{R} \end{array} \right] \left[\begin{array}{c} \text{ALL or A} \\ \textit{rid\#} \\ \textit{rid\#1 TO rid\#2} \end{array} \right] \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] \quad \textit{e1 TO e2} \quad \textit{walk-value1} \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] \quad \textit{e3 TO e4} \quad \textit{walk-value2} \\
 \\
 \dots \\
 \left[\begin{array}{c} \text{REGION} \\ \text{REG} \\ \text{R} \end{array} \right] \left[\begin{array}{c} \text{ALL or A} \\ \textit{rid\#} \\ \textit{rid\#1 TO rid\#2} \end{array} \right] . \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] \quad \textit{e1 TO e2} \quad \textit{walk-value3} \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] \quad \textit{e3 TO e4} \quad \textit{walk-value4} \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] \quad \textit{e5 TO e6} \quad \textit{walk-value5} \\
 \\
 \dots
 \end{array}$$

$$\begin{array}{c}
 \text{particle-type} \left[\begin{array}{c} \text{REGION} \\ \text{REG} \\ \text{R} \end{array} \right] \left[\begin{array}{c} \text{ALL or A} \\ \text{rid\#} \\ \text{rid\#1 TO rid\#2} \end{array} \right] \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] e1 \text{ TO } e2 \text{ walk-value1} \\
 \\
 \left[\begin{array}{c} \text{ENERGY} \\ \text{ENG} \\ \text{E} \end{array} \right] e3 \text{ TO } e4 \text{ walk-value2} \\
 \\
 \dots
 \end{array}$$

Where:

particle-type is one of: NEUTRON, PHOTON (or aliases);

ALL or A denotes all REGIONS;

rid# is the number of the **single** REGION specified;

rid#1 TO rid#2 specifies REGIONS numbered *rid#1* up to and including *rid#2*;

e1 TO e2 ... denote ENERGY ranges. Energies are in MeV unless specified otherwise in the BASIC Data Block;

walk-value1 ... are appropriate random-walk-modification-parameters for this particular WALK-XX, to be applied to this specified range of REGION and ENERGY (i.e. the specified phase-space cell).

For example, for WALK-AGE the *walk-value* is t_{co} , the AGE cutoff value. For WALK-PS the *walk-values* are **q** (the path-stretching parameter) and **refdir** (a reference direction specification).

Note the order in which WALK-XX parameters are specified: The **particle-type** is first, followed by one **REGION** specification, followed by a set of **ENERGY** cells for that **REGION**. Each energy-cell specification is followed by the random-walk-modification-parameter value, to be assigned to this phase-space cell. Another **REGION** is then specified, followed by its set of energy-cell specifications and random-walk-modification-parameter values. When all **REGIONS** have been specified for the first particle-type, the next particle-type and its **REGION/ENERGY**

grid can be specified. The WALK-XX modification applies only to the specified REGIONs and ENERGY cells.

Example: *A general WALK-XX specification.*

```
WALK-XX
NEUTRON      REGION 1 TO 7      ENERGY 0. TO 1.      walk-value
                                     ENERGY 1. TO 5.      walk-value
                                     ENERGY 5. TO 20.     walk-value
              REGION 8 TO 10    ENERGY 0. TO 1.      walk-value
                                     ENERGY 1. TO 10.     walk-value
PHOTON       REGION 1 TO 3      ENERGY 0. TO 20.     walk-value
              REGION 4          ENERGY 0. TO 20.     walk-value
              REGION 5 TO 10    ENERGY 0. TO 1.      walk-value
                                     ENERGY 1. TO 20.     walk-value
```

Each *particle-type* may occur only once in a specific WALK-XX Data Block. The last specification overrides any previous specification; i.e., if for NEUTRON, the entry REGION 8 TO 10 had been REGION 7 TO 10, then REGION 7 would be defined twice and would be specified by the latter energy ranges and walk-values rather than the former.

11.2.2 WALK-AGE

Particle Termination Based On Age

WALK-AGE is a random-walk modification that allows terminating a particle based on its AGE. The *walk-value* is the cut-off time t_{co} for particle AGE. (Time units are seconds unless specified otherwise in the BASIC Data Block.) A particle's AGE is equal to its time of birth (at the source or at a collision point) plus the travel time to its current position.

WALK-AGE terminates a random walk when the particle has reached the specified AGE cut-off either when crossing a boundary or upon exiting a collision. All particles in the specified volume of phase space whose AGE, *after* a collision, exceeds t_{co} , are "counted" and "killed", i.e. the current weight and energy are appropriately tallied and the particle is discarded. Normally, AGE termination is used when you know that particles of AGE $> t_{co}$ cannot contribute to the events of interest in your detector. t_{co} must be large enough for the particle to transit from the source region to the detector region. Beware of killing neutrons too soon, because slow neutrons can produce fast photons, making some transit times a composite of neutron and gamma flight times.

Because AGE checks are done *after* each boundary crossing or collision, detectors may count particles that are too old or secondaries may be produced by primaries that are too old. A detector time mask could be used to remove this uncertainty.

AGE termination is very straightforward and normally produces no unusual effects, other than those listed above. A late time particle can not produce any effect at an earlier time regardless of what it does. An incorrect t_{co} larger than the desired or proper value just makes the problem run longer. A smaller than proper value results in an underestimate of the true result.

Example: *Neutrons in all REGIONS which exit a collision with energy between 0. and 30. MeV and with AGE greater than 1.E-7 seconds will be terminated.*

WALK-AGE

NEUTRON REGION ALL ENERGY 0. TO 30. 1.E-7

Energies are in MeV and times are in seconds unless otherwise specified in the BASIC Data Block.

11.2.3 WALK-BC

Splitting/Russian Roulette at a Boundary

WALK-BC is a random walk modification which will split (increase) or Russian-roulette (decrease) the number of particles which cross a boundary between problem REGIONS. The *walk-value* is the relative IMPORTANCE value I_{val} of the REGION/ENERGY phase-space cell.

Whenever a particle crosses a boundary between problem REGIONS, COG uses the parameter b , to control the average number of particles exiting from the boundary crossing—where b is defined as the ratio of the IMPORTANCES of the REGIONS, i.e.

$$\beta = I_{val\text{-of-the-REGION-entered}} / I_{val\text{-of-the-REGION-left}} .$$

If $b > 1$, then splitting is applied to the particle—more particles enter the new REGION than leave the old REGION. If $b < 1$, then Russian roulette is applied to the particle—fewer particles enter the new REGION than leave the old REGION. The splitting/Russian roulette process is applied here just as it is for the WALK-COLLISION modification (see the section **WALK- COLLISION— Splitting/Russian Roulette at a Collision**).

IMPORTANCES not specified in the input are automatically set to one.

Note: Splitting or Russian roulette applies only to the specified particle-type. Gammas produced by neutrons are not affected (split or killed) by a neutron specification.

For a problem with a set of splitting boundaries between the source and the detector, a general guide for setting splitting parameters is that approximately the same number of random walk particles should cross each boundary. Although the degree of splitting depends in detail upon the energy of the particle and material of the REGIONS, in general particles should be split ~ 2.7 to 1 at boundaries spaced roughly one mean free path apart. Rough estimates of "appropriate" boundary locations usually pay dividends in reduced calculational time for a given statistical accuracy.

The use of boundaries to determine where things are to be modified gives the user considerable control. It also requires more effort defining the problem. COG does not consider the value of particle weights when deciding to split or kill. This eliminates some pitfalls but may not achieve some desired effects.

All the comments, caveats and warnings made about Russian roulette and splitting in the WALK-COLLISION section apply to WALK-BC also.

Example: Neutrons from 0 to 30 MeV leaving REGION 1 and entering REGION 2 are split in 2 (the ratio of IMPORTANCES is 2/1); those moving from REGION 2 to REGION 1 play Russian roulette with a survival probability of 0.5 (the ratio of IMPORTANCES is 1/2).

WALK-BC

NEUTRON	REGION 1	ENERGY 0. TO 30.	1.
	REGION 2	ENERGY 0. TO 30.	2.

The last number on each line is the relative importance of the phase-space region.
Energies are in MeV unless otherwise specified in the BASIC Data Block.

11.2.4 WALK-COLLISION

Splitting/Russian Roulette at a Collision

Splitting and its complementary operation Russian roulette represent the oldest and most successful biasing techniques used to modify the random walk process for deep penetration studies. To illustrate a WALK-COLLISION use, assume a thick slab of material, many mean free paths thick. A source of particles is near one end of the slab and the detector at the other. For materials greater than 10 mean free paths (mfp) thick, the flux exiting the slab will be attenuated on the order of $mfp/2.3$ decades. This implies that for every 10,000 or so particles released from the source, only one would be expected to penetrate to the detector region. We could increase the number reaching the detector if, at every collision site in the slab, we could split each particle into b new particles each of weight w/b , where w is the incoming particle weight. This would improve detector statistics (by increasing the number of scoring particles) while maintaining the correct answer (through our weight adjustment).

WALK-COLLISION is a random walk modification which will split (increase) or Russian-roulette (decrease) the number of particles which emerge from a *collision* site. The *walk-value* is the parameter b , which controls the number of particles which emerge from a collision.

β is defined as the desired number of particles exiting a collision (on average), divided by the number entering the collision, i.e.

$$\beta = \# \text{ particles exiting collision} / \# \text{ particles entering collision} .$$

Particle splitting ($b > 1$) is used to increase particle density in important phase-space volumes. Russian roulette ($b < 1$) is used to reduce the number of unimportant particles. For $b = 0.5$, statistically one half of all exiting primaries will be killed and the remaining half, with their weights doubled, will continue to be followed. For $b = 1.$, a normal random walk will result with no Russian roulette or splitting. For $b = 1.5$, statistically one half of the exiting primaries are unmodified and the remainder are split into two particles, each with half the weight of the original particle. For $b = 2.3$, statistically 3/10 of the exiting primaries are split into three particles, with weight 1/3 of original weight, while the remainder are split into two particles, with weight 1/2 of original weight.

One of the nice features of collision splitting is that the relative distance between splitting events is determined by the cross section properties of the material being traversed. It is not necessary to establish boundaries roughly one mean-free-path apart; the technique itself spaces the splitting and Russian roulette events at collision sites.

Normally, splitting and Russian roulette are treated as separate modifications but within COG we have combined them, so a problem can use only one at a given position in space and energy.

Note: *Splitting or Russian roulette applies only to the specified particle-type. Gammas produced by neutrons are not affected (split or killed) by a neutron specification.*

Example: *Kill off all photons ($\beta = 0.$) which exit collisions in REGION 5 with energies below 0.5 MeV.*

```
WALK-COLLISION
  PHOTON  REGION 5  ENERGY 0. TO 0.5    0.
```

Example: *Russian roulette at a collision, with survival probability of 0.5, neutrons in all REGIONs with energies below 0.1 MeV; split in 2 neutrons exiting a collision in all REGIONs with energies above 5 MeV.*

```
WALK-COLLISION
  NEUTRON  REGION ALL  ENERGY 0. TO 0.1    0.5
                                ENERGY 5. TO 30.    2.
```

Energies are in MeV unless otherwise specified in the BASIC Data Block.

Values of β may take on any positive value and are not limited to those examples given above. Often, β has values greater than 1 at high particle energies and less than 1 at low energies, i.e. split at high energies and play Russian roulette at low energies. Similarly, at regions in phase space where the importance is high, β should be greater than 1, and where the importance is low, β should be less than 1.

A drawback to splitting is that you may over split. One source particle can be divided so often that there are thousands of particles being scored at the detector per source particle. It is easy to forget that the source has not been sampled sufficiently enough to obtain a good result or that adequate initial paths have not been followed. COG indicates this by summing all detector results over a single source particle before statistical parameters are calculated.

Another drawback to splitting is its effect on some neutron problems. It is easy to get a situation where a neutron scatters hundreds of times before it is eventually absorbed or is lost from the system. If even modest splitting were being done at each of these collisions, the population of neutrons could grow without bound and the code would never finish processing the original neutron and all its split progeny.

In COG, splitting and Russian roulette occur independently of the weight of the particle. It is not difficult to devise a problem where particles of very low weight

contribute almost all of the score to a time-dependent detector result, while the higher-weight particles contribute little. If splitting or Russian roulette were made dependent on particle weight, very wrong answers would be obtained in this case. By not considering particle weight, COG does not get into trouble with this form of splitting.

Russian roulette is not a precisely fair game, i.e. a particle's weight is conserved only in a statistical sense. Russian roulette is played by comparing a random number to β . Random numbers are uniformly distributed between zero and one, so a fraction β of all random numbers have values less than β . If the selected random number, r , is less than β then the particle survives and its weight is multiplied by $1/\beta$, while if r is greater than β the particle is killed, i.e. its weight is set to zero and it is no longer followed. Suppose 10 weight 1 particles undergo Russian roulette with a β of 0.7. If exactly 70% survive then the resultant total weight is $7(1/0.7) + 3(0) = 10$, which is just the initial total weight—a fair game. But it could easily happen that less than 7, say 6, survive, then $6(1/0.7) + 4(0) = 8.57 < 10$ —a net loss in weight; or more than 7, say 8, survive, then $8(1/0.7) + 2(0) = 11.43 > 10$ —a net gain in weight. The "lost" particles are locally deposited, producing a positive energy deposition entry under Russian roulette in the Summary Tables. The "gained" particles are created, resulting in a negative energy deposition entry.

Splitting, on the other hand, is always a fair game, even though the splitting factor is only statistically attainable. Splitting is done by comparing a random number, r , to $\beta - [\beta]$, where $[\beta]$ is the integer part of β . If r is less than $\beta - [\beta]$ the particle is split into $1 + [\beta]$ particles, with each weight multiplied by $1/(1 + [\beta])$. If r is greater than $\beta - [\beta]$, the particle is split into $[\beta]$ particles, each with a weight $1/[\beta]$ times the original weight. For example, if $b = 2.73$, then (statistically) 73% of the particles are split into 3 and the remaining 27% are split into 2. Because all the particle weights have been adjusted by the number of particles into which they were split, the total initial weight is conserved and the game is always fair.

11.2.5 WALK-ENERGY

Particle Termination Based On Energy

Transport of low-energy particles is stopped by use of this option. When a particle's kinetic energy drops below a user-specified threshold, tracking of the particle is terminated and the residual energy is dumped at the current location. This is a time-saving feature which can safely be used in regions where it is known that low-energy particles cannot contribute to the desired detector score.

The *walk-value* is the low-energy cutoff value **Ecut**, which applies to particles in a specified region.

Example: Kill off all neutrons in REGIONS 1-4 which have energies below 0.010 MeV, and also all neutrons in REGION 10 which have energies below 0.001 MeV.

WALK-ENERGY

NEUTRON	REGION 1 TO 4	0.010
	REGION 10	0.001

Energies are in MeV unless otherwise specified in the BASIC Data Block.

Note: The usual WALK-Block ENERGY syntax, “ENERGY xxx TO xxx”, is NOT used with the WALK-ENERGY option.

11.2.6 WALK-FC

Forced Collisions

In some types of problems, the particles arriving at a detector come predominately from a particular scattering region of the geometry (a scattering foil, say). If the scattering region is thin, most particles arriving at the scattering region will pass on through it without interaction, and thus few will score in your detector. If a successful problem result depends on having many collisions in this thin region, then you will have to run a very large number of source particles to get a meaningful number of collisions and an acceptable detector result. WALK-FC is a random walk modification which provides a way to solve this type of problem by forcing collisions to occur where desired. No *walk-value* needs to be specified.

When a particle encounters a REGION designated by the user as a WALK-FC (forced collision) region, it will be split into two particles. One particle will go through the region without interaction, while the other will be forced to have a collision. The particle forced to have a collision will not be allowed to have another forced collision during the remainder of its random walk. The other particle may undergo another forced collision if it encounters another WALK-FC region.

Note: One of the peculiarities of the WALK-FC biasing is that it does not apply to the SOURCE region, even if REGION ALL is specified. The particle has to encounter a boundary before it is tested to see if it is in a WALK-FC region. If it is born in a particular region it can transit that region and exit without ever being forced to collide. The effect of having forced collisions in the SOURCE region would be to essentially double the number of particles in the problem. If forced collisions are desired near the SOURCE, the problem should be devised so that another region closely surrounds the SOURCE region (e.g. place a small sphere around a POINT SOURCE).

Caution: Just like the WALK-SURVIVAL technique, WALK-FC can give you many random walk events based upon just a few histories. Be aware of how many events are taking place and whether you have created too many forced collision events. It is also possible to force collisions in a thick region. In this case, the exit particles represent uncollided particles, and the forced collisions will all be toward the entry point of the region. Both of these scenarios can yield incorrect histories. The WALK-FC method was designed for *thin* regions—those less than *about 0.1 mean-free-paths* in thickness. If the region is thicker than this, you should probably not use this method.

Example: Force NEUTRONS of all energies entering REGION 3 to collide.

WALK-FC

NEUTRON REGION 3 ENERGY 0. TO 30.

Energies are in MeV unless otherwise specified in the BASIC Data Block.

11.2.7 WALK-FPP

Fast Photon Physics

This feature speeds up transport of photons by replacing COG's normal physics models with faster approximate-physics versions. Of course, this feature should only be used in problems where these approximations are warranted. The *walk-value* to be specified is the level of approximation **level**, which takes on integer values from 0 to 3. Higher values of **level** correspond to more approximate scattering physics.

level = 0 gives the normal COG full physics; all reactions are fully treated. These reactions include Rayleigh, photoelectric, Compton, pair and triplet production, and (gamma,n) reactions.

level = 1 gives these photon scattering approximations:

Photoelectric reactions are treated as pure absorption events;

(g,n) reactions do not occur.

level = 2 is the same as 1, with this additional approximation:

Rayleigh events do not scatter the photon;

level = 3 is the same as 2, with this additional approximation:

Compton scattering does not use form-factors;

The greatest single time-saving approximation for most jobs arises from treating the photoelectric reaction as a pure absorption, saving the time required to sample the relaxation data for the target atom and transport the secondaries. In many cases the secondary particles may be too low in energy to contribute to scores at a distant detector, and so can be safely ignored.

***Example:** Turn on Fast Photon Physics, using approximation level 2.*

WALK-FPP

PHOTON REGION 3 ENERGY 0. TO 30. 2

Energies are in MeV unless otherwise specified in the BASIC Data Block.

11.2.8 WALK-ISOTOPE

Forced Collision with an Isotope

The WALK-ISOTOPE feature is used to force scattering with a particular isotope of a target material. This is an efficiency measure used to increase the number of particle collisions with an isotope which is scarce but important to the detector result. The *walk-value* is **ZA**, the isotope's ZA-ID number.

Example: Force neutron collisions with N^{15} in the specified region.

```
WALK-ISOTOPE
      NEUTRON  REGION 3  ENERGY 0. TO 30.    7015
```

Energies are in MeV unless otherwise specified in the BASIC Data Block.

At every collision in the designated regions of the specified particle whose energy lies in the specified energy range, two collisions are made to occur. The incident particle is split into two particles, which are given weights in the ratio of $\mu_i/(\mu_t - \mu_i)$, where μ_i is the total cross section of the specified isotope, and μ_t is the total cross section of the material. The particle labelled with weight μ_i is forced to collide with the specified isotope, while the other particle collides with an isotope sampled from the remaining isotopes in the material. This is a fair game, because the weights are adjusted by the relative probabilities of scattering. This method also generates the correct flux field, as would be found from an analog calculation.

Note1: particle-type must be NEUTRON or PHOTON.

Note2: for particle-type NEUTRON, walk-value (ZA) should not represent a fissile isotope ($ZA > 86000$). Forced collisions in a fissile isotope will cause COG's internal EHS (event history bank) to overflow, causing COG to stop with an error message. This may also occur if the remaining isotopes contain a large fissile component.

11.2.9 WALK-PRODUCTION

Secondary Particle Control

The WALK-PRODUCTION is a random-walk modification used to suppress all gamma production from neutron collisions. For WALK-PRODUCTION, *particle-type* must be NEUTRON; no *walk-value* needs to be specified.

This option prevents COG from following secondary particles from REGIONs which the user knows cannot make a significant contribution to the detector results. This saves computational time. No adverse effects can occur unless the suppressed particles really would have made a significant contribution.

Example: Suppress secondary (gamma) production by neutrons in REGIONs 2 and 4 for ENERGIES below 0.5 MeV.

```
WALK-PRODUCTION
```

```
NEUTRON  REGION 2  ENERGY 0. TO 0.5  
          REGION 4  ENERGY 0. TO 0.5
```

Energies are in MeV unless otherwise specified in the BASIC Data Block.

11.2.10 WALK-PS

Path Stretching/Exponential Transform

WALK-PS is a random walk modification which increases the distance a particle travels before colliding. In a deep-penetration problem one often wishes to generate particle collisions in some region distant from the source, where scatterings are thought to contribute heavily to a detector result. In an analog random walk, it may happen that too few source particles make it to this volume to give good detector statistics. WALK-PS allows you to reduce the effective number of mean free paths between the source and the volume of interest, so that many particles can arrive and scatter at this location. The *walk-value* is the path stretching parameter q , and the preferred direction specification **refdir**. These are described below.

The theoretical basis for this method is as follows. Within a volume containing a single material, the probability of a particle traveling a distance x along its line of flight without colliding, and then undergoing a collision in the next incremental distance dx is:

$$p(x)dx = \exp(-\sum_t x) \sum_t dx ,$$

where $\sum_t$ is the macroscopic total cross section of the material and has units of probability-of-a-collision per unit length. Setting r , a random number on the interval $[0,1]$, equal to the cumulative probability distribution $P(x)$ gives:

$$r = P(x) = \int_0^x \exp(-\sum_t s) \sum_t ds = 1 - \exp(-\sum_t x) .$$

Solving for x yields:

$$x = -\ln(1 - r) / \sum_t .$$

Since $1 - r$ is distributed the same as r , we obtain the well-known expression for the distance, x , to the next collision:

$$x = -\ln(r) / \sum_t$$

Choosing r uniformly on the interval $[0,1]$ generates values of x which are exponentially distributed as $p(x)$. This is how COG selects the particle next-free-flight distance. For deep-penetration problems, the next-free-flight distance calculation can be modified to emphasize particles going toward the volume of interest and de-emphasize particles going in the other directions. When path stretching is employed, we substitute for $\sum_t$ in the next-free-flight distance an effective cross section $\sum_t^{\text{eff}}$:

$$\Sigma_t^{\text{eff}} = (1 - q \mu) \Sigma_t .$$

The quantity μ is the cosine of the angle between the particle's direction of travel and the preferred direction, and is calculated by the code for each particle interaction. To determine this angle a preferred direction **refdir** must be specified as part of the *walk-value*.

q is the user-specified path stretching parameter, and **must** fall within the range [-1,1]. For **q** > 0 particles are preferentially moved toward the reference direction, for **q** = 0 the method is not used, and for **q** < 0 particles are preferentially moved away from the reference direction.

The reference direction (**refdir**) can be specified using any one of the following reference direction forms:

POINT *x y z*

the reference direction is from the particle position towards a **POINT** at (*x*, *y*, *z*);

[SPHERE]
SPH
S *r x y z*

the reference direction is from the particle position towards the closest point on a **SPHERE** of radius *r*, with center at (*x*, *y*, *z*);

[CYLINDER]
CYL
C *r x1 y1 z1 x2 y2 z2*

the reference direction is from the particle position towards the closest point on a **CYLINDER** of radius *r* defined by two points on its axis, (*x1*, *y1*, *z1*) and (*x2*, *y2*, *z2*);

[RING]
R *r x1 y1 z1 x2 y2 z2*

the reference direction is from the particle position towards the closest point on a **RING** of radius *r*, with center at (*x1*, *y1*, *z1*) and with a point on its axis at (*x2*, *y2*, *z2*);

[DIRECTION]
DIR
D *u v w*

the reference direction is in the **DIRECTION** defined by direction cosines (*u*, *v*, *w*).

Path stretching is a powerful technique for deep penetration problems. Obviously, one of the problems with this technique is how to determine the proper value of q to use in any specific problem. A single value of q used throughout all of phase space rarely provides a good solution. The value of q should at least depend on significant regional differences. To understand the choice of q one should understand the physics of particle movement through materials. One general rule is that if you wish to go some number of mean-free paths in the reference direction ($\mu = 1$), choose:

$$q \approx 1 - 1/\text{mfp}$$

where mfp is the desired number of mean-free paths—this has the effect of "condensing" mfp mean-free paths to 1 mean-free path. For example, this equation would imply that for a region which is 20 mean-free paths distant from the source, then $q \approx 0.95$. A region only 5 mfps away would use $q \approx 0.8$. A region only one mfp away probably should not use the path stretching technique ($q = 0$).

Too small a q will not achieve the desired particle penetration. Too large a q will move particles through the region with little or no interactions. Both situations may lead to invalid results due to inadequate sampling of phase space.

A note of caution concerning WALK-PS path stretching. In a neutron transport problem, the neutrons that reach the vicinity of the detector typically fall into two broad categories, those with high energy that have scattered only a few times and those with low energy that have scattered many times. This is the old *two group* idea used for hand calculations. Doing path stretching on the high energy group is generally valid since these particles had only scattered a few times to reach the detector, and path stretching will not change that. However the low energy group had scattered many times, and the energy and angular distributions in the detector region are very much dependent on this fact. Path stretching these particles so that they reach the detector in only a few scatters will seriously alter the energy and angular distributions and hence may very well distort the results you are trying to calculate.

The good points of this path stretching technique are that it does not require many extra boundaries, it is employed at each collision site, and it is sensitive to particle direction. The only problem is the proper choice of q .

Example: Neutrons with energies of 0 to 30 MeV in REGION 1 are stretched (the equation: $q = 1 - 1/\text{mfp}$ would imply about 10:1) **toward** the closest point on a sphere of radius 10 centered at (0,0,0), while those in REGION 2 are stretched (20:1) **away** (the minus .95) from the point (0,0,0).

WALK-PS

NEUTRON R 1 E 0. TO 30. 0.9 SPHERE 10. 0. 0. 0.

R 2 E 0. TO 30. -0.95 POINT 0. 0. 0.

Energies are in MeV, distances in cm unless otherwise specified in the BASIC Data Block.

11.2.11 WALK-REACTION—Forced Reaction

WALK-REACTION is used to force a scattering particle to undergo a specified reaction with the target material. This is an efficiency measure used to increase the number of particles which undergo a particular reaction which is rare but important to the detector result. The *walk-value* is **reactno**, the reaction number. See **COG Reaction Numbers**.

Example: Force neutrons which collide in the specified regions to undergo (n,2n) reactions (**reactno** = 12).

```
WALK-REACTION
NEUTRON REGION 3 TO 7 ENERGY 5. TO 30. 12
```

Energies are in MeV unless otherwise specified in the BASIC Data Block.

At every collision in the designated regions of the specified particle whose energy lies in the specified energy range, two collisions are made to occur. The incident particle is split into two particles, which are given weights in the ratio of $\mu_r / (\mu_t - \mu_r)$, where μ_r is the partial cross section of the specified reaction, and μ_t is the total cross section. The particle labelled with weight μ_r is forced undergo the specified reaction, while the other particle undergoes a reaction sampled from the remaining available reactions. This is a fair game, because the weights are adjusted by the relative probabilities of the reactions. This method also generates the correct flux field, as would be found from an analog calculation.

Note1: *particle-type must be NEUTRON or PHOTON. Not all materials may have data for a specified reaction number. In any collision, the specified reaction must be energetically possible or else this technique will not be applied.*

Note2: *this feature is often used in concert with the WALK-ISOTOPE feature. Be aware that reaction biasing is applied to all isotopes, not just the one selected by the WALK-ISOTOPE feature.*

11.2.12 WALK-SDB—Scattered Direction Bias

WALK-SDB is a technique which biases the angular distribution of particles emerging from a collision, as specified by the user. This is often done to increase the number of particles scattered in a preferred direction. This enables one, in some degree, to "channel" particles into more important regions of the geometry, thus obtaining results which may be difficult if not impossible to get in an unbiased problem. The *walk-value* is the biasing strength function b and the preferred direction specification **refdir**. The preferred direction **refdir** is specified as for path stretching (see the section **WALK-PS—Path Stretching/Exponential Transform**). For example, you may specify **refdir** using one of these forms: **POINT**, **SPHERE**, **CYLINDER**, **RING**, or **DIRECTION**.

The relative IMPORTANCE assigned to any direction is determined by the biasing strength function b , where $b \geq 0$. The IMPORTANCE is:

$$\text{IMPORTANCE} = 1 / (1 + b (1 - \mu))$$

where μ is the cosine of the angle between the preferred direction and the exit direction of the particle.

Because of the form of the biasing function and the assumption that the scattering probability distribution has azimuthal symmetry, the biased scattering probability distribution is a function of μ_s , the cosine of the scattering angle, b , the strength parameter, and W , the cosine of the angle between the incoming particle direction and the preferred direction.

To get an idea of the effects of b and W on the biased scattering probability distribution, we evaluate the biased distribution for the case of isotropic scattering, i.e. the unmodified distribution has a value of $1/2$ for all μ_s . Tables 1, 2, and 3 are for the cases $W = 1.$, 0.8 , and $0.$, respectively, and for values of b of $0.$ (no biasing), $10.$, $30.$, $100.$, and 300 . Note that as W approaches $1.$ the effects of b are enhanced, i.e. for a given b the distribution for $W = 1.$ is more sharply peaked about the reference direction than the distribution for $W = 0.$

Table 1

Particle parallel to reference direction, $W = 1$.

Effect of Strength Parameter (b) on Scattering Distribution

Scattering Distribution Value					
μ_s	$b = 0.$	$b=10.$	$b=30.$	$b=100.$	$b=300.$
-1.0	0.500	0.156	0.120	0.094	0.078
-0.9	0.500	0.164	0.126	0.099	0.082
-0.8	0.500	0.173	0.133	0.104	0.087
-0.7	0.500	0.182	0.140	0.110	0.092
-0.6	0.500	0.193	0.149	0.117	0.097
-0.5	0.500	0.205	0.159	0.125	0.104
-0.4	0.500	0.219	0.170	0.134	0.111
-0.3	0.500	0.235	0.182	0.144	0.120
-0.2	0.500	0.253	0.197	0.156	0.130
-0.1	0.500	0.274	0.215	0.170	0.142
0.	0.500	0.299	0.235	0.187	0.156
0.1	0.500	0.328	0.261	0.207	0.173
0.2	0.500	0.365	0.292	0.233	0.195
0.3	0.500	0.411	0.332	0.266	0.222
0.4	0.500	0.469	0.384	0.309	0.259
0.5	0.500	0.547	0.456	0.370	0.310
0.6	0.500	0.657	0.561	0.460	0.387
0.7	0.500	0.821	0.730	0.609	0.515
0.8	0.500	1.095	1.043	0.898	0.769
0.9	0.500	1.642	1.824	1.714	1.512
1.0	0.500	3.285	7.298	18.856	46.885

Table 2

Particle at 36.87° to reference direction, $W = 0.8$

Effect of Strength Parameter (b) on Scattering Distribution

Scattering Distribution Value					
μ_s	$b = 0.$	$b=10.$	$b=30.$	$b=100.$	$b=300.$
-1.0	0.500	0.173	0.133	0.104	0.087
-0.9	0.500	0.182	0.140	0.110	0.092
-0.8	0.500	0.193	0.149	0.117	0.097
-0.7	0.500	0.205	0.159	0.125	0.104
-0.6	0.500	0.218	0.169	0.134	0.111
-0.5	0.500	0.233	0.182	0.144	0.120
-0.4	0.500	0.251	0.197	0.156	0.130
-0.3	0.500	0.271	0.214	0.170	0.142
-0.2	0.500	0.295	0.234	0.186	0.156
-0.1	0.500	0.323	0.259	0.207	0.173
0.	0.500	0.356	0.289	0.232	0.194
0.1	0.500	0.397	0.327	0.264	0.222
0.2	0.500	0.448	0.377	0.307	0.258
0.3	0.500	0.512	0.443	0.366	0.309
0.4	0.500	0.594	0.535	0.452	0.385
0.5	0.500	0.700	0.672	0.590	0.510
0.6	0.500	0.837	0.884	0.839	0.750
0.7	0.500	0.999	1.210	1.372	1.374
0.8	0.500	1.147	1.535	2.207	3.183
0.9	0.500	1.191	1.410	1.505	1.434
1.0	0.500	1.095	1.043	0.898	0.769

Table 3

Particle at 90° to reference direction, $W = 0.0$ Effect of Strength Parameter (b) on Scattering Distribution

Scattering Distribution Value					
μ_s	$b = 0.$	$b=10.$	$b=30.$	$b=100.$	$b=300.$
-1.0	0.500	0.299	0.235	0.187	0.156
-0.9	0.500	0.325	0.260	0.207	0.173
-0.8	0.500	0.356	0.289	0.232	0.194
-0.7	0.500	0.393	0.326	0.264	0.222
-0.6	0.500	0.435	0.372	0.306	0.258
-0.5	0.500	0.484	0.432	0.363	0.308
-0.4	0.500	0.540	0.510	0.444	0.383
-0.3	0.500	0.600	0.612	0.568	0.503
-0.2	0.500	0.657	0.741	0.769	0.723
-0.1	0.500	0.700	0.872	1.087	1.210
0.	0.500	0.717	0.934	1.330	1.912
0.1	0.500	0.700	0.872	1.087	1.210
0.2	0.500	0.657	0.741	0.769	0.723
0.3	0.500	0.600	0.612	0.568	0.503
0.4	0.500	0.540	0.510	0.444	0.383
0.5	0.500	0.484	0.432	0.363	0.308
0.6	0.500	0.435	0.372	0.306	0.258
0.7	0.500	0.393	0.326	0.264	0.222
0.8	0.500	0.356	0.289	0.232	0.194
0.9	0.500	0.325	0.260	0.207	0.173
1.0	0.500	0.299	0.235	0.187	0.156

Table 4 lists, for the case of isotropic scattering and $W = 1.$, $\theta_{1/2}$ and FoB

[Forward/Back] versus $b.$, $\theta_{1/2}$ is the half-angle of the cone which contains 1/2 the scattered particles, and FoB is the ratio of particles scattered into the forward solid angle [$\mu = 0.9$ to $1.$, or 0° to 26°] divided by the number scattered into the backward solid angle [$\mu = -1.$ to $-0.9.$ or 180° to 154°]

Table 4
Calculation of the effect of Strength Parameter b
and the scattering half-angle cone.

b	$\theta_{1/2}$	FoB
0	90.°	1.00
1	75.°	2.81
3	63.°	5.99
10	50.°	14.2
30	39.°	27.5
100	30.°	47.0
300	23.°	67.1
1000	17.°	90.0
3000	13.°	111.
10000	9.6°	135.
30000	7.3°	156.
100000	5.4°	180.

As Table 4 shows, very large values of the strength parameter b are needed to bias the scattering into a reasonably small cone. As with Russian roulette, this biasing technique produces a fair result only on average. With photons, the Compton and Rayleigh distributions are sampled as usual, then angular biasing is done by rejection techniques. This will be very inefficient if you are trying to force photons into directions they are unlikely to go, based on their normal scattering probabilities. We have had good success using this scheme in conjunction with other modification techniques, specifically path stretching.

Note that b is a simple function of only one angle, for it is expected that there will be problems enough in finding the correct value of b with these simplifying assumptions. An importance based on a full three-dimensional angular representation would be too difficult to express with any degree of accuracy. A value of b equal to zero results in equal importance in all directions, while increasingly larger positive values of b results in importance peaking towards the preferred direction. **Negative values of b are not allowed.**

***Example:** The exit direction of scattered neutrons from 0 to 30 MeV in REGION 1 are forced (hard, $b = 10000$) toward the closest point on a RING of radius 0.75 centered at 0,0,1 with axis in the z direction, those in REGION 2 are forced (harder, 100000) in the z DIRECTION and those in REGION 4 are forced (hard, 10000) toward the POINT 0,0,2.*

WALK-SDB

```
NEUTRON  REGION 1  ENERGY 0. TO 30.  10000.
                                RING 0.75 0. 0. 1. 0. 0. 2.
                                REGION 2  ENERGY 0. TO 30.  100000.
                                DIRECTION 0. 0. 1.
                                REGION 4  ENERGY 0. TO 30.  10000.
                                POINT 0. 0. 2.
```

Energies are in MeV, distances in cm, unless otherwise specified in the BASIC Data Block.

11.2.13 WALK-SURVIVAL

Survival of a Particle in a Collision

WALK-SURVIVAL is a random walk modification which forces a particle to survive every collision in the designated phase-space volumes, even though there is a finite probability of particle absorption. No *walk-value* needs to be specified.

When a particle undergoes a collision, there is a probability that it will be absorbed, producing no scattered primary particle. But there are times when much calculational effort has been expended to get the particle to a given location in the problem, and it would be desirable to keep the particle "alive" to get more information from this random walk history. WALK-SURVIVAL does not allow the primary particle to be completely absorbed. Each particle which would normally scatter, scatters and has its weight reduced by a factor equal to the scattering probability. Each particle which would normally be absorbed is split into two particles. One particle is absorbed; its weight is unchanged. The other particle scatters; its weight is reduced by a factor equal to the scattering probability. In this way, the net weight of scattered and absorbed particles is maintained, but every absorption event contributes a scattered particle.

Two problems have been encountered when using the survival method. The first is that you can get significant contributions to your detector results from only a few, or even one, random walk histories. There are many paths in a real problem for moving a particle from point A to a detector at point B. If WALK-SURVIVAL is turned on, it is possible for a particle to move from A to B along one path and scatter many times in the vicinity of B (while avoiding absorption). This single particle can thus contribute heavily to the detector's results. The results may look significant, but you have sampled just a few of the possible paths from A to B, and you may not have obtained a reliable answer. COG points this problem out to you by determining only one score per source particle—the sum of all these contributions. Check the listing to see how many source particles contribute to the total detector result.

The second problem does not cause calculational errors but does create computational difficulties. If you never allow a particle to be absorbed and if it never escapes the geometry, then it *never dies* unless you put in some other method of killing it. Generally you should avoid using WALK-SURVIVAL in large volumes and perhaps mix the SURVIVAL method with one of the other options to kill off undesired particles. Weight windows are excellent for this purpose.

***Example:** Do survival weighting on neutrons in REGIONS 4 through 7 for energies above 1. and less than 30. MeV and for REGION 10 for energies above 5. and below 30 MeV.*

WALK-SURVIVAL

NEUTRON R 4 TO 7 E 1. TO 30.

R 10 E 5. TO 30.

Energies are in MeV unless otherwise specified in the BASIC Data Block.

11.2.14 WALK-WT

Splitting/Russian Roulette Based on Weight

WALK-WT is a random walk modification which will split (increase) or Russian roulette (decrease) the number of particles which emerge from a collision site, based on particle weight. The *walk-value* is the pair of quantities **minwt** and **maxwt**, which define the lower and upper limits of a weight-window. COG examines the emerging particle-weight, and takes an action based on weight-window comparisons.

If **minwt** $\leq$ particle-weight $\leq$ **maxwt**, the particle is within the weight-window and no modification is made. If particle-weight $<$ **minwt**, Russian roulette is played with β equal to particle-weight/**minwt** and the survivors are assigned a weight equal to **minwt**. If particle-weight $>$ **maxwt**, then splitting is done with β equal to particle-weight/**maxwt** and the resulting particles are assigned a particle-weight equal to **maxwt**.

This process insures that all particles in the specified volume of phase space have weights within the defined weight-window. This is a technique used in the ORNL code O5R¹ and in the LANL code MCNP,² where it is called by the name weight window. As long as you know that high-weight particles are high-importance particles and that low-weight particles are low-importance particles, this method offers some nice features; but it is difficult to use effectively.

It is difficult for the user to determine the proper values for **minwt** and **maxwt**. It is of little value to say that they are related to the values of particle flux, because if you knew the flux, you would know the answer and wouldn't run the problem.

Note: Splitting or Russian roulette applies only to the specified particle-type. Gammas produced by neutrons are not affected (split or killed) by a neutron specification.

Example: Photons in all REGIONs and with energies between 0 and 30 MeV will have their post-collision weights adjusted by splitting or Russian roulette to be between 0.0001 and 2.

WALK-WT

PHOTON REGION ALL E 0. TO 30. 0.0001 2.

Energies are in MeV unless otherwise specified in the BASIC Data Block

11.3 Random Walk Biasing Examples

In this section we present several examples to guide the user in employing the various biasing techniques. In each example we will proceed in an orderly fashion, always building toward an "optimal" solution. We realize that in the real world what is needed is a satisfactory—not necessarily an optimal—solution; but we hope that this step-by-step procedure will be informative. These problems run quite quickly—in 1.5 to 5 minutes. They were chosen for instructional purposes, not necessarily as examples of problems which require biasing. It may prove helpful for users to run these problems and study the complete COG output. (Keep in mind, because of different random number sequences, you will not get exactly the same results as presented here).

11.3.1 Example 1— Splitting at a Boundary

This problem consists of a sphere of Al with a point source of 1 MeV gammas at the center, emitting isotropically. The radius of the sphere is more than 10 mean free paths (mfp) for the 1 MeV gammas. The energy build-up factors—the ratio of the total gamma energy to the uncollided energy—have been calculated using another (non-Monte-Carlo) method⁽¹⁾ for radii corresponding to 1, 2, 4, 7, 10 mfp. We start with a rather simple goal—to obtain a good result at 10 mfp. We have placed a BOUNDARY-CROSSING detector at each of these radii. Each of the COG detectors is normalized by dividing the computed detector score by the "correct" result at each location (obtained from the reference). Thus if COG calculates the "correct" value of flux at a detector position, the detector response will equal unity.

The initial input to COG is as follows:

```
ENERGY BUILD UP IN AL @ 1 MEV (NYO-3075)
$  V31.1A
BASIC
    PHOTON
GEOMETRY
    SECTOR   1  Z01      -1
    SECTOR   2  Z02      1 -2
    SECTOR   3  Z03      2 -3
    SECTOR   4  Z04      3 -4
    SECTOR   5  Z05      4 -5
    SECTOR   6  Z06      5 -6
    SECTOR   7  Z07      6 -7
    SECTOR   8  Z08      7 -8
    SECTOR   9  Z09      8 -9
```



```

        SECTOR 10 Z10      9 -10
        SECTOR 11 Z11      10 -11

        BOUNDARY VACUUM    11
SURFACES
    1  S 6.033
    2  S 12.066
    3  S 18.1
    4  S 24.133
    5  S 30.166
    6  S 36.199
    7  S 42.232
    8  S 48.266
    9  S 54.299
   10 S 60.332
   11 S 66.365
MIX
    MAT 1  AL 2.7
ASSIGN-ML
    1  2 3 4 5 6 7 8 9 10 11
DETECTOR
$  ALL DETECTORS ARE NORMALIZED TO 1.
    NUMBER 1
        BOUNDARY  1 2 7.3944E-1
        DRF-E  PHOTON  ENERGY-FLUX

    NUMBER 2
        BOUNDARY  2 3 4.4525E-1
        DRF-E  PHOTON  ENERGY-FLUX

    NUMBER 4
        BOUNDARY  4 5 1.1942E-1
        DRF-E  PHOTON  ENERGY-FLUX

    NUMBER 7
        BOUNDARY  7 8 1.1809E-2
        DRF-E  PHOTON  ENERGY-FLUX

```

```

NUMBER 10
BOUNDARY 10 11 9.534E-4
DRF-E PHOTON ENERGY-FLUX

$EXAMPLE OF AN DETECTOR IMPORTANCE CALCULATION
BIN E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 1 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 2 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 3 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 4 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 5 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 6 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 7 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 8 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 9 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 10 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 11 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
ANALYSIS
PHOTON 0. 65. 0. 0. 0. 0. 65. 0. 0. 6.
SOURCE
NPART 20000
DEFINE P 1
POINT 0. 0. 0.
DEFINE E 1
PHOTON LINE 1. 1.
INC 1. P 1 E 1
END

```

SUMMARY OF PROBLEM INPUT FILE

First line: Problem title and comments.

BASIC: Photon only problem.

GEOMETRY: 11 regions bounded by 11 spherical surfaces. The vacuum boundary tells COG to not follow particles which cross surface 11.

SURFACES: 11 concentric spherical surfaces at 1 mfp intervals.

MIX: Region 1 contains material 1, Al at a density of 2.7 gm/cc.

ASSIGN-ML: Regions 2 through 11 also contain material 1.

DETECTOR: Boundary crossing detectors measuring photon energy flux are located at distances corresponding to 1, 2, 4, 7, and 10 mfp. All detectors are normalized to unity using the results from the reference. The results are binned by energy. And an importance calculation for Detector 10 is requested by region and energy.

ANALYSIS: Scattering events are plotted for a 6 cm thick zone in the first quadrant of the sphere.

SOURCE: 20000 photons are started from a point at the center with energy 1 MeV and isotropic (default) angular distribution.

Notice initially there are no WALK-XX blocks, so this problem will initially be running in the default analog (unbiased) mode.

A summary of the detector results for the base calculation is given in Table 1. The last column is the figure-of-merit. Repeating the calculation would get similar numbers, but not the same results because of the different random numbers used.

Table 1

Detector	Response	FOM
1	0.990 ± 0.006	24100.
2	1.008 ± 0.020	2130.
4	0.976 ± 0.019	2150.
7	0.977 ± 0.061	206.
10	1.252 ± 0.338	11.0

We will not reproduce the entire COG output for this run, but just indicate some of the most important results.

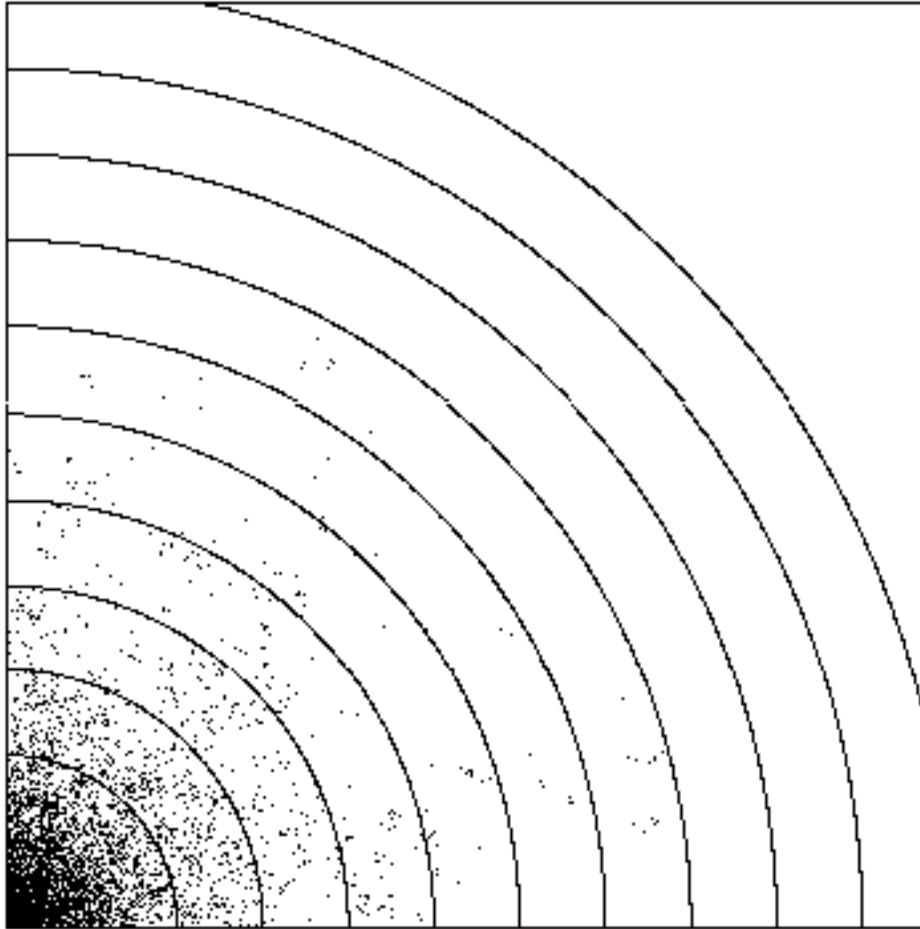
Example: ANALYSIS picture option and resulting output from the initial unbiased problem

ANALYSIS

PHOTON 0. 65. 0. 0. 0. 0. 65. 0. 0. 6.

Total Monte Carlo collisions in the limits 17795

Number plotted below 5000



- The ANALYSIS picture (above) shows few if any scattering events past 8 mfp. (The inter-ring distance is one mfp in this problem).
- The summary tables in the COG output file indicate less than 100 photons from a source of 20000 penetrate past 10 mfp.
- The output also shows that more than 60% of the total response for detector 10 comes from the 10 largest contributing particles. Contrast this to ~1% for detector 1.

Viewing the summary tables, we conclude that the results for detector 1 are excellent, for detectors 2, 4, and 7 satisfactory, and for detector 10 questionable at best.

To improve statistics for detector 10, we obviously need to get more particles to penetrate deeper. We could just run longer—this is the simplest and safest approach—and wait for the photons to reach the deepest regions by "natural" processes. But Poisson statistics tells us that we must run ~100 times longer in order to reduce the standard deviations by a factor of 10.

For our first attempt at biasing this problem, we will use particle splitting at a boundary to generate more deeply-penetrating particles. The WALK-BC biasing scheme causes particles moving in the desired direction (outwards) to be split at each boundary crossing, thus increasing the density of particles with distance.

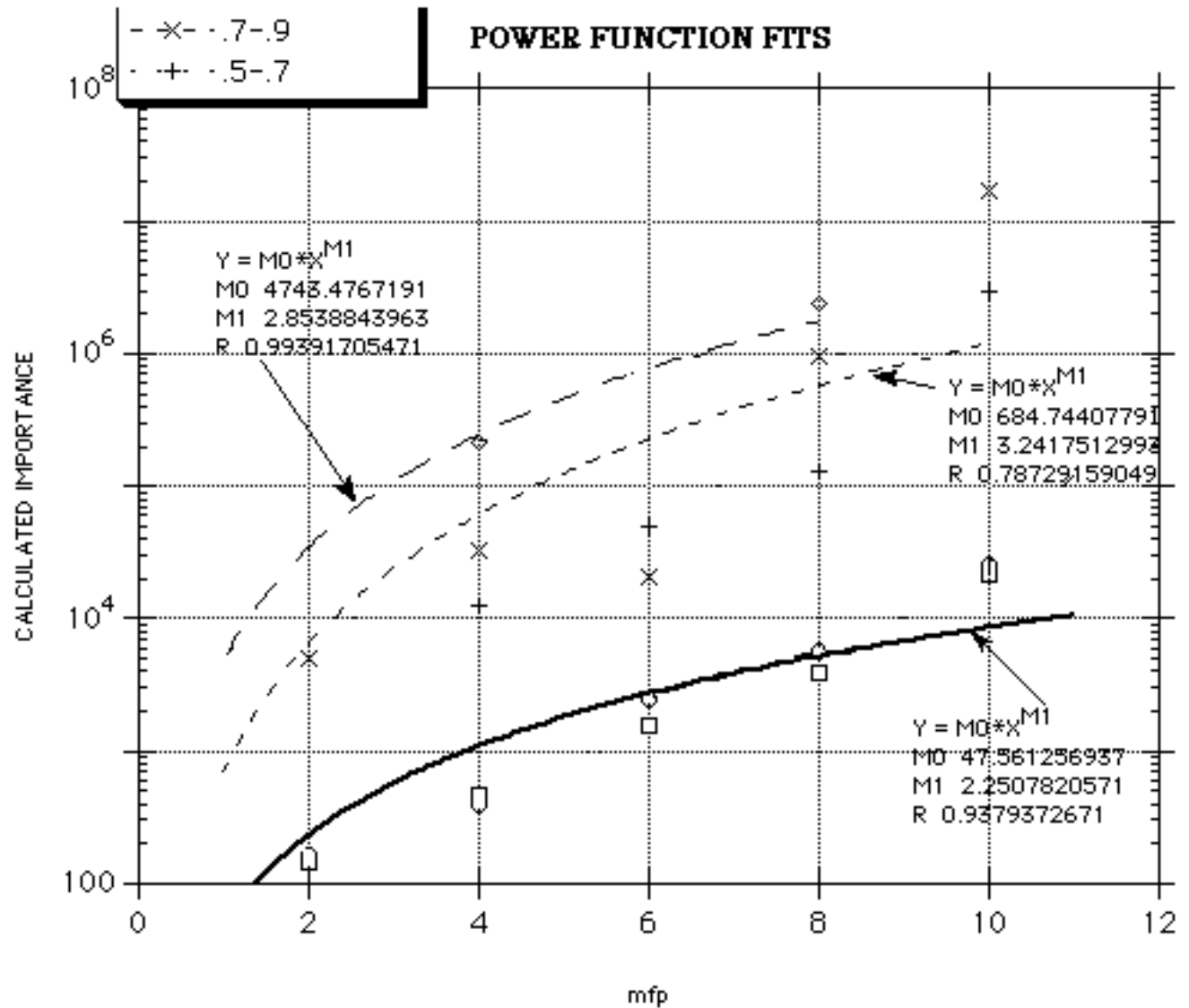
WALK-BC biasing requires us to specify the relative IMPORTANCE of phase-space cells for each of the REGIONS forming the boundary. Where do we get this information? Since we asked for an IMPORTANCE calculation for detector 10, COG will list the IMPORTANCES of each REGION to our detector 10 result. These calculated IMPORTANCES are derived from the detector response and hence are as good (or bad) as that result. Because of the somewhat poor statistics in each REGION, we will not place much confidence in the individual calculated IMPORTANCES, but rather look for trends.

We show below the results of the IMPORTANCE calculations. For a specified range of particle energy, we plot the REGIONal IMPORTANCES versus REGION distance, expressed in mfp. We have overlaid the data with a power-law approximation to each data set. The top curve is the REGIONal importance data for energy in the range [0.9,1.] MeV, the middle two curves are for energy ranges [0.7,0.9] MeV and [0.5,0.7] MeV. The bottom curve is the importance averaged over all particle energies. A power-law fit of the form:

$$I = M_0 X^{M_1}$$

is natural to associate with the WALK-BC biasing scheme due to the exponential attenuation of flux with distance.

These approximations aren't very good, but they indicate that importances change by a factor of 2.25 to 3.25 per mfp. Another way to approach the problem is to plot the natural log of importance versus distance (measured in mfp at 1 MeV). This plot yields an approximate straight line fit with slope ≈ 0.92 e-folds/mfp. This implies that the splitting parameters for WALK-BC biasing at 1 mfp intervals are in an approximately constant ($e^{0.92} \approx 2.5$) ratio.



We can now rerun the problem with splitting at a boundary crossing. We use a WALK-BC Data Block with the IMPORTANCE estimated from our fits.

WALK-BC

PHOTON	REGION 1	ENERGY 0. TO 1.	1.
	REGION 2	ENERGY 0. TO 1.	2.5
	REGION 3	ENERGY 0. TO 1.	6.3
	REGION 4	ENERGY 0. TO 1.	16.
	REGION 5	ENERGY 0. TO 1.	39.
	REGION 6	ENERGY 0. TO 1.	98.
	REGION 7	ENERGY 0. TO 1.	240.
	REGION 8	ENERGY 0. TO 1.	610.
	REGION 9	ENERGY 0. TO 1.	1500.
	REGION 10	ENERGY 0. TO 1.	3800.
	REGION 11	ENERGY 0. TO 1.	3800.

Since the total response is roughly independent of energy, to keep the problem simple we have used energy-independent IMPORTANCES.

The detector results are listed in Table 2 for this WALK-BC run. Note that the figure-of-merit has decreased for detectors 1 through 7 and increased for detector 10. This is to be expected since the problem is being "tuned" for detector 10.

Table 2

Detector	Response	FOM
1	0.970 ± 0.022	461.
2	0.989 ± 0.032	235.
4	1.041 ± 0.050	106.
7	1.053 ± 0.072	52.1
10	1.132 ± 0.096	33.2

- The ANALYSIS picture now shows scattering events over the whole range of interest.
- The summary tables indicate almost 5000 photons from a source of 20000 penetrate past 10 mfp.
- Less than 20% of the total response for detector 10 comes from the 10 largest contributors.

The results (for detector 10) are greatly improved; the standard deviations are less and the figure-of-merits (FOM) are better. Getting more particles into the outer regions will also improve the statistics on the calculated detector IMPORTANCES. Feeding these better values of IMPORTANCE back into the problem should lead to still better results. We will run this problem once again, this time using the actual calculated

detector IMPORTANCES as the REGIONal IMPORTANCES in the WALK-BC Data Block.

With these calculated detector IMPORTANCES, the WALK-BC Data Block is now:

```
WALK-BC
PHOTON  REGION 1  ENERGY 0. TO 1.  0.081
        REGION 2  ENERGY 0. TO 1.  0.25
        REGION 3  ENERGY 0. TO 1.  0.34
        REGION 4  ENERGY 0. TO 1.  0.59
        REGION 5  ENERGY 0. TO 1.  1.2
        REGION 6  ENERGY 0. TO 1.  2.4
        REGION 7  ENERGY 0. TO 1.  5.4
        REGION 8  ENERGY 0. TO 1.  14.
        REGION 9  ENERGY 0. TO 1.  36.
        REGION 10 ENERGY 0. TO 1.  120.
        REGION 11 ENERGY 0. TO 1.  76.
```

The SOURCE size (NPART) has been reduced to 5000 to keep the running time manageable.

The detector results are listed in Table 3; note that the figure-of-merit has increased for all detectors over the previous run.

Table 3

Detector	Response	FOM
1	1.005 ± 0.011	1270.
2	1.011 ± 0.016	644.
4	1.022 ± 0.028	210.
7	0.992 ± 0.036	121.
10	0.943 ± 0.044	74.5

- The ANALYSIS picture still shows scattering events in all regions.
- The summary tables now indicate more than 1200 photons from a source of 5000 penetrate past 10 mfp.
- And only about 7% of the total response for detector 10 comes from the 10 largest contributors.

We could iterate on the splitting parameters once again, but any improvement would be marginal and probably lost in the statistical noise. There is still room for improvement however. We note that, for all detectors, particles which scatter many times represent a very small fraction of the total response—e.g. for detector 10,

particles which have scattered 15 or more times contribute less than 1% to the result. We should be able to improve the figure-of-merit by playing Russian roulette to reduce the number of particles which have many scatters and hence reduce the running time.

We add to our input file the WALK-COLLISION Data Block, with parameters set to Russian-roulette particles in all REGIONS with energies in the range [0.,1.]. The value of β is 0.9.

WALK-COLLISION

PHOTON REGION ALL ENERGY 0. TO 1. 0.9

This walk modification is done everywhere since in this case we are trying to improve the results for all the detectors. A rather modest Russian roulette parameter, 0.9, is used so as not to seriously perturb

the particle distribution—a small β will eliminate rather than reduce the number of many-scattered particles.

The new results are listed in Table 4; the figure-of-merit has indeed increased for all detectors over the previous run.

Table 4

Detector	Response	FOM
1	0.995 ± 0.009	2890.
2	1.017 ± 0.016	1110.
4	1.036 ± 0.024	502.
7	1.042 ± 0.037	206.
10	1.045 ± 0.054	99.5

We now have a good result at a cost of less than 2 hours computer time.

11.3.2 Example 2—Path Stretching

This problem—as in Example 1—consists of a sphere of Al with a point source of 1 MeV gammas at the center, emitting isotropically. The radius of the sphere is more than 20 mean free paths (mfp) for the 1 MeV gammas. The energy build up factors—the ratios of the total energy to the uncollided energy—have been calculated using another (non-Monte-Carlo) method⁽¹⁾ for radii corresponding to 1, 2, 4, 7, 10, 15, and 20 mfp. We have placed a BOUNDARY-CROSSING detector at each of these radii. Each of the COG detectors is normalized by dividing the computed detector score by the "correct" result at each location (obtained from another method). Thus if COG calculates the "correct" value of flux at a detector position, the detector response will equal unity.

In this case we will attempt a much more difficult problem, that of getting a good result at 20 mfp. We estimate 3 *years* of continual running in the analog mode to get 100 particles to penetrate to 20 mfp; this is definitely a problem requiring random-walk biasing.

The input to COG, including the path stretching option, is as follows:

```
ENERGY BUILD UP IN AL @ 1 MEV (NYO-3075)
$  V31.2A
BASIC
      PHOTON
GEOMETRY
      SECTOR  1  Z01      -1
      SECTOR  2  Z02      1 -2
      SECTOR  3  Z03      2 -3
      SECTOR  4  Z04      3 -4
      SECTOR  5  Z05      4 -5
      SECTOR  6  Z06      5 -6
      SECTOR  7  Z07      6 -7
      SECTOR  8  Z08      7 -8
      SECTOR  9  Z09      8 -9
      SECTOR 10  Z10      9 -10
      SECTOR 11  Z11     10 -11
      SECTOR 12  Z12     11 -12
      SECTOR 13  Z13     12 -13
      SECTOR 14  Z14     13 -14
      SECTOR 15  Z15     14 -15
      SECTOR 16  Z16     15 -16
      SECTOR 17  Z17     16 -17
```

```

        SECTOR 18 Z18      17 -18
        SECTOR 19 Z19      18 -19
        SECTOR 20 Z20      19 -20
        SECTOR 21 Z21      20 -21
        BOUNDARY VACUUM 21
SURFACES
    1  S 6.033
    2  S 12.066
    3  S 18.1
    4  S 24.133
    5  S 30.166
    6  S 36.199
    7  S 42.232
    8  S 48.266
    9  S 54.299
   10 S 60.332
   11 S 66.365
   12 S 72.398
   13 S 78.432
   14 S 84.465
   15 S 90.498
   16 S 96.531
   17 S 102.564
   18 S 108.598
   19 S 114.631
   20 S 120.664
   21 S 150.83
MIX
    MAT 1  AL 2.7
ASSIGN-ML
    1  2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21
DETECTOR
$  ALL DETECTORS ARE NORMALIZED TO 1.
    NUMBER 1
    BOUNDARY  1 2 7.3944E-1
    DRF-E  PHOTON  ENERGY-FLUX
    NUMBER 2
    BOUNDARY  2 3 4.4525E-1
    DRF-E  PHOTON  ENERGY-FLUX

```

NUMBER 4
 BOUNDARY 4 5 1.1942E-1
 DRF-E PHOTON ENERGY-FLUX

NUMBER 7
 BOUNDARY 7 8 1.1809E-2
 DRF-E PHOTON ENERGY-FLUX

NUMBER 10
 BOUNDARY 10 11 9.534E-4
 DRF-E PHOTON ENERGY-FLUX

NUMBER 15
 BOUNDARY 15 16 1.1502E-5
 DRF-E PHOTON ENERGY-FLUX

NUMBER 20
 BOUNDARY 20 21 1.1975E-7
 DRF-E PHOTON ENERGY-FLUX
 BIN E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 1 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 2 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 3 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 4 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 5 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 6 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 7 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 8 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 9 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 10 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 11 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 12 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 13 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 14 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 15 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 16 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 17 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
 IMP REGION 18 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.

```
IMP REGION 19 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 20 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
IMP REGION 21 E PHOTON 0. 0.1 0.3 0.5 0.7 0.9 1.
ANALYSIS
  PHOTON 0. 125. 0. 0. 0. 0. 125. 0. 0. 10.
WALK-PS
  PHOTON REG 1 ENG 0.999 TO 1. -0.95 POINT 0. 0. 0.
SOURCE
  NPART 20000
  DEFINE P 1
    POINT 0. 0. 0.
  DEFINE E 1
    PHOTON LINE 1. 1.
  INC 1. P 1 E 1
END
```

SUMMARY OF PROBLEM INPUT FILE

First line: Problem title and comments.

BASIC: Photon only problem.

GEOMETRY: 21 regions bounded by 21 spherical surfaces. The vacuum boundary tells COG to not follow particles which cross surface 21.

SURFACES: 21 concentric spherical surfaces, all except the last at 1 mfp intervals.

MIX: Region 1 contains material 1, Al at a density of 2.7 gm/cc.

ASSIGN-ML: Regions 2 through 21 also contain material 1.

DETECTOR: Boundary crossing detectors measuring photon energy flux are located at distances corresponding to 1, 2, 4, 7, 10, 15, and 20 mfp. All detectors are normalized to unity using the results from the reference. The results are binned by energy. And an importance calculation for Detector 20 is requested by region and energy.

ANALYSIS: Scattering events are plotted for a 10 cm thick zone in the first quadrant of the sphere.

WALK-PS: Path stretching by a factor of 20 ($q=-0.95$) is done in all directions away from the source (the center point of the sphere). This is necessary to get some result, and therefore a calculated IMPORTANCE, for DETECTOR 20.

SOURCE: 20000 photons are started from a point at the center with energy 1 MeV and isotropic (default) angular distribution.

The DETECTOR results are listed in Table 1. The last column is the figure-of-merit, FOM.

Table 1		
Detector	Response	FOM
1	1.035 ± 0.024	954.
2	1.017 ± 0.028	676.
4	0.996 ± 0.046	242.
7	0.961 ± 0.107	40.8
10	0.841 ± 0.210	8.11
15	0.411 ± 0.101	8.44
20	0.207 ± 0.052	8.12

We will not reproduce the entire COG output for this run, but just indicate some of the most important results.

- The ANALYSIS picture shows that we are getting interactions in and around 20 mfp.
- The summary tables indicate almost 8000 photons from a source of 20000 reach 20 mfp.

This is encouraging information, but we should realize that the ANALYSIS picture and the summary tables don't indicate the weights of the scoring particles. If we look at the detector results for detector 20, we see:

- Almost 60% of the Detector 20 result comes from the 10 largest contributors; i.e. we have poor statistics on this answer.

Let us now use the IMPORTANCE information for detector 20, as calculated by COG, to come up with improved path-stretching parameters. The output of the IMPORTANCE calculation is the relative IMPORTANCE of each phase-space cell (in REGION and ENERGY), and also the relative IMPORTANCE of each REGION (independent of energy). Since we do not have much confidence in the IMPORTANCES obtained from this run (because of the poor detector statistics), we will use just the average REGION IMPORTANCES. What we shall do is to plot the $\ln$ of REGION IMPORTANCE versus distance (measured in mfp at 1 MeV). We then determine the slope of this "straight" line. The stretching parameter q , for each REGION, is then given by

$$q = 1 - 1/(\text{slope} \times \text{distance})$$

where **slope** is the slope of the line, and **distance** is the average distance from the REGION to the detector (in mfp at 1 MeV). Set any negative q 's to zero.

Let's now rerun the problem with these values of q .

WALK-PS

```

PHOTON  REG 1  ENG 0. TO 1.  -0.9469 POINT 0. 0. 0.
        REG 2  ENG 0. TO 1.  -0.946 POINT 0. 0. 0.
        REG 3  ENG 0. TO 1.  -0.9428 POINT 0. 0. 0.
        REG 4  ENG 0. TO 1.  -0.9393 POINT 0. 0. 0.
        REG 5  ENG 0. TO 1.  -0.9352 POINT 0. 0. 0.
        REG 6  ENG 0. TO 1.  -0.9306 POINT 0. 0. 0.
        REG 7  ENG 0. TO 1.  -0.9252 POINT 0. 0. 0.
        REG 8  ENG 0. TO 1.  -0.919 POINT 0. 0. 0.
        REG 9  ENG 0. TO 1.  -0.9117 POINT 0. 0. 0.
        REG 10  ENG 0. TO 1.  -0.9028 POINT 0. 0. 0.
        REG 11  ENG 0. TO 1.  -0.892 POINT 0. 0. 0.
        REG 12  ENG 0. TO 1.  -0.8785 POINT 0. 0. 0.
        REG 13  ENG 0. TO 1.  -0.8612 POINT 0. 0. 0.
        REG 14  ENG 0. TO 1.  -0.838 POINT 0. 0. 0.
        REG 15  ENG 0. TO 1.  -0.8056 POINT 0. 0. 0.
        REG 16  ENG 0. TO 1.  -0.757 POINT 0. 0. 0.
        REG 17  ENG 0. TO 1.  -0.6761 POINT 0. 0. 0.
        REG 18  ENG 0. TO 1.  -0.5141 POINT 0. 0. 0.
        REG 19  ENG 0. TO 1.  -0.0282 POINT 0. 0. 0.
        REG 20  ENG 0. TO 1.  0. POINT 0. 0. 0.
        REG 21  ENG 0. TO 1.  0.0282 POINT 0. 0. 0.

```

The detector results are listed in Table 2.

Table 2

Detector	Response	FOM
1	0.970 ± 0.045	202.
2	0.895 ± 0.049	151.
4	0.806 ± 0.049	121.
7	1.142 ± 0.243	9.75
10	0.742 ± 0.127	15.0
15	0.923 ± 0.198	9.62
20	0.826 ± 0.175	9.82

- The summary tables now indicate 9000 photons from a source of 20000 reach 20 mfp.

- 42% of the Detector 20 result comes from the 10 largest contributors.

This is still a poor result, but the IMPORTANCE calculation is now more reliable with regards to energy. Let us now plot, for each energy group, in (IMPORTANCE)

versus distance (measured in mfp at 1 MeV). We then determine the slopes of each of these "straight" lines. As before, q is obtained from:

$$q = 1 - 1/(\text{slope} \times \text{distance})$$

where **slope** is the slope of the line (and contains the energy information), and **distance** is the average distance from the REGION to the detector (in mfp at 1 MeV). Set any negative q 's to zero. We now place these REGION- and ENERGY-dependent values of q into the WALK-PS Data Block, and rerun our problem:

WALK-PS

```

PHOTON REG 1 ENG 0. TO 0.1  -0.9831 POINT 0. 0. 0.
                ENG 0.1 TO 0.3  -0.9761 POINT 0. 0. 0.
                ENG 0.3 TO 0.5  -0.9788 POINT 0. 0. 0.
                ENG 0.5 TO 0.7  -0.956  POINT 0. 0. 0.
                ENG 0.7 TO 0.9  -0.944  POINT 0. 0. 0.
                ENG 0.9 TO 1.   -0.941  POINT 0. 0. 0.
REG 2 ENG 0. TO 0.1  -0.9822 POINT 0. 0. 0.
                ENG 0.1 TO 0.3  -0.9747 POINT 0. 0. 0.
                ENG 0.3 TO 0.5  -0.9776 POINT 0. 0. 0.
                ENG 0.5 TO 0.7  -0.9535 POINT 0. 0. 0.
                ENG 0.7 TO 0.9  -0.9409 POINT 0. 0. 0.
                ENG 0.9 TO 1.   -0.9377 POINT 0. 0. 0.
REG 3 ENG 0. TO 0.1  -0.9812 POINT 0. 0. 0.
                ENG 0.1 TO 0.3  -0.9733 POINT 0. 0. 0.
                ENG 0.3 TO 0.5  -0.9763 POINT 0. 0. 0.
                ENG 0.5 TO 0.7  -0.9508 POINT 0. 0. 0.
                ENG 0.7 TO 0.9  -0.9374 POINT 0. 0. 0.
                ENG 0.9 TO 1.   -0.9341 POINT 0. 0. 0.
REG 4 ENG 0. TO 0.1  -0.98  POINT 0. 0. 0.
                ENG 0.1 TO 0.3  -0.9716 POINT 0. 0. 0.
                ENG 0.3 TO 0.5  -0.9746 POINT 0. 0. 0.
                ENG 0.5 TO 0.7  -0.9477 POINT 0. 0. 0.
                ENG 0.7 TO 0.9  -0.9335 POINT 0. 0. 0.
                ENG 0.9 TO 1.   -0.93  POINT 0. 0. 0.
REG 5 ENG 0. TO 0.1  -0.9787 POINT 0. 0. 0.
                ENG 0.1 TO 0.3  -0.9697 POINT 0. 0. 0.
                ENG 0.3 TO 0.5  -0.9731 POINT 0. 0. 0.
                ENG 0.5 TO 0.7  -0.9443 POINT 0. 0. 0.
                ENG 0.7 TO 0.9  -0.929  POINT 0. 0. 0.
                ENG 0.9 TO 1.   -0.9253 POINT 0. 0. 0.

```



```
REG 6 ENG 0. TO 0.1 -0.9771 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.9675 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.9712 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.9403 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.924 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.92 POINT 0. 0. 0.
REG 7 ENG 0. TO 0.1 -0.9754 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.965 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.969 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.9357 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.9181 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.9138 POINT 0. 0. 0.
REG 8 ENG 0. TO 0.1 -0.9733 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.9621 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.9664 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.9303 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.9113 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.9066 POINT 0. 0. 0.
REG 9 ENG 0. TO 0.1 -0.9709 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.9587 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.9634 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.924 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.9032 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.8981 POINT 0. 0. 0.
REG 10 ENG 0. TO 0.1 -0.968 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.9545 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.9597 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.9164 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.8935 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.8879 POINT 0. 0. 0.
REG 11 ENG 0. TO 0.1 -0.9644 POINT 0. 0. 0.
      ENG 0.1 TO 0.3 -0.9495 POINT 0. 0. 0.
      ENG 0.3 TO 0.5 -0.9552 POINT 0. 0. 0.
      ENG 0.5 TO 0.7 -0.9071 POINT 0. 0. 0.
      ENG 0.7 TO 0.9 -0.8817 POINT 0. 0. 0.
      ENG 0.9 TO 1. -0.8755 POINT 0. 0. 0.
```

```
REG 12 ENG 0. TO 0.1  -0.96 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.9432 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.9496 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.8955 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.8669 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.8599 POINT 0. 0. 0.
REG 13 ENG 0. TO 0.1  -0.9543 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.935 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.9424 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.8806 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.8479 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.8399 POINT 0. 0. 0.
REG 14 ENG 0. TO 0.1  -0.9466 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.9242 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.9328 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.8606 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.8226 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.8132 POINT 0. 0. 0.
REG 15 ENG 0. TO 0.1  -0.936 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.9091 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.9194 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.8328 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.7871 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.7759 POINT 0. 0. 0.
REG 16 ENG 0. TO 0.1  -0.9199 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.8863 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.8992 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.791 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.7339 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.7199 POINT 0. 0. 0.
REG 17 ENG 0. TO 0.1  -0.8933 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.8484 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.8656 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.7213 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.6452 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.6265 POINT 0. 0. 0.
```

```
REG 18 ENG 0. TO 0.1  -0.8399 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.7726 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.7985 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.5819 POINT 0. 0. 0.
      ENG 0.7 TO 0.9  -0.4677 POINT 0. 0. 0.
      ENG 0.9 TO 1.   -0.4397 POINT 0. 0. 0.
REG 19 ENG 0. TO 0.1  -0.6798 POINT 0. 0. 0.
      ENG 0.1 TO 0.3  -0.5453 POINT 0. 0. 0.
      ENG 0.3 TO 0.5  -0.5969 POINT 0. 0. 0.
      ENG 0.5 TO 0.7  -0.1639 POINT 0. 0. 0.
      ENG 0.7 TO 0.9   0. POINT 0. 0. 0.
      ENG 0.9 TO 1.   0. POINT 0. 0. 0.
REG 20 ENG 0. TO 0.1   0. POINT 0. 0. 0.
      ENG 0.1 TO 0.3   0. POINT 0. 0. 0.
      ENG 0.3 TO 0.5   0. POINT 0. 0. 0.
      ENG 0.5 TO 0.7   0. POINT 0. 0. 0.
      ENG 0.7 TO 0.9   0. POINT 0. 0. 0.
      ENG 0.9 TO 1.   0. POINT 0. 0. 0.
REG 21 ENG 0. TO 0.1   0.6798 POINT 0. 0. 0.
      ENG 0.1 TO 0.3   0.5453 POINT 0. 0. 0.
      ENG 0.3 TO 0.5   0.5969 POINT 0. 0. 0.
      ENG 0.5 TO 0.7   0.1639 POINT 0. 0. 0.
      ENG 0.7 TO 0.9   0. POINT 0. 0. 0.
      ENG 0.9 TO 1.   0. POINT 0. 0. 0.
```

The detector results for this run are listed in Table 3. As expected, using walk modification based on IMPORTANCES for Detector 20 has improved the results there at the expense of results at other detector locations.

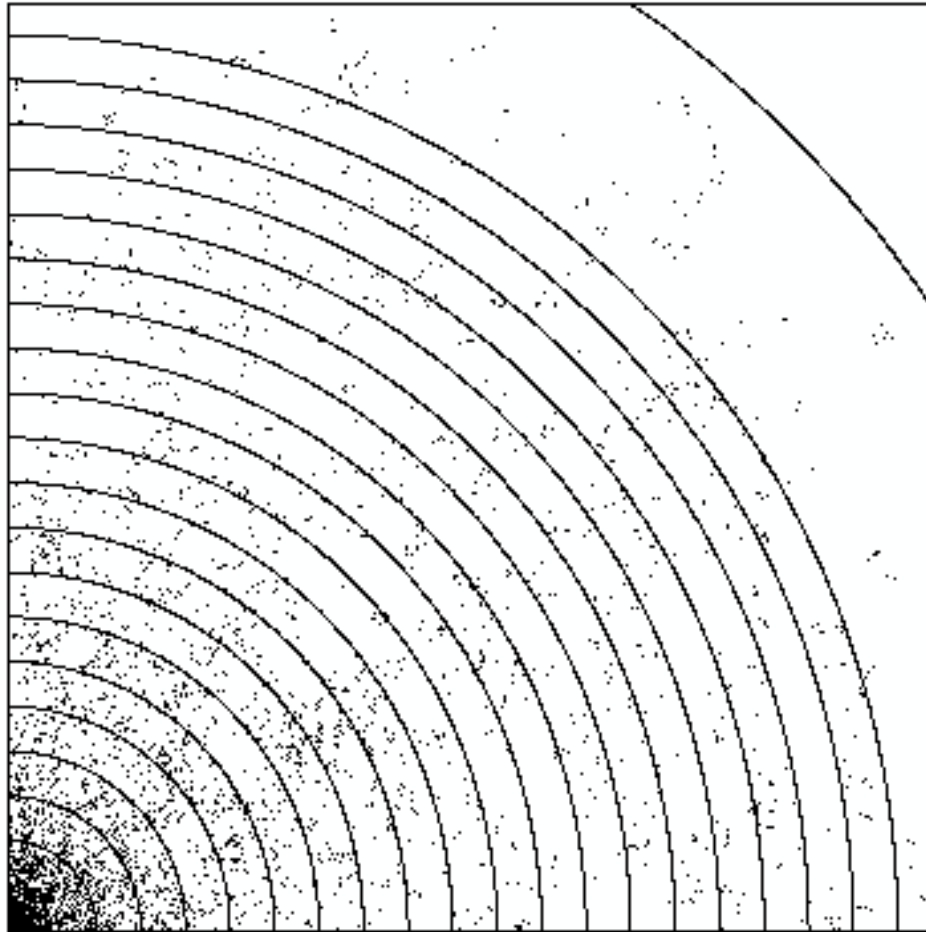
Table 3

Detector	Response	FOM
1	1.006 ± 0.035	314.
2	1.008 ± 0.055	124.
4	0.953 ± 0.065	78.4
7	1.191 ± 0.149	23.6
10	1.608 ± 0.567	2.97
15	0.784 ± 0.117	16.5
20	0.790 ± 0.084	32.3

- The summary tables now indicate about 7000 photons from a source of 20000 reach 20 mfp.
- But only about 28% of the Detector 20 result comes from the 10 largest contributors.

This is still not a good result, but considering it was obtained in less than 3 minutes—as opposed to 3 years—it is quite reasonable estimate. Once again we could iterate on the IMPORTANCES, but we don't think that would be too enlightening. We will instead use another walk modification technique to try to improve the result of this example.

An ANALYSIS picture showing the results of the calculation of this problem with 4257 particles plotted out of 20,000 started. As can be seen, the collisions are reasonably well distributed throughout the volume.



If we could keep the photons moving radially outward, then we should get more photons to penetrate further and hence get a better result for the deeper detectors. Scattering Direction Bias is designed for this purpose. We add to our input file a WALK-SDB Data Block:

```
WALK-SDB
PHOTON REG ALL ENG 0. TO 1. 100 SPHERE 120.664 0. 0. 0.
```

The rather modest value of 100 for **b** was chosen for two reasons. First, so as to not seriously disturb the distribution of photons. And second, because it is difficult (and expensive) to dramatically influence photon distributions, we felt it might be counterproductive to use a large value for **b**. The detector results are listed in Table 4.

Table 4

Detector	Response	FOM
1	0.908 ± 0.088	12.8
2	0.824 ± 0.048	36.1
4	0.790 ± 0.060	20.9
7	0.855 ± 0.057	26.8
10	0.913 ± 0.090	12.5
15	0.922 ± 0.076	17.8
20	0.998 ± 0.056	38.6

- The summary tables now indicate about 23000 photons from a source of 20000 reach 20 mfp.

- And less than 13% of the Detector 20 result comes from the 10 largest contributors.

This is now a good result—using simple techniques—for a problem which essentially cannot be solved by analog means.

11.3.3 Example 3—Direction Biasing

This problem consists of a cylinder of scattering material, 3 mean free paths long by 1 mean free path in radius. The scattering material has a one-group cross section consisting of 70% absorption and 30% isotropic scattering. The cylinder is surrounded by vacuum, while in the center of the cylinder is a cup-like region of infinite absorber. A source is located below the cup and a detector is located in the cup (see Figure 1). Because of the vacuum leakage and the infinite absorber, the only paths from the source to the detector are into the scattering material, around the cup, and back into the cup-mouth.

The input file is:

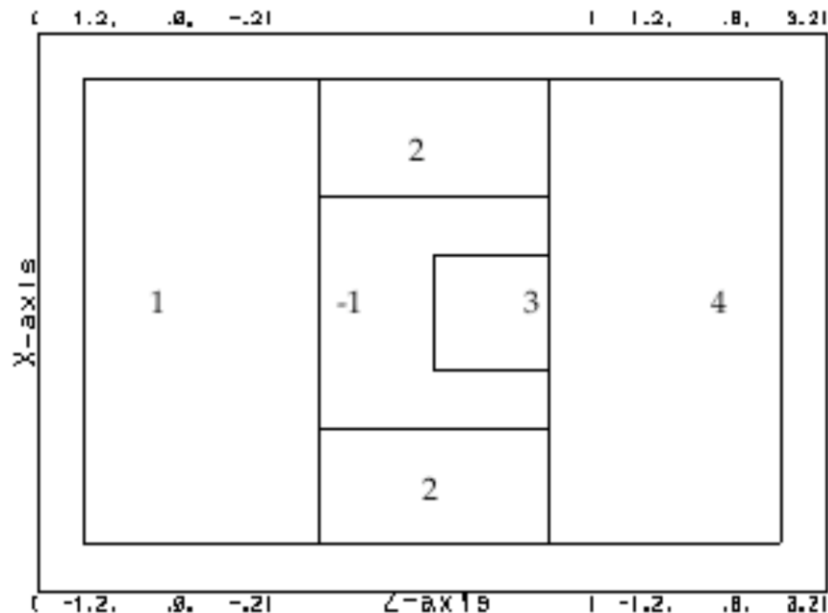
```
$ V31.3A
BASIC
  NEUTRON
GEOMETRY
  SECTOR 1 SO-SCAT      1 -2 -13
  SECTOR 2 RING-SC      2 -4 12 -13
  SECTOR 3 DET          3 -4 -11
  SECTOR 4 BK-SCAT      4 -5 -13
  SECTOR -1 ABS         2 -3 -12 OR 3 -4 11 -12
  BOUNDARY VACUUM      -1 OR 5 OR 1 -5 13
SURFACES
  1 P 0. 0. 0. 0. 0. 10.
  2 P 0. 0. 1. 0. 0. 10.
  3 P 0. 0. 1.5 0. 0. 10.
  4 P 0. 0. 2. 0. 0. 10.
  5 P 0. 0. 3. 0. 0. 10.
  11 C 0.25 TR 0. 0. 0. 0. 0. 1.
  12 C 0.5 TR 0. 0. 0. 0. 0. 1.
  13 C 1. TR 0. 0. 0. 0. 0. 1.
MIX
  MAT 1 C*3 1.6606E10
ASSIGN-ML
  1 2 3 4
ANALYSIS
  NEUTRON 1.2 0. -.2 -1.2 0. -.2 -1.2 0. 3.2 0.1
```

```

DETECTOR
  NUMBER 1
  POINT 0. 0. 1.75
  DRF-E NEUTRON NUMBER-FLUX
  IMP A 0. 0. 1. -1.
    -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1
    0. 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.
SOURCE
  NPART 200000
  DEFINE P 1
    POINT 0. 0. 0.5
  DEFINE E 1
    NEUTRON LINE 1. 1.
  INC 1. P 1 E 1
END

```

Below is a sketch of the problem, from the PICTURE command. The source is in REGION 1, REGION 2 is an annular ring, REGION 3 is the detector REGION, REGION 4 is a cylinder behind the detector. REGION -1 is a perfect absorber. The only way particles can get to REGION 3 is to scatter in REGION 2 and 4 and then back scatter into REGION 3.



SUMMARY OF PROBLEM INPUT FILE

First line: Problem title and comments.

BASIC: Neutron only problem.

GEOMETRY: Cylinder, 3 cm long by 1 cm radius, with cup, 1 cm long by 1/2 cm radius by 1/2 cm deep, located at center.

SURFACES: Planes at $z = 0, 1, 3/2, 2$, and 3 cm. Cylinders, axis along z -axis, with radii = 1/4, 1/2, and 1 cm.

MIX: Region 1 contains material 1, a special material consisting of a 1 group cross section with 70% absorption and 30% isotropic, elastic scattering. The data is available on the neutron library COGYXS.

ASSIGN-ML: Regions 2, 3, and 4 also contain material 1.

ANALYSIS: Scattering events are plotted for a 0.1 cm thick zone at the center of the figure.

DETECTOR: A boundary detector between regions 3 and 4 measures the neutron number flux. An IMPORTANCE calculation on the angle from the source is done.

SOURCE: 200000 neutrons are started from a point at $z = 0.5$ cm with energy 1 MeV and isotropic (default) angular distribution.

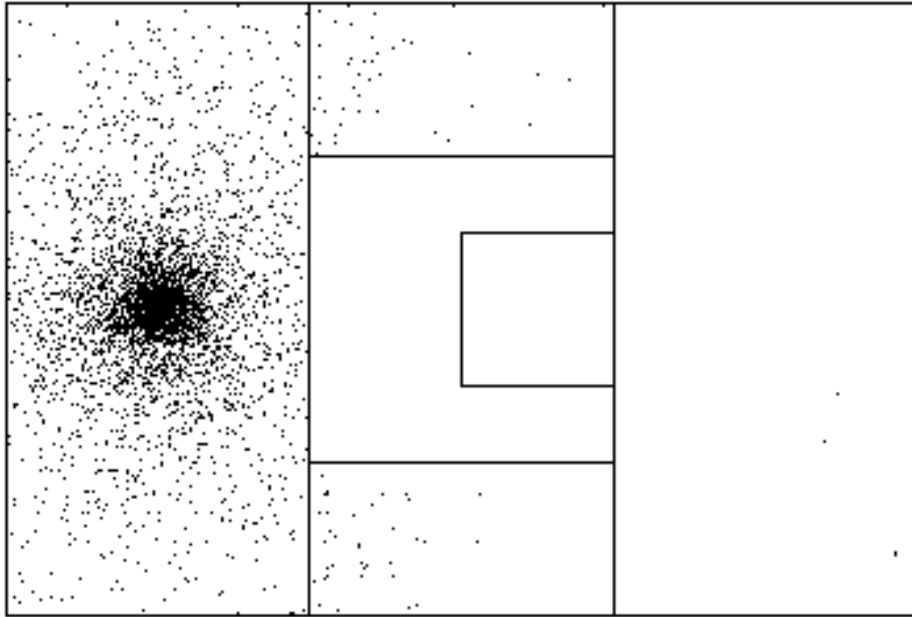
Because there are no WALK-XX Data Blocks, this problem will run in the default analog mode.

The results for the detector are

Detector	Response	FOM
1	$1.928\text{e-}4 \pm 8.41\text{e-}5$	0.89

We summarize the COG results for this run:

- The ANALYSIS picture, shown below, indicates few events in region 4 and none in region 3, the DETECTOR region.



- The summary tables indicate that indeed only 8 particles reach region 3.
- The fractional standard deviation is large, $\approx 44\%$.
- The 10 largest contributors result in 100% of the total response.

This is not a good result. The IMPORTANCE-by-ANGLE calculated results at the detector (where ANGLE represents the initial direction from the source) are determined by so few scoring events that it is probably best not to use them. We need to get more scattering events in the regions 3 and 4. We do this by biasing the emission direction of particles at the source. We assume the importance peaks for directions toward the center of the channel around the cup, i.e. for $m \approx 0.55$, and drops linearly to zero for $m = -1$ and 1 . We redefine the SOURCE Data Block to add IMPORTANCES to the ANGLE-dependent SOURCE:

```
SOURCE
  NPART 200000
  DEFINE P 1
    POINT 0. 0. 0.5
  DEFINE E 1
    NEUTRON LINE 1. 1.
  DEFINE A 1
    0. 0. 1. ISOTROPIC
```

IMPORTANCE

```
-1. 0.00323 -0.9 0.00968 -0.8 0.01613 -0.7 0.02258
-0.6 0.02903 -0.5 0.03548 -0.4 0.04194 -0.30.04834
-0.2 0.05484 -0.1 0.06129 0. 0.06774 0.1 0.07419 0.2
0.08065 0.3 0.08710 0.4 0.09355 0.5 0.09642 0.6
0.07778 0.7 0.05556 0.8 0.03333 0.9 0.01111 1.
```

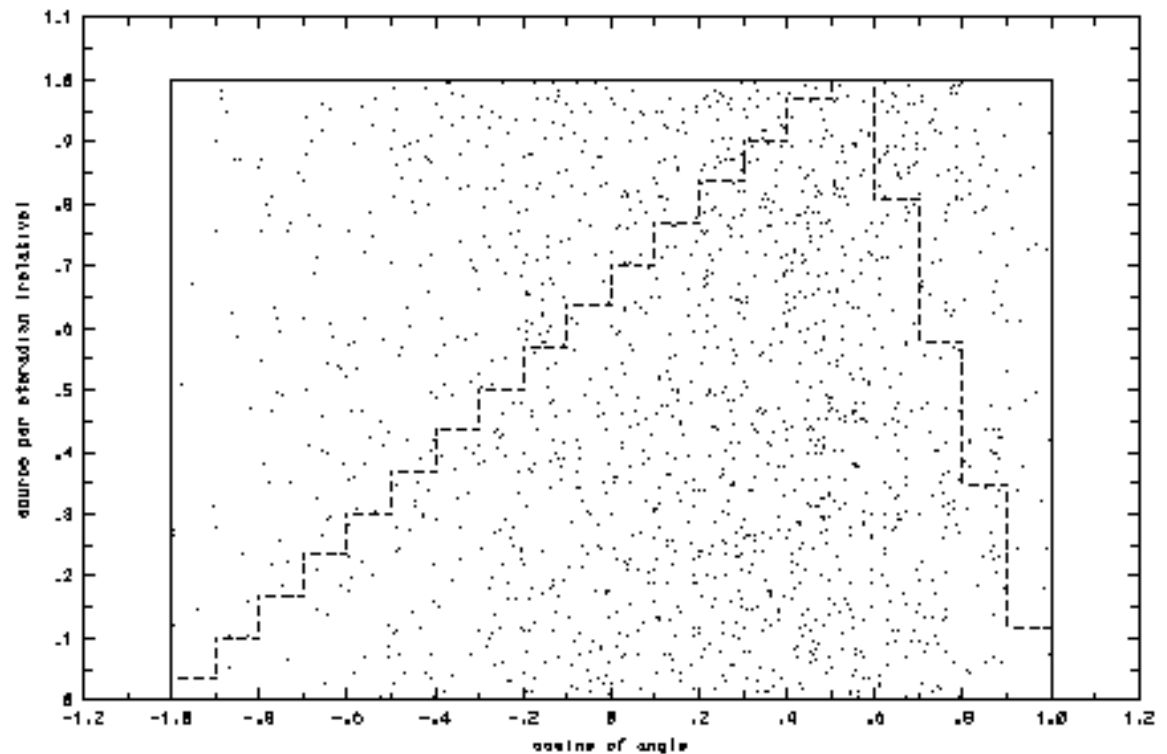
```
INC 1. P 1 E 1 A 1
```

In addition we will use scattering direction bias to guide the particles around the infinite absorber and into the detector region. The scattering direction bias parameters are chosen based on the geometry. In region 1—below the cup in the source region—the preferred direction for scattering is toward a ring with radius 0.75 cm at $z = 1$ cm, i.e. toward the mouth of the channel around the cup. The bias strength parameter is such that the half-angle about this direction is about 10 degrees (see the section **WALK-SDB—Scattering Direction Bias**). In region 2—the channel region—the particles are scattered preferentially in the z direction with half-angle of about 5 degrees. In region 4—above the cup—the scattering is toward a point at $z = 2$ cm, i.e. toward the mouth of the cup. The half-angle is again about 10 degrees. We add to our input file the WALK-SDB Data Block:

WALK-SDB

```
NEUTRON REG 1 ENG 0. TO 1. 100. RING 0.75 0 0 1 0
0 2
REG 2 ENG 0. TO 1. 1000. DIRECTION 0 0 1
REG 4 ENG 0. TO 1. 100. POINT 0 0 2
```

Below is COG's representation of the ANGLE-dependent SOURCE. The histogram shows the user-specified IMPORTANCEs, while the density of sample points indicates the actual angular distribution used. It clearly shows the biasing peaked around $\mu = 0.55$.



The results are now

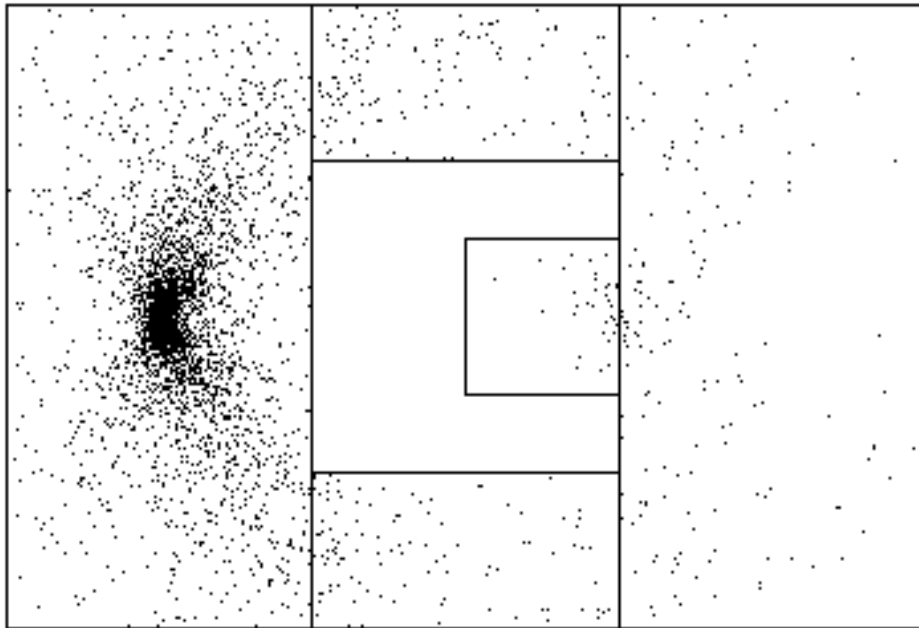
Detector	Response	FOM
1	$2.371\text{e-}4 \pm 1.76\text{e-}5$	9.86

The figure-of-merit has increased by a factor of 11.

Below is the ANALYSIS picture for this case and it clearly shows how the particles have been channeled around the infinite absorber to get improved statistics in the detector region, SECTOR 3. Remember that ANALYSIS pictures only show collision sites and tell you **nothing** about the type of collision or the particle weights involved.

We summarize the COG results for this run:

- The ANALYSIS picture now shows many more interactions in region 4, and several in region 3.



- The summary tables show 5332 particles entering region 3.
- The fractional standard deviation has decreased to about 7.4%.
- The 10 largest contributors now account for just 17.6% of the total response.

This result is much improved. Our detector IMPORTANCE calculations are also better and could now be used in the SOURCE Data Block as ANGLE-dependent SOURCE IMPORTANCES for another problem iteration, if we wished to try for improved detector statistics.

A few final words of advice when using random-walk modification techniques:

- keep things simple, so you don't lose track of what you are doing;
- sometimes combinations of walk modifications are beneficial, but add them carefully, one at a time;
- quit when you get a satisfactory result.

11.4 References

- 1.0 D. C. Irving, R. M. Freestone, Jr., and F. B. K. Kam, O5-R, A General Purpose Monte Carlo Neutron Transport Code, Oak Ridge National Laboratory, Oak Ridge, TN, ORNL-3622, (1965).
- 2.0 LASL Group TD-6, MCNP—A General Monte Carlo Code for Neutron and Photon Transport, Los Alamos National Laboratory, Los Alamos, NM, LA-7396-M, (1978).

12 The CRITICALITY Data Block

In the reactor theory of nuclear criticality, the quantity k_{eff} is the ratio of the number of neutrons born in successive generations. In a single neutron generation, neutrons which are created by fission move through the problem geometry, interacting with the materials until they either escape from the system, undergo absorption, or enter into fission events, which create new neutrons for the next neutron generation. For critical systems, $k_{\text{eff}} = 1.0$. For subcritical systems, $k_{\text{eff}} < 1.0$ and for supercritical systems, $k_{\text{eff}} > 1.0$. The purpose of a criticality calculation is to determine k_{eff} for an assembly of fissionable materials.

Determining k_{eff} consists of calculating the mean number of fission neutrons produced in one generation per fission neutron started. COG computes k_{eff} for a criticality problem by tracking a "batch" of neutrons through the geometry to simulate each successive fission generation. The sites where the batch neutrons cause fissions are saved as the starting locations (the fission source) for the next batch. The fission source will evolve from the initial distribution (specified by the user) until some equilibrium source distribution (the "settled" source) is achieved. After a user-specified number of "settling" batches are run, COG averages the batch k_{eff} over all subsequent batches.

The effect of delayed neutrons from fission may be included by including **DELAYEDNEUTRONS** in the BASIC Data Block. For **ENDL90** and **RED2002** the spectrum of delayed neutrons is assumed to be the same as the prompt fission neutrons. Processes such as (n,2n) and (n,3n) are included.

The CRITICALITY option is turned on by the inclusion of the CRITICALITY Data Block in the input file. In this block, the user must specify the initial points within the geometry at which fission neutrons will be created, for the first-generation source.

The standard SOURCE Data Block is *not* needed and will be skipped if present. COG options which require a SOURCE file will not work for CRITICALITY problems—these include the RETRACE option and the CORRELATED sampling technique.

All other options will function but, of course, many are not applicable for criticality work. Results for fluxes, currents, and detector responses may be obtained by specifying the usual detector forms in the DETECTOR Data Block. Such results will however *not* include contributions from the first (NFIRST-1) "settling" batches (see below). To calculate the prompt photon fluxes produced by fission, you should include PHOTON as a particle type in the BASIC Data Block.

The CRITICALITY Data Block has the following form:

CRITICALITY

NPART *npart-per-batch*
NBATCH *number-of-batches*
NFIRST *first-batch-to-count*
SDT *sdev-to-terminate*
NORM *normalization-factor*
NSOURCE *nsources* $x_1 y_1 z_1 x_2 y_2 z_2 \dots$

Where:

npart-per-batch is the number of neutrons to start and follow per batch (this number is exact only for the initial batch). (DEFAULT: 500)

number-of-batches is the maximum number of batches to run. The problem is terminated either by the **SDT** condition being met, or this batch limit being reached. (DEFAULT: 200)

first-batch-to-count is the ordinal number of the first batch to be used by COG to calculate an average k_{eff} , or detector fluxes or problem leakage, or to check for problem convergence. Normally a few batches must be run to allow the fission source to "settle" towards the equilibrium distribution. Results obtained from these early batches will not be accurate, so COG discards them. (DEFAULT: 6)

sdev-to-terminate is used to terminate the run when convergence of k_{eff} occurs. Convergence is said to occur if 10 consecutive batches (after *first-batch-to-count*) have a standard deviation for k_{eff} which is less than *sdev-to-terminate*. (DEFAULT: 0.005)

normalization-factor is used to normalize flux and related quantities for listing in the output file. Units are fission/sec. (DEFAULT: 1.)

nsources is the number of user-specified point sources, which make up the initial fission source. (DEFAULT: NONE, this value must be specified).

$x_1 y_1 z_1 x_2 y_2 z_2 \dots$ are the coordinates of the *nsources* points, where COG will start fission neutrons in the initial batch. (DEFAULT: NONE, *nsources* triplets must be specified).

The number and origin of the initial neutrons are given by the *nsources* at (x,y,z) locations. (The starting points for neutrons of subsequent batches are the fission-event sites of the previous batch). If the fissionable material in the problem has significant "clumps" (e.g. several fuel rods), at least one initial point source should be specified in each "clump" to speed up the convergence of the calculation. Criticality calculations initially show a significant dispersion in the values of k_{eff} ,

with the calculation normally settling down, after perhaps many batches, to a believable answer with small dispersion.

Example: A typical CRITICALITY Data Block.

```
CRITICALITY
NPART=1000
NBATCH=200
SDT=.005
NFIRST=8
NORM=1.
NSOURCE 3
  0. 0. 0.
  1. 1.1 1.2
  1. 4. 4.
```

12.1 CRITICALITY Results

The CRITICALITY output file, ...out, contains the normal "echoing" of the input—as COG interprets it—and should be examined to detect input errors. The summary tables provide a breakdown of particle random-walk events by REGION.

Example: CRITICALITY results in the out file.

```

** COG10.12          gps01          -- Version 10.12   Sep 18 2002 ***
COG  --  LISTINGS OF INPUT LINES      THU SEP 19      10:58:25 2002

[LISTING OF INPUT DECK FOLLOWED BY INPUTS AS COG INTERPRETS THEM]

BASIC PARAMETERS--INPUTS AS COG INTERPRETS THEM...

MIXTURE DEFINITIONS--INPUTS AS COG INTERPRETS THEM..

REGIONS, MEDIA, AND DENSITY FACTORS--INPUTS AS COG INTERPRETS
THEM... GEOMETRICAL SURFACE SPECIFICATIONS--INPUTS AS COG INTERPRETS
THEM... CRITICALITY CALCULATION PARAMETERS--INPUTS AS COG INTERPRETS
THEM...

DICTIONARY FILE ..... 7/23/02
/USR/GAPPS/COG/DEC/DATAX/COGLEX
NEUTRON CROSS SECTION FILE ... 5/13/02
/USR/GAPPS/COG/DEC/DATAX/ENDL90

STARTING RANDOM NUMBER SEEDS FOR TRANSPORT PHASE = 19170 4529
STARTING SEQUENCE NUMBER = 1

```

TEST CRITICALITY PROBLEM - GODIVA - JUNE 1993

K CALCULATION RESULTS

(AVERAGED FROM THE LAST BATCH TOWARDS THE FIRST BATCH.)

BATCH	BATCH K	AVE TO THIS BATCH		AVE OF LAST 5		AVE OF LAST 10	
-----	-----	-----	-----	-----	-----	-----	-----
100	0.9527	0.9527	0.9527				
99	1.0213	0.9870	0.0343				
98	0.9739	0.9826	0.0203				
97	1.0433	0.9978	0.0209				
96	1.0284	1.0039	0.0173	1.0039	0.0173		
95	1.0637	1.0139	0.0173	1.0261	0.0150		
94	0.9674	1.0072	0.0161	1.0153	0.0191		
93	0.9702	1.0026	0.0147	1.0146	0.0195		
92	1.0106	1.0035	0.0130	1.0081	0.0182		
91	1.0036	1.0035	0.0116	1.0031	0.0174	1.0035	0.0116
90	0.9914	1.0024	0.0105	0.9887	0.0087	1.0074	0.0103
11	0.9932	0.9981	0.0033	1.0093	0.0080	1.0050	0.0077
10	0.9686	0.9977	0.0033	0.9998	0.0110	1.0052	0.0076
9	0.9598	0.9973	0.0033	0.9868	0.0114	1.0036	0.0084
8	1.0051	0.9974	0.0033	0.9904	0.0119	1.0042	0.0084
7	1.0076						
6	1.0443						
5	1.1068						
4	1.0312						
3	1.0652						
2	1.1508						
1	1.3942						

AVERAGE MULTIPLICATION = 0.9974

STANDARD DEVIATION = 0.0033

Note the averages are taken backwards from the last batch (the most reliable) down to the first (least reliable) batch.

The screen output from COG gives a running value of k_{eff} and its average, e.g.

```

FINISHED BATCH 1 K= 1.3942 AVE-K= 0.0000 SD= 0.0000 NPART= 2000
FINISHED BATCH 2 K= 1.1508 AVE-K= 0.0000 SD= 0.0000 NPART= 2025
FINISHED BATCH 3 K= 1.0652 AVE-K= 0.0000 SD= 0.0000 NPART= 2013
FINISHED BATCH 4 K= 1.0312 AVE-K= 0.0000 SD= 0.0000 NPART= 2010
FINISHED BATCH 5 K= 1.1068 AVE-K= 0.0000 SD= 0.0000 NPART= 2008
FINISHED BATCH 6 K= 1.0443 AVE-K= 0.0000 SD= 0.0000 NPART= 2002
FINISHED BATCH 7 K= 1.0076 AVE-K= 0.0000 SD= 0.0000 NPART= 1990
FINISHED BATCH 8 K= 1.0051 AVE-K= 1.0051 SD= 1.0000 NPART= 2001
FINISHED BATCH 9 K= 0.9598 AVE-K= 0.9824 SD= 0.0226 NPART= 1999

```

There are normally only two CRITICALITY output files created. The first file (the ps file) contains any geometry pictures, plots of k_{eff} , plots of k_{eff} averaged over 5 batches, etc. There is also a plot of the "Fraction of Fissioning Neutrons with Energy Below E vs. E".

The second file (the out file) contains the problem listing.

The out file for CRITICALITY calculations includes:

Criticality calculation parameters: The input parameters for criticality are listed. The CRITICALITY Data Block is interpreted.

Criticality results: The individual and average multiplications are listed by batch number. This is followed in turn by:

"Fraction of Fissioning Neutrons with Energy below E vs. E", the spectrum of fissioning neutrons.

"Mean Generation Time", time from neutron birth (by fission) to absorption- producing fission.

"Prompt Neutron Lifetime", time from neutron birth (by fission) to any absorption.

"Mean Time to Escape", time from neutron birth (by fission) to escape from the problem.

"Optical Path by Region", average distance between collisions.

"Totals for the Last xxx Batches Normalized to 1 Fission" for Production, Removal, Capture, Leakage, and Fission.

"Rates and Mean-Times for the Last xxx Batches" for Production, Removal, Capture, Leakage, and Fission.

"k-eff" as Production Rate/Removal Rate

"alpha" as Production Rate – Removal Rate

12.2 Alpha Calculations

To do alpha calculations, follow the usual **CRITICALITY** input with either **ALPHA EIGENVALUE ...**, **ALPHA DYNAMIC ...**, or **ALPHA NEW...**

12.2.1 Alpha Eigenvalue

An initial value of alpha is computed from the first-order approximation to the equation of exponential increase:

$$n(t) = n(0) \cdot \exp(a \cdot t)$$

$$n(t)/n(0) = k \sim (1 + a \cdot t), \text{ where } t \text{ is the generation time, so}$$

$$a \sim (k - 1) / t$$

A k-eigenvalue solution is computed using this value of a in the pseudo-absorption term. A new value of alpha is computed from the new k, and the process is iterated until convergence.

For alpha eigenvalue (alpha static) calculations use...

ALPHA EIGENVALUE {NEVPART *np*} {NEVBATCH *nb*} {NEVFIRST *nf*} {NVITER *ni*}
{NEVITERF *nif*} {EVSOLVER *n*}

where:

np is the number of particles per batch (defaults to npart);

nb is the number of batches (defaults to nbatch);

nf is the first batch to use (defaults to nfirst);

ni is the number of iterations (defaults to 10);

nif is first iteration to use (defaults to 3);

n is 1 for first-order approximation, $\alpha \sim (K_{\text{eff}} - 1) / \text{Mean-Fission-Time}$ (default), 2 for linear fit to $K_{\text{eff}}(\alpha)$.

12.2.2 Alpha Dynamic

Using a settled criticality source, we begin tracking neutrons in shielding mode, in which a fission event does not terminate the particle history. Instead, secondary fission neutrons are followed, as are neutrons of all generations, until all trajectories have terminated or until the alpha time cutoff is reached. The resulting event history table is analyzed and all fission events are sorted into a set of time bins. The bin counts vs. time distribution reflects the exponential increase (or decrease) of the system, and alpha is extracted by fitting an exponential to the distribution.

For alpha dynamic calculations use...

ALPHA DYNAMIC {NDYPART *np*} {NDYBATCH *nb*} {DYTMAX *tm*} {DYTFIRST *tl*}
{NDYGEN *ng*}

where:

np is the number of particles per batch (defaults to npart);

nb is the number of batches (defaults to nbatch);

tm is the neutron history cutoff time in seconds (defaults to code estimate);

tl is the first time to use in exponential fit (defaults to 0.);

ng is neutron history cutoff time in generations (defaults to 6).

12.2.3 Alpha New

Fits the neutron population vs. time to determine a.

For alpha new (neutron growth) calculations use...

ALPHA NEW {NITER *ni*}

where:

ni is the number of iterations (defaults to 5).

12.3 Criticality ENDL Limitation

For moderated critical experiments containing U238 in dilute solution, errors in the ENDL capture cross section for U238 in the thermal region may cause a low value of k_{eff} to be computed. This problem disappears when the ENDFB6R7 library is used. See the MIX Data Block section for information on specifying ENDFB6R7.

12.4 Criticality Restart—the DUMP Data Block

The DUMP Data Block causes the code to make restartable dumps of your CRITICALITY problem during the run. The problem can later be restarted from any of your DUMP files. This can save you considerable computer time if the machine crashes and takes hours/days of calculations with it. The DUMP file can also be used to “jump start” a new criticality calculation which has a similar geometry. The DUMP input line is a separate data block and should *not* be entered inside any other data block. The form of the input line is:

DUMP n1 n2 n3 ...

Restart DUMP files will be made following the completion of batches **n1**, **n2**, **n3**, etc. The DUMP file names are of the form**rdnnnn**, where *nnnn* is the batch number. For example, if a criticality input file called *crit1* contained the data block, DUMP 20 50 100, the following files would be created as the code finished batch 20, 50 and 100: *crit1.rd0020* *crit1.rd0050* and *crit1.rd0100*. These CRITICALITY dump files, along with the original input file, contain all the information necessary to restart the job at the next batch.

If the problem is interrupted, for any reason, it can be restarted from any of the dump files. If you insert the following line in your input file

RESTART n2

then COG will restart the problem with batch number **n2 + 1**, using information read from a dump file previously made at the end of batch **n2**. The RESTART input line is a separate data block and should *not* be entered inside any other data block.

Note: Because criticality jobs are inherently batch oriented, this DUMP/RESTART process works equally well in both serial and parallel modes.

Note: RESTART is a separate data block and should *not* be entered inside any other data block.

Example: A Criticality file with a DUMP specification to produce restartable dumps after batches 20, 50, 100, 150. Note that the DUMP line is a separate data block.

```
CRITICALITY
NPART=3000
NBATCH=200
SDT=.005
NFIRST=5
NORM=1.
NSOURCE 3
    0. 0. 0.
    1. 1.1 1.2
    1. 4. 4.
DUMP
20 50 100 150
```

Example: A Criticality file with a RESTART specification. This will restart the previous calculation at the end of batch 50 (starting with new batch 51). Note that the RESTART line is a separate data block.

```
CRITICALITY
NPART=3000
NBATCH=200
SDT=.005
NFIRST=5
NORM=1.
NSOURCE 3
    0. 0. 0.
    1. 1.1 1.2
    1. 4. 4.
RESTART 50
```


Another use of the DUMP file is to “jump start” a new criticality calculation which has a similar geometry. Because the DUMP file only contains neutron source locations for the next batch (and *not* any details of the geometry), it can be used to specify the starting source for a new (but similar) criticality problem. To do this, rename the ...**rdnnnn** file to correspond to the new input file, set the NFIRST value to $nnnn + nsettle$ (to skip the first $nsettle$ "settling" batches in the new problem, before values of k_{eff} are saved), and put a RESTART $nnnn$ line into your input file. This will start your new problem with a near-equilibrium distribution of source points from the previous problem. If these problems are similar, you are effectively starting the calculation much closer to an equilibrium source distribution and hence should converge to a final answer much faster.

13 The ACTIVATION Data Block

An activation problem computes the photon flux rate at a detector due to the radiative decay of neutron-activated isotopes in the problem. An activation problem has a neutron source, specifies a standard COG neutron library and additionally specifies a neutron activation library. The input file must include a set of tally times at which the decay photon flux rate is to be computed at the detector.

The activation library has data on activated isotopes and one- and two-step decay schemes. At every neutron collision, COG checks the library for activated species and samples their decay data. COG computes the photon flux rate arising from the decay at the detectors, at each tally time. Prompt secondary photons are also emitted as in a normal neutron transport problem.

The ACTIVATION Data Block has the form:

```
ACTIVATION t1 t2 t3 ... tn
```

where the t_i are the times at which the photon flux rate is to be scored.

A detector HALF-LIFE MASK can be applied to the detector to limit the scoring to photons emitted from radio-isotopes that have a specified half-life. See the **MASK- HL** writeup in the DETECTOR Data Block section.

Example: An activation problem with Co59

```
TEST OF C)59 ACTIVATION
BASIC
  NEUTRON PHOTON $ NEED TO SPECIFY BOTH

SURFACES
  100 SPHERE 5. $ TARGET SPHERE
  101 SPHERE 10.
  102 SPHERE 12.

GEOMETRY
  SECTOR 100 CO59 -100 $ TARGET SPHERE
  SECTOR 101 VAC +100 -101
  SECTOR 102 VAC +101 -102
  BOUNDARY VACUUM +102

MIX
  NLIB = ENDL90
  MAT 1 CO59 8.
```

ASSIGN-ML

1 100 /
0 101 102

ACTIVATION

1. 10. 100. 1000. 10000. \$ SET OF TALLY TIMES

DETECTOR

NUMBER = BC_DECAY
BOUNDARY 101 102 1.
DRF-E PHOTON NUMBER-FLUX

SOURCE

NPART = 1000
DEFINE P = 1
POINT 0. 0. 0. ! NEUTRON SOURCE AT ORIGIN
DEFINE E = 1
NEUTRON LINE 1. 14. ! 14 MEV SOURCE
INCREMENT 1. P=1 E=1

END

The resulting .out file displays the detector results at each of the tally times, and indicates which isotopes contributed to the scores.

RESULTS FOR DETECTOR bcdecay
Neutron activation Co59 problem tactCo59
TYPE -- boundary

This is an activation problem - photon results for total and differential response
are given at a time of 1.000E+00 sec
and represent 'activities', i.e. they include a 1/sec term arising from
the decay constant(s).
TOTAL RESPONSE PER SOURCE PARTICLE
(Integrated over all energies, times and angles and with all masks and response functions included. Units of response are those of the response function (if any) times particles/cm^2. fsd is the fractional standard deviation = std. dev./response.)

The ACTIVATION Data Block

10/30/2024

particle	response	std. dev.	fsd	fom
photon	6.0659E-11	1.6692E-11	2.7518E-01	7.1153E+02

time dependent results for activation problem

time, sec	response/sec
1.0000E+00	6.0659E-11
1.0000E+01	6.0060E-11
1.0000E+02	5.4391E-11
1.0000E+03	2.0203E-11
1.0000E+04	8.6557E-14

activation results as a function of half-life (and assumed isotope and

reaction - check ACTIVATION LISTING BY HALFLIFE to see what other isotopes and/or reactions may contribute to specific results)

time = 1.0000E+00 sec

isotope	reaction	half-life	half-life	response/sec	fsd	% of total
Co59	n,2ng	1.05E+01 min		6.062E-11	0.275	9.99E+01
Co59	n,g	5.27E+00 yr		3.656E-14	0.021	6.03E-02
Co59	n,g	1.05E+01 min	5.27E+00 yr	5.407E-17	0.019	8.91E-05

time = 1.0000E+01 sec

isotope	reaction	half-life	half-life	response/sec	fsd	% of total
Co59	n,2ng	1.05E+01 min		6.002E-11	0.275	9.99E+01
Co59	n,g	5.27E+00 yr		3.656E-14	0.021	6.09E-02
Co59	n,g	1.05E+01 min	5.27E+00 yr	5.380E-16	0.019	8.96E-04

time = 1.0000E+02 sec

isotope	reaction	half-life	half-life	response/sec	fsd	% of total
Co59	n,2ng	1.05E+01 min		5.435E-11	0.275	9.99E+01
Co59	n,g	5.27E+00 yr		3.656E-14	0.021	6.72E-02
Co59	n,g	1.05E+01 min	5.27E+00 yr	5.122E-15	0.019	9.42E-03

time = 1.0000E+03 sec

isotope	reaction	half-life	half-life	response/sec	fsd	% of total
Co59	n,2ng	1.05E+01 min		2.013E-11	0.275	9.97E+01
Co59	n,g	5.27E+00 yr		3.656E-14	0.021	1.81E-01
Co59	n,g	1.05E+01 min	5.27E+00 yr	3.276E-14	0.019	1.62E-01

time = 1.0000E+04 sec

isotope	reaction	half-life	half-life	response/sec	fsd	% of total
Co59	n,g	1.05E+01 min	5.27E+00 yr	4.902E-14	0.019	5.66E+01
Co59	n,g	5.27E+00 yr		3.655E-14	0.021	4.22E+01
Co59	n,2ng	1.05E+01 min		9.799E-16	0.275	1.13E+00

14 Parallel Processing

COG will execute concurrently on a set of networked workstations or on a MPP (Massively Parallel Processor), using the MPI² (Message Passing Interface) library. ***COG works only with MPICH software.*** Concurrent processing is the fastest way to solve long COG problems, and also one of the best ways to take advantage of High Performance Computing (HPC) capacity which frequently goes unused on nights and weekends.

Briefly described, COG runs on a set of computational nodes using a master-slave relationship. The instance of COG which the user starts up becomes the master process for the COG job. If HPC is used, the master COG starts up slave COG processes on a specified number of CPUs (cores) on an MPP. If MPI is used, the user must specify to the parallel- computations control system outside of COG, the set of computer cores on which the COG problem is to be run.

The parallel processing control structure used is master/slave with domain replication. Each slave runs its particles using the complete time, energy, and spatial domains of the problem. This obviates the need for slaves to pass particle information to each other. Communication only occurs between the master and the slaves, and typically takes only a tiny fraction of the total run time. This allows the performance of COG to scale nearly linearly with the number of slave cores used in a calculation.

For shielding problems, each slave is given a starting random number seed, a number of particles to run, and a reporting time interval (in wall-clock minutes). Each slave runs until it has transported the specified number of particles or until the reporting time interval has elapsed. The slave then reports its results to the master, which accumulates them. Each reporting slave is immediately restarted with another number of particles to run. This sequence continues until the number of reported particles processed meets or exceeds the number of particles specified by the user for the entire run. Then all slaves are stopped, and the master completes the processing of the accumulated results. Because each particle trajectory is independent of all others, it does not matter which trajectories are calculated on which machine. We get as good an answer from a parallel calculation as from a serial one.

Criticality problems have a different organization than shielding problems. In the usual serial mode, a batch of fixed size is run and the number of fissions caused by the batch is tallied. The desired quantity k_{eff} is defined as the ratio of fission-source neutrons of successive generations. Because the fissions created by one batch are used as the neutron source for the next batch, the source evolves from the start of the calculation until some equilibrium source distribution is achieved (this is termed the

"settled" source). After the specified number of "settling" batches are run, COG averages the batch k_{eff} over all subsequent batches.

In parallel mode, each slave node is given a starting random number seed and requested to calculate an indefinite sequence of batches. Once the criticality source distribution on a node has settled into an equilibrium state, one batch result for k_{eff} is as good as any other. Each slave works independently according to its speed and workload and reports batch results for k_{eff} to the master as they are computed. The master collects the individual batch results and averages them across all nodes. When convergence tests are satisfied, the master stops the slaves and compiles the final results.

These methods are efficient, in that they produce an answer in the shortest possible wall-clock time, and robust, in that they are resilient to processor failures. All slaves are kept busy (never waiting to receive instructions from the master). If one or more processors fails or bogs down due to increased loads from other processes, the computational load automatically shifts to the remaining slaves. Satisfactory job completion is assured as long as just one slave and the master node remain alive.

14.1 Setting Up A Parallel Processing Run

To run COG in parallel processing mode, the MPICH libraries must be installed on the workstation with multiple cores, or HPC/MPPs you wish to run on.

14.2 The PARALLEL Data Block (optional)

The configuration desired for a COG parallel processing run is specified by the PARALLEL Data Block.

It has the form:

```
PARALLEL {NSLAVES ns} {TSLAVES ts} {NPARTSLAVE npts}
        {SYNC} {RNSLAVES rn1a rn1b....} {SEQNO seqn1 seqn2 ...}
        {HOSTFILE path-to-hostfile}
```

where:

ns is the number of COG slaves to be run,—default =1, maximum number determined by availability.

ts is the slave reporting interval (in minutes)—default = 10.

npts is the number of particles which each slave will be requested to run—default is the total number of particles left to run in the entire job. For the usual shielding problem, this parameter should be omitted.

SYNC means to use the Synchronous control mode, which is primarily useful for debugging. The normal user mode is the robust and efficient Asynchronous (the default mode).

rn1a rn1b ... are starting random number seeds (one pair of seeds is required for each COG slave). Each seed is an integer in the range (1,30000), approximately (See section: **BASIC Data Block : Starting Random Numbers – RN**). The default is that the master COG chooses starting seeds.

seqn1 seqn2 ... may optionally be provided, one per slave. This starting sequence number is the ordinal number in the sequence determined by the starting seeds, of the first random number to be used by the slave. Default is 1.

path-to-hostfile gives the complete pathname to the MPICH hostfile for this run. This is only needed for runs under MPICH. COG will read the hostfile for COG inputs embedded in COG Comment lines.

Most users will only need to set the number of COG slaves *ns* and the slave reporting time interval *ts*. (*ts* is ignored for Criticality jobs).

Under MPICH, COG will assign slaves to the MPICH hosts in round-robin fashion, beginning with the host on which you start up COG, then progressing down the list of hosts given in the MPICH hostfile, until all *ns* COG slaves have been started up. For each host, COG will start up *ncpu* slaves (as specified in the COG Comment lines in the MPICH hostfile) or one slave (by default). If COG has cycled through all hosts in the

list, and more slaves remain to be started, COG begins another slave assignment cycle. Under MPI, the assignment of slaves to nodes occurs outside of COG, in the parallel-processing control system used on the machine. The number of computational nodes requested must equal $ns + 1$.

14.2.1 Control Modes for Shielding Calculations

In the normal Asynchronous control mode, each COG slave runs until the requested number of particle trajectories *npts* is run or until *ts* wall-clock minutes have elapsed. Each slave returns its results to the master, and is immediately restarted with another batch. The problem terminates when the master has accumulated results for the number of particles specified for the entire run.

In this Asynchronous mode, *ts* is the controlling factor for efficiency. If *ts* is set too large, then the calculation tends toward single-processor performance. In the extreme, if *ts* is so large that the fastest processor completes the entire calculation in time *ts*, then the efforts of all the other slaves are wasted. At the other extreme, if *ts* is very small, then the overhead associated with passing results to the master and accumulating them could become large. It is expected that values of *ts* in the range of 5 to 20 minutes should prove satisfactory in most situations. The amount of wall-clock time spent in "wasted" calculations (i.e., those which are discarded at the end of a run) is expected to be of order $ts(ns-1)/2$. In the Asynchronous mode you need not specify *npts*; it will default to the number of particles remaining in the problem.

By setting the **SYNC** flag, the user can run in the Synchronous control mode. In this mode, each slave is started to run *npts* particles (which must be specified), for an unlimited time. Every slave must report back its results to the master before any new assignments of *npts* particles are given out. This synchronizes the slaves to run in well-defined cycles. By the end of the run, each slave will have run exactly the same number of particles. By specifying the starting random number seeds, this parallel run can be exactly reproduced. This is very useful for debugging, although inefficient for processing. By contrast, the Asynchronous mode is efficient but essentially irreproducible, because the order in which new assignments are given out, and the total number of particles transported by any one slave, depend on the (changing) machine load and network traffic.

14.2.2 Control Modes for Criticality Calculations

In the normal Asynchronous mode, each slave independently runs batches of size $npts = npart$ (where $npart$ is set in the CRITICALITY Data Block). ts is ignored. Each slave returns its results to the master, and is immediately restarted to run another batch. The results from all slaves which have run the requisite number ($nfirst$) of "settling" batches are used to determine the average k_{eff} . The problem terminates when the master determines that convergence has occurred. The "unsettled" results from the first reporting node are saved so that the progress towards settling can be later reported.

By setting the SYNC flag, the user can run in the Synchronous mode. In this mode, each slave independently runs batches of size $npts = npart/ns$ (where $npart$ is set in the CRITICALITY Data Block). ts is ignored. Every slave must report back its results to the master before any new assignments are given out. This synchronizes the slaves to run in well-defined cycles. By specifying the starting random number seeds, this parallel run can be exactly reproduced. This is very useful for debugging, although inefficient for processing.

14.3 Running COG in PARALLEL Mode under MPI

MPI requires a controlling routine to start up parallel processes. Most systems have the **mpirun** routine available for this purpose.

Start COG in parallel-processing mode by typing:

```
mpirun -n nproc path-to-COG -master -ns nproc-1 inputfile
```

where:

nproc is the number of processors (*up to Maximum nproc = 65535*) to use in this calculation;

path-to-COG is the path to the COG executable,
e.g., /opt/apps/cog/COG11/COG11.3;

-master -ns nproc-1 signals COG to organize a concurrent calculation, with itself as the process master. *If this is omitted, then COG will run the job in ordinary serial mode, regardless of any PARALLEL Data Block in the input file.*

inputfile is the name of the COG input file.

Output files from the Master contain the final results. These files have the name of **inputfile** with an appropriate suffix appended, such as **in1.out**, **in1.ps**, etc. They reside in the same directory as **inputfile**. Slave COG processes on other nodes also produce output files which are written to local directories. These are named like the Master output files, but with an **Snn**, such as **in1S03.out**. These files are only of interest if the slave process fails to finish normally, and in fact these are purged by COG if the slave terminates normally.

For longer-running COG problems, **mpirun** can be run as a background job, or there may exist a batch-processing system to which you can submit a parallel COG job. Regardless of the system which launches COG, the COG execute line must include:

```
mpirun -n nproc path-to-COG -master -ns nproc-1 inputfile
```

14.3.1 When Trouble Occurs...

- Check the log files created by MPI. One will contain the console listing which would normally go to your screen when COG runs. This file may contain error messages from the slaves on this machine.
- Check the slave output files for error messages.

14.4 Running COG in PARALLEL Mode under HPC (LLNL LC Only)

In order to run COG under LLNL LC, a Slurm Batch Script must be created first to specify all the necessary information of bank on a machine. An example of a Slurm Batch Script in either short or heavily annotated is listing below.

Short Slurm Batch Script My_sbatch_script.cmd:

```
#!/usr/bin/tcsh
#####
#SBATCH --job-name=tcf19
#SBATCH --qos=normal
#SBATCH --account=wbronze
#SBATCH --time=00:05:00
#SBATCH --partition=pbatch
#SBATCH --nodes=80
#SBATCH -o /p/lscratchh/<USERNAME>/tcf19.log
#SBATCH -e /p/lscratchh/<USERNAME>/tcf19.eelog

#####
##
set echo
cd /p/lscratchh/<USERNAME>
setenv PGPLOT_FONT /usr/gapps/cogdev/grfont11.3.dat
srun -n1440 -k /usr/gapps/cogdev/COG11.3 -master -ns 1439 tcf19
echo 'Done'
```

...OR ...

Heavily annotated Slurm Batch Script My_sbatch_script.cmd:

```
#!/usr/bin/tcsh
## The top line is for bash users. If you use a different shell,
## change to that shell.
##
## Take this code block and copy it into a file called sbatchScript
## then run the script by typing "sbatch sbatchScript"
##
## I urge you to cd to the directory where you want to
## do things, then copy this script there, then run.
## Trust me, it is easier this way.
##
```

```

## Remember, two pound signs means a comment, while
##      one pound sign means a command to SLURM.
##      after all the SLURM setup, put the commands you need.
##
## This script as written does the following;
##  run with a quality of service (qos) set to normal
##  name the job tcf19
##  use bank wbronze
##  set the max runtime to 5 minutes
##  use the pbatch partition, not pdebug
##  run on 80 nodes (QUARTZ node/18 cores: 80 nodes x 18 cores = 1440 CPUs)
##  run a couple of COG commands
##  complete
## If you run this and then suddenly realize the job
## is going to go horribly wrong, type the first line
## to get the list of jobs you have in the queue.
## Then type the second line to kill the job before it
## throws out gigabytes of trash to your home directory.
##  showq -v -u `whoami`
##  scancel <jobid>

#####
## Give your job a name - let's call this one tcf19
## Don't put spaces in the name.
#SBATCH --job-name=tcf19

#####
## Pick your quality of service
## normal - means that it can't be pre-empted and you have
##      to wait to get to the top of the queue
## standby - you will be pre-empted if someone using qos
##      normal wants the nodes (used in special cases)
## expedite - you get to the top of the queue - only use
##      this if you are approved
#SBATCH --qos=normal

#####
## Set your bank
## If you don't set your bank, you will be using your
## default bank -which may not exist or it may be set to
## only allow you qos standby
##
## To find out which banks are available to you on a
## machine type "sacctmgr list user `whoami` witha" at the prompt
#SBATCH --account=wbronze

```

```
#####
## Set your time limit in hours:minutes:seconds
## to find out how long a batch job can run, type
## news job.lim.`uname -n | sed -e 's/[0-9]*$/'`
## at the prompt
#SBATCH --time=00:05:00

#####
## Choose to run in the pbatch or pdebug pool. The pdebug
## pool is only for debugging, not production work!
## To find out what pools are available on the machine,
## type 'sinfo' at the prompt.
#SBATCH --partition=pbatch

#####
## Set the number of nodes you want to use :
## on most of our machines, you automatically get all the
## processors on those nodes for your exclusive use.
## if you want to see the number of processors on a given
## node of a machine, type "news job.lim.`uname -n | sed -e 's/[0-9]*$/'`"
#SBATCH --nodes=80

#####
## To direct slurm run errors and output files to directory
## location
#SBATCH -o /p/lscratchh/<USERNAME>/tcf19.log
#SBATCH -e /p/lscratchh/<USERNAME>/tcf19.eelog

#####
## Run the test
## To get an idea of when the job is going to start
## showstart <jobID>
## To get an idea of where the job is in the queue
## showq -v -i | more
## the next lines are where you put your actual commands

set echo
cd /p/lscratchh/<USERNAME>
setenv PGPLOT_FONT /usr/gapps/cogdev/grfont11.3.dat
srun -n1440 -k /usr/gapps/cogdev/COG11.3 -master -ns 1439 tcf19
echo 'Done'
```

Start COG in parallel processing mode on LLNL LC by typing:

sbatch My_sbatach_script.cmd

14.4.1 When Trouble Occurs...

- Check the log files created by Slurm log and elog. One will contain the console listing which would normally go to your screen when COG runs. This file may contain error messages from the slaves on this machine.

- Check the slave output files for error messages.

Your job may have been running out of time slots and Slurm have cancelled your job “DUE TO TIME LIMIT”.

14.5 Very Long COG Run with Restart

COG is capable of running a very long job with maximum 65535 CPUs, and maximum particles of 10000000000000 with COG11.3 or later. It's always a good idea to set up your input deck to accommodate any job interruption due to cancellation of your job's “DUE TO TIME LIMIT” or job crashed.

14.5.1 Criticality Run with Restart

Example: A typical initial setup of CRITICALITY Data Block with creating restart files at 1000000, 2000000, 3000000, and 4000000 nbatch.

```
CRITICALITY
NPART=1000
NBATCH=5000000
SDT=.005
NFIRST=8
NORM=1.
NSOURCE
3 0. 0. 0.
  1. 1.1 1.2
  1. 4. 4.

DUMP 1000000 2000000 3000000 4000000
```

...output files listing

```
input
input.ps
input.det
input.rd01000000
input.rd02000000
```


...And subsequent to setup of CRITICALITY Data Block with restart continue run beginning at 2000000 nbatch.

```
CRITICALITY
NPART=1000
NBATCH=5000000
SDT=.005
NFIRST=8
NORM=1.
NSOURCE
3 0. 0. 0.
  1. 1.1 1.2
  1. 4. 4.

DUMP 3000000 4000000
RESTART 2000000
```

14.5.2 Source Run with Restart

Example: A typical initial setup of Basic Data Block for Source calculation problem with creating restart files at approximately 1000000000 particles.

```
Basic
neutron deuteron
rn 17328 23435
deuttransport
matdt 6
ndump 1000000000
```

```
Source
npart=3.30E+11
increment 1 p=1 e=1 a=1 t=1
define position = 1
cylinder -3 0 0 0 0 0 0 6
define energy = 1 deuteron
  line 0. 4 1
define angle = 1
  1 0 0 fixed
define time = 1
Gaussian 0 2.0e-9
```

...output files listing

```
input
input.ps
input.det
input.rsn318028080413
input.rsn319000022355
```

...And subsequent to setup of Basic Data Block with restart continue run beginning at 319000022355 particles.

```
Basic
neutron deuteron
rn 17328 23435
deuttransport
matdt 6
ndump 1000000000
restart n 319000022355
```

14.6 References

- 1.0 MPICH is distributed as source (with an open-source, freely available license). The MPICH software is freely available at url address: <https://www.mpich.org>
- 2.0 W. Gropp, E. Lusk, and A. Skjellum, Using MPI, 2nd Edition, MIT Press, (November, 1999).

15 Understanding Your COG Run

Each COG run produces several output files, which reside in the same directory as the COG input file. These files give you the detector results (which are the principal "answers" of a COG run) and also provide much additional information to help you evaluate the validity of your answers. This section describes the contents of these files, and presents some statistical considerations to help you evaluate uncertainties in the results and detect under-sampling problems.

15.1 COG OUTPUT Files

COG output files are named by adding suffixes to a root name, which is the name of the user-supplied input file. Where output file names are discussed in this manual, the root name is indicated by ...

15.1.1 Graphics Output – The ps File

The following graphics output is included in the ps file.

Geometry pictures (optional). w detailed cross sectional and/or perspective pictures of the problem geometry. The By including appropriate PICTURE commands in the GEOMETRY Data Block, the user can drase PICTURES serve two purposes. First, they help the user verify that the geometry is correct in concept and execution — are the collimators properly situated? — is the lead shield in place? Second, it is at this point that COG does error checking on the GEOMETRY, since error checking during the random walk would be too time consuming. While generating these pictures, COG checks that all points in the view fall in one and only one defined SECTOR. Particles which are transported through an improperly-defined geometry — one in which two or more sectors (elementary volumes) overlap — may get "lost" and do unexpected things. Unless the final problem results are obviously wrong, the user may not be aware there is a problem and the code will calculate an inaccurate result — if it doesn't blow up during execution. Only the portions of the GEOMETRY contained inside each PICTURE is checked for overlapping SECTOR definitions, consistent SECTOR specifications, etc. The user should check **all** the GEOMETRY by drawing a **complete** set of PICTURES. See the section **PICTURES of the Geometry**.

Detector response functions. For each detector, all response functions specified in the form of a distribution are plotted. This includes all time response functions, all polar angle response functions, and energy response functions in the form of energy- response pairs. (Energy response functions given as dose, energy-deposition, etc. are not plotted). These plots help the user catch data-entry errors. See section **Detector Response Functions (DRFs)**.

Source description (for non-criticality problems). Each SOURCE Data Block DEFINE statement (which specifies a POSITION, ENERGY, ANGLE, or TIME dependence of the source) is plotted. See section **SOURCE Data Block. Material cross sections (optional)**. You can set flags in the I/O Data Block to:

- generate plots of the **material** total macroscopic cross section of all materials specified in the MIX Data Block, for each particle type specified. Cross-section units are in inverse cm.
- generate plots of the **isotope/element** total cross section of all materials specified in the MIX Data Block, for each particle type specified. Cross section units are in barns/atom. See section **The I/O Data Block**.

ANALYSIS (optional). By specifying an ANALYSIS Data Block, the user can have COG plot the densities of particle collisions, in specified portions of the problem phase space. These pictures show collision sites within a physical volume for a specified particle type, energy range, and weight window. These analysis pictures are very helpful in learning where reactions are occurring in your problem. See section **ANALYSIS Data Block**.

Criticality results (for criticality problems).

For a Criticality problem, COG will produce these plots:

- Individual batch k_{eff} vs. batch number;
- Average k_{eff} (calculated using last n batches) vs. first batch used in average;
- Average k_{eff} (averaged over 5 batches) vs. batch number;
- Average k_{eff} (averaged over 10 batches) vs. batch number;
- Average k_{eff} (averaged over 15 batches) vs. batch number;
- Distribution of k_{eff} , with a test for normality;
- Fraction of fission neutrons born with energy $< E$, vs. E .

Detector results. Each 1-D set of **differential results** (BIN structure) specified for a DETECTOR will generate three plots in the ...ps graphics file. The first plot is the detector response as a function of the differential (BINned) quantity. The second plot shows the same data, with the addition of dotted lines to indicate both plus and minus one standard deviation. The third plot is the running integral response. Each set of 2-D **differential results** (BIN structure) specified for a DETECTOR will generate a density plot representation of the results vs. the two variables. See section **Plotting Detector Differential Results**.

15.2 Graphics Output

Graphics output files are in postscript format and are produced via the PGPLOT graphics library, developed at the California Institute of Technology. These files have the name of **inputfile** with the suffix **ps** appended, such as **input.ps**. You can view the **input.ps** file by running the freely distributed programs **gs (ghostscript)**, or **gv (ghostview)**, or any other postscript viewer.

15.2.1 Text Output – The out File

The COG out file is the main output file which gives COG's interpretation of the input data, the DETECTOR results, and several standard tables. This output file is an ASCII text file. You should examine it to see if the input is what you anticipated and if the results seem "reasonable." Examples of the out file are given after some of the example problems following this section.

SOURCE Data Block: Average source parameters. COG tabulates for the source, the minimum, maximum, and average values of the various source parameters. These are listed in this section. This table is intended as a check, in broad terms, of the source description,

Regional Summary Result tables. COG produces a detailed synopsis of all the random walk histories in the form of summary tables for all problem regions. An example is given below. The tables consist of, for each REGION, seven columns headed by Event, Part, # of Events, Removal Weight, Addition Weight, Deposited Energy (MeV), and Energy Balance (MeV). "Event" lists types of events, and may include entries such as fixed source, current-in, current-out, leakage, splitting, elastic, (n,g), Compton. "Part" indicates the particle types involved in the event. "# of Events" records the number of such events. "Removal/Addition Weight" denotes the total statistical weight removed from/added to the REGION by the event. "Deposited Energy (MeV)" tallies the energy deposited in the REGION by the event. "Energy Balance (MeV)" records the change in particle kinetic energy in the region, due to this event.

Scaling of Regional Summary Results: The values in the REMOVAL, ADDITION, DEPOSITED, and BALANCE columns are computed per source particle, then multiplied by the source strength. As an example, we reproduce a part of the summary tables for a COG neutron-source problem (source strength = 1) and discuss the table entries.

REGION NUMBER		10				
		# OF	WEIGHT		ENERGY (MEV)	
EVENT	PART	EVENTS	REMOVAL	ADDITION	DEPOSITED	BALANCE
-----	----	----	-----	-----	-----	-----
CURRENT IN	NEUT	1000	.0000E+00	1.0000E+00	.0000E+00	3.0000E+00
LEAKAGE	NEUT	978	9.7800E-01	.0000E+00	.0000E+00	-8.5487E-01
	PHO	186	1.9696E-01	.0000E+00	.0000E+00	-1.5616E-01
ELASTIC	NEUT	28317	2.8317E+01	2.8317E+01	9.1444E-01	-9.1444E-01
(N,N'G)	NEUT	878	8.7800E-01	8.7800E-01	1.4798E-01	-1.2281E+00
	PHO	878	.0000E+00	9.2059E-01	.0000E+00	1.0801E+00
(N,G)	NEUT	22	2.2000E-02	.0000E+00	8.8243E-03	-2.6279E-03
	PHO	22	.0000E+00	4.2176E-02	.0000E+00	1.6386E-01
RAYLEIGH	PHO	370	4.0781E-01	4.0781E-01	.0000E+00	.0000E+00
COMPTON	PHO	6331	6.8776E+00	6.8776E+00	9.8058E-01	-9.8058E-01
PHOTOELECTRIC	PHO	740	8.0652E-01	2.9176E-02	6.2675E-02	-6.2675E-02
PAIR PROD	PHO	7	1.1547E-02	2.3094E-02	4.4521E-02	-4.4521E-02
-----	----	----	-----	-----	-----	-----
TOTALS	NEUT		3.0195E+01	3.0195E+01	1.0712E+00	-1.5673E-07
	PHO		8.3004E+00	8.3004E+00	1.0878E+00	4.1936E-10
	ALL				2.1590E+00	
			STATISTICS ON PRE-COLLISION WEIGHTS			
			MINIMUM	MAXIMUM	AVERAGE	
			NEUT	1.0000E+00	1.0000E+00	
			PHO	1.0000E+00	3.8122E+00	
					1.0880E+00	

CURRENT IN: 1000 neutrons ADD a weight of 1.000 to the region. The BALANCE term indicates the change in total neutron particle energy in the region due to CURRENT IN. This term is the product of particle weight and particle kinetic energy, summed over all CURRENT IN events.

LEAKAGE: 186 photons REMOVE a net photon weight of 1.9696E-01 by leaving (leaking from) the problem geometry.

ELASTIC: 28317 neutrons undergo elastic collisions, which leave the net weight of neutrons in the region unchanged. However, nuclear recoils deposits energy (9.1444E-01 MeV) and cause an equal decrease in the neutron energy in the region, as indicated by the BALANCE term.

(N,G): 22 neutrons enter into (n,gamma) collisions and are absorbed (REMOVAL weight of 2.2000E-02). These reactions deposit energy and decrease the neutron energy in the region (BALANCE term is negative). These reactions also produce gammas (ADDITION of 4.2176E-02) and increase the photon energy in the region (BALANCE term is positive).

PHOTOELECTRIC: 740 photons undergo photoelectric interactions, resulting in a REMOVAL weight of 8.0652E-01. Radiative relaxation of the target atoms produces an ADDITION weight of 2.9176E-02 photons. Some energy is deposited locally, decreasing the energy in the photon field.

The TOTALS show that the ADDITIONs equal the REMOVALs for both neutrons and photons, and the BALANCE values are approximately zero, as they should be, showing that the changes in particle energies for all events have been properly accounted for. The total energy deposition is 2.1590E+00 MeV, approximately evenly divided between neutron and photon deposition events.

The "Statistics on Pre-Collision Weights" show the minimum, maximum, and average weights for each particle type, just before a collision occurs. This gives the user some idea of the weights of particles interacting in this region.

Note that the "# of Events" counts the number of particles that undergo the specified event; it does not measure the particle statistical weight, so the value is not a real answer (e.g., a flux estimate). However, if this number is small, the results will have a large uncertainty. If it is large, the results will be more certain.

Each table row lists a particular event or reaction that occurred within the region. There are about 100 different such events, so this example, or any real problem, will show only a few of them. The labels will properly identify each event. Listing each reaction type greatly lengthens the summary tables, but it serves to inform the user of the underlying transport physics.

There are other facts you should keep in mind when examining the summary tables:

- Any REGION which was specified in a problem and had **no** events occurring in it, will be identified by printing the **reg-ID#** and the titles, but it will have no additional data.
- A REGION with **reg-ID#** = 0 exists in all problems, even if the user did not define one. It is a REGION which encompasses **all undefined volumes in the problem**. Generally, it represents a void outside of the defined geometry, extending to infinity.
- A *low-energy* event signifies that a particle has reached an energy less than the tabulated cross-section data.
- The "# of Events" means the number of particle **entering** into a reaction or an event, not the number of particles exiting from such events.
- If secondary particles are included in a problem, the sum of all deposited and leaked energy may be more than the total source energy. This can occur for a couple of reasons. Reactions with positive Q-values will create particle energy in the region. And limitations in the COG database for some reactions, particularly (n,Xg), permit COG to conserve energy for those reactions only on the average.

Detector results: Mean and Standard Deviation. The result obtained by a Monte Carlo calculation of a problem is a *mean value*, obtained by averaging N individual particle contributions. If we reran the calculation many times, using the same value of N but different sets of random numbers, we would get a set of mean results, which would be distributed about some average value, with some dispersion or standard deviation. The size of the standard deviation tells us how closely any one mean result comes to the "true" answer – one we would get if we reran the problem an infinite number of times and averaged all results.

Many Monte Carlo codes operate in what is known as a batch mode, e.g. 10000 histories will be run as, say, 50 batches of 200. The contributions within each batch are averaged to yield a batch result; the statistics, the mean and the standard deviation of the mean, are then calculated based on these batch answers, not on the original individual contributions. COG, on the other hand, uses an effective batch size of 1, i.e., all contributions to a result arising from a single source particle and all its progenies are summed to form a "batch" answer. The mean and the standard deviation of the mean are calculated from each contribution made by each source particle. See the following section **COG Statistical Considerations** for more information on how to interpret the statistical significance of COG results.

Detector results: Ten largest contributors. For each detector COG lists the source particles which made the ten largest contributions to the total detector response — by particle sequence number and by percent contribution. The particle sequence numbers are given to enable a RETRACE to determine the phase space parameters corresponding to these "important" contributors. If the top ten particle percent contributions are small and about the same, then one of two situations prevails. Either there are many significant contributions and the results are valid, or the problem is under sampled, there are no significant contributors, and the results are not valid. There is no sure way of distinguishing between these two alternatives. On the other hand, if only a few particles contribute almost all the answer, then it is obvious that the problem is under sampled (see section **The Problem of Under Sampling**) and steps can be taken to rectify the situation. The RETRACE option can be used to determine why certain particles are important. This information can then be used to bias the problem so as to produce more of these high scores and, hopefully, a more believable result.

Detector results: Contribution by scattering generation. The total-response-as-a-function-of-scattering-events-since-source-generation is given to provide added insight into the problem. If almost all the answer comes from 0-scattered (uncollided) particles, the problem is very simple and could probably have been solved more quickly by hand. If a large fraction of the answer comes after several or many scatterings, then the problem is fairly complex and the particles probably travel a convoluted path to reach the detector.

If particles scatter many times and make only small contributions to the result, it may prove beneficial to do biasing to avoid spending time following these unimportant particles. For example, doing Russian roulette, (see **WALK-COLLISION** or **WALK-BC**) with a survival probability, $b = 0.9$, reduces the probability of particles scattering 10 times by about 3, scattering 20 times by about 9, etc.

Detector results: Contribution by region (for POINT DETECTOR). For a POINT detector, COG lists the percentage of the total detector response arising from scattering events in each REGION, thus identifying the important REGIONS in the problem. This information can be used in future runs to bias the problem to force more particles to interact in the important regions, or to LIMIT the POINT detector calculation to reduce the time spent doing unimportant computations.

Detector results: Differential (BINned) results. COG lists differential results for each requested binning structure, in ENERGY, ANGLE, or TIME.

15.2.2 Detector Output The det File

Key DETECTOR results for each DETECTOR are reproduced in the det file in a tab-delimited form acceptable to many common plotting applications. Some application programs may require minimal editing of the file for it to be useable. This same data is contained inside the out file, but in a less convenient form. Because different types of data may be requested by different DETECTORS, column headings and labels are not stated in the det file. The user should read the out file to find the proper units and headings for the various det file columns.

15.2.3 Cross Section Files

By setting appropriate flags in the I/O Data Block, COG will write disk files that contain:

- the total macroscopic cross section in inverse length units, e.g. cm^{-1} , for each material and particle type in the problem;
- the total cross section in barns/atom for each isotope/element and particle type in the problem.

See section **The I/O Data Block**.

15.2.4 LIST of Scoring Particles – The lst File

Every time a particle is scored into one of the ENERGY BINS of your detector, COG checks to see if the LIST option is enabled, and if the scored energy also falls into one of the LIST energy ranges. If so, the scoring particle properties are listed in a file named**list**. Particle properties LISTed include the particle sequence number, the scored energy, and the index of the ENERGY BIN in which the particle was tallied. See section **LISTing Properties of Scoring Particles**.

15.3 COG Statistical Considerations: Elementary Statistics

We give here simple definitions of some basic statistical concepts and quantities, which may help users better understand their COG results.

For a COG problem, let x_i be the contribution to a specified detector from a particular event history i . The collection of *all* possible x_i 's is called the *population*. The answer to the problem is just the average of the x_i 's over the population. This *population mean* is given by

$$\mu = \Sigma x_i / N,$$

where N is the *population size*, i.e., the total number of possible event histories. The *population variance*

$$\sigma^2 = \Sigma (x_i - \mu)^2 / N,$$

measures the average squared displacement of the individual x_i 's from the mean. The *population standard deviation*, σ , is just the square root of the population variance. A small variance indicates the x_i 's are tightly bunched about the mean, while a large variance denotes a large spread (dispersion) in the x_i 's. μ and σ^2 are *parameters* of the population.

A (*random*) *sample* is a (randomly selected) subset of the population, i.e., a collection of n x_i 's (randomly) chosen from the collection of all possible x_i 's, where n is the *sample size*. The *sample mean* and *sample variance* are given by

$$\langle x \rangle = \Sigma x_i / n$$

and

$$s^2 = \Sigma (x_i - \langle x \rangle)^2 / (n - 1)$$

respectively, where again the *sample standard deviation*, s , is just the square root of the sample variance. $\langle x \rangle$ and s^2 are *statistics* of the sample.

As we stated, the answer to the problem is μ . ($\mu = \sum x_i / N_{\infty}$ as N approaches infinity, being defined as *truth*.) But, since in reality N is infinite, it is very difficult indeed to compute each of the required x_i 's. So, in effect, the exact value of m is unobtainable.

We are saved by the fact that the Law of Large Numbers tells us that in the limit as n approaches N , $\langle x \rangle$ approaches m . So if we can take a large enough sample, then $\langle x \rangle$ is a good approximation to m (and, in addition, s is a good approximation to σ).

The quantity $\langle x \rangle$ is a random variable since its value depends on the x_i 's corresponding to the particular event histories making up the sample. That is, each sample of size n would produce a different value for $\langle x \rangle$. The distribution of $\langle x \rangle$'s for all possible samples of a given size is called the sampling *distribution of the mean*. This leads to another important result in the theory of statistics.

The Central Limit Theorem.

For a population with mean m and variance σ^2 , the sampling distribution of the mean for sample size n has a mean $m_m = m$, and a variance $\sigma_m^2 = \sigma^2/n$. The sampling distribution of the mean approaches a normal distribution as n increases, even if the underlying population distribution is far from normal.

This means that as the sample size increases, the distribution gets more and more narrow — the "width" or dispersion of the distribution goes like one over the square root of n , i.e.,

$$\sigma_m \propto 1/\sqrt{n}$$

Hence as n increases, the possible values of $\langle x \rangle$ are grouped tighter and tighter about $\mu_m = \mu$. In addition, the distribution tends toward a normal distribution, so that the standard deviation begins to take on its usual meaning. The following statement, though not quite correct, is close enough for our purposes. For a normal (Gaussian) probability distribution, the probability that m lies in the interval:

$(\langle x \rangle - \sigma_m, \langle x \rangle + \sigma_m)$ is $\approx 66\%$, answer within 1 sigma;

$(\langle x \rangle - 2\sigma_m, \langle x \rangle + 2\sigma_m)$ is $\approx 95\%$, answer within 2 sigma;

$(\langle x \rangle - 3\sigma_m, \langle x \rangle + 3\sigma_m)$ is $\approx 99.7\%$, answer within 3 sigma.

This sets real limits on μ and, therefore, on the possible answers. The standard deviation quoted by COG for each result is the estimate s_m of σ_m , where $s_m = s/\sqrt{n}$.

If n , the number of significant contributions to a COG result, is ≥ 30 , then the Central Limit Theorem tells us that we have sampled from an approximately normal distribution. In this case the $1\sigma_m$, $2\sigma_m$, ... confidence limits given above are reliable. On the other hand, if the number of significant contributors is small, an approximately normal distribution is not guaranteed. For this case a theorem due to Chebyshev states that the probability μ lies in the interval:

$(\bar{x} - 4.5\sigma_m, \bar{x} + 4.5\sigma_m)$ is $\approx 95\%$.

Keeping in mind that there is still a 5% chance for the "true" result to lie outside the given range, these broad limits on m are not very satisfactory. One needs to run longer or bias the problem so as to get a larger number of "significant" contributions to narrow the dispersion of the mean.

15.4 The Problem of Under Sampling

We now deal with a situation that is very common in deep penetration studies — under sampling. As an introduction let us paraphrase a question we have been asked by many COG users. "I made a COG run, got an answer with good statistics, and thought I had a believable result. I repeated the run with minor (or no) changes, got a much different answer, and the statistics were much worse! What happened?" .

The following example may help explain what is going on. We are going to construct a problem simple enough that we can calculate the probability of any detector result which COG might realize for this problem. Then we can compare the COG-computed result statistics with the a-priori probability of the result. This example is, of course, idealized to keep the calculations simple, but the basic results are valid and apply in a general way to under sampled problems.

Consider a problem consisting of a source, geometry, and detector, with the caveat that the probability of a scoring event - i.e., a particle leaving the source, traversing the geometry, and contributing to the detector - is very small. Let us define:

p_1 = probability that a source particle will score = .001;

p_0 = probability that a source particle will not score = .999;

c_1 = detector contribution for a scoring particle = 1;

c_0 = detector contribution for non-scoring particle = 0 to 0.000002.

If the detector in question was of the BOUNDARY or REACTION type, then c_0 would be 0, because particles must actually reach the detector to score. POINT detectors, on the other hand, must calculate, for each scattering event, some contribution, albeit maybe a very small amount, to the detector. We assume we are dealing with a POINT detector, i.e., c_0 can only take on non-zero values. The expected (average) result, per source particle, is:

$$\langle c \rangle = p_1 c_1 + p_0 c_0 \approx 0.001.$$

The actual result obtained by running COG on this problem will be:

$$c = [mc_1 + (n - m)c_0]/n,$$

where m is the number of scoring particles, and n is the sample size, i.e., the total number of particles leaving the source. The expected number of scoring particles is:

$$\langle m \rangle = np_1.$$

Under sampling occurs when $\langle m \rangle$ is of order 1. This is the case when very few particles contribute appreciably to the result.

The law of large numbers tells us that as n approaches ∞ , m approaches $\langle m \rangle$ and, therefore, c approaches $\langle c \rangle$. For finite n the situation is much different. The particle scoring probabilities describe a binary experiment (where the score is either 1 or effectively 0), so we can use the binomial probability distribution function to compute the probability of various outcomes. We can compute the probability of getting 0 scoring particles, 1 scoring particles, 2 scoring particles, etc. for a given sample size. We will compare this to the COG-computed FSD statistic, which estimates the Fractional Standard Deviation of the mean result:

$$\text{FSD} = s_m/r,$$

where

$$r = \text{detector result} = \sum c_i/n;$$

$$s_m = \text{estimate of the standard deviation of the mean result}$$

$$= s/\sqrt{n};$$

where

$$s = \text{estimate of the sample standard deviation}$$

$$= [\sum (c_i - \langle c \rangle)^2 / (n - 1)]^{1/2}$$

Table 1 summarizes the possible outcomes for sample size $n = 1000$. For this sample size, the expected number of scoring particles $\langle m \rangle = 1$, and the expected score $\langle c \rangle = .001$. Columns are m (the number of scoring particles), $P(m)$ (the binomial probability of this outcome), c (the detector score), and FSD (the COG-computed statistic).

TABLE 1			
$n = 1000$ $\langle m \rangle = 1$ $\langle c \rangle = .001$			
# of scoring parts.	Pr (m)	Detector Score	FSD
m		c	
0	0.368	0.001	0.02
1	0.368	0.001	1.00
2	0.184	0.002	0.71
3	0.061	0.003	0.58
4	0.015	0.004	0.50
5	0.003	0.005	0.45

If we run our COG problem many, many times with different starting seeds, we would get a frequency distribution of results that approximates the Table probabilities. We would find that in $\approx 1/3$ of the runs we get no scoring events ($m = 0$). The result c for this case is smaller than the expected result by a factor of 1000, but it exhibits a very good FSD statistic. If we didn't know better, we might well believe this result. In another $\approx 1/3$ of the runs, we would get one scoring particle ($m = 1$) and compute the expected answer of 0.001. But in this case the FSD is so large we would ordinarily reject this result! Continuing with the analysis, $\approx 1/5$ of the runs would yield a result a factor of 2 higher than the expected result, again with a large (but decreasing) FSD.

This analysis points out two pertinent facts. First, if your problem *is* under sampled and you have tallied few if any *real* scoring events, you can't easily tell! You may be spending all your time calculating a very precise *wrong* answer. To help you detect this problem, COG provides several scoring diagnostics which include the number and types of reactions in each problem REGION, the percent of the detector result which comes from the top ten scoring particles, etc.

The second fact obtained from Table 1 is that to improve statistics you need to increase the number of scoring events. This may be accomplished in either of two ways: by making longer runs, or by making use of biasing techniques to increase the probability of a scoring event. To demonstrate this, we show in Table 2 the outcomes for a sample size of $n = 10000$, which yields an expected number of scoring events $\langle m \rangle = 10$. Alternatively, we could achieve the identical outcomes by using the former

sample size $n = 1000$, but biasing the problem so that the probability of a source particle scoring is now $p = .01$.

TABLE 2

$n = 10000$ $\langle m \rangle = 10$ $\langle c \rangle = .001$			
# of scoring parts.	Pr(m)	Detector Score c	FSD
m			
0	4.5×10^{-5}	$1. \times 10^{-6}$	0.005
1	4.5×10^{-4}	0.0001	1.00
2	0.002	0.0002	0.71
3	0.007	0.0003	0.58
4	0.019	0.0004	0.50
5	0.037	0.0005	0.45
6	0.063	0.0006	0.41
7	0.090	0.0007	0.38
8	0.113	0.0008	0.35
9	0.126	0.0009	0.33
10	0.126	0.0010	0.32
11	0.114	0.0011	0.30
12	0.095	0.0012	0.29
13	0.073	0.0013	0.28
14	0.052	0.0014	0.27
15	0.035	0.0015	0.26
16	0.021	0.0016	0.25
17	0.013	0.0017	0.24
18	0.007	0.0018	0.24
19	0.004	0.0019	0.23
20	0.002	0.0020	0.22
21	0.001	0.0021	0.22

As can be seen from the table, the probability of getting no scoring events ($m = 0$) (and thus an answer too small by a factor of 1000) is now very small, < 0.0005 . The frequency distribution of possible outcomes is much more sharply peaked at the "true" result ($\langle m \rangle = 10$) and the FSD statistic values are much better. The problem is no longer under sampled. A better answer, as measured by a smaller fsd, requires an even larger $\langle m \rangle$.

15.5 Precision and Accuracy – a Reality Check

There is an extremely important distinction that should be made between the precision and accuracy of a Monte Carlo calculation. **Precision** is the uncertainty in $\langle x \rangle$ caused by the statistical fluctuation of the various x_i 's utilized in sampling the particular parameter space of the problem. **Accuracy** is a measure of how close the calculated value, $\langle x \rangle$, is to the *true* value, μ . The difference between the *true* value, μ , and the calculated value, $\langle x \rangle$, is often called ***the systematic error, and is seldom known very well.***

The sources of **systematic error** include errors in the database — the cross sections, errors in the angular scattering distributions, scattered energy distributions, etc. Errors can also be made in setting up the input model, or the material compositions. Errors can occur in the way the data is handled by the code. These errors can be minimized but never completely eliminated.

In a given Monte Carlo calculation, the quoted uncertainties refer **only** to the estimated **precision** of the result, **not its accuracy**. There are several things that can affect both the precision and the accuracy of a given calculation. Important portions of physical phase space may not have been sampled because of **incorrect energy and time cutoffs**, **inappropriate variance reduction** techniques (biasing) may have been used, **too few** particles employed in the calculation, **incomplete modeling** of the problem, **insufficient sampling** of low probability events, etc. The user needs to be concerned about all of these things in order to minimize their effects on his calculation.

Accuracy is mainly affected by four different factors:

- the data bases (cross sections) the code uses;
 - the Monte Carlo transport code methods;
 - the user's modeling of the problem;
 - the user's choice of code parameters.
- The available **cross section data sets are a major limitation on the accuracy of any Monte Carlo** code like COG. Remember that the cross sections are used in the exponent of transmission equations. For 20 mean free paths of particle transmission, a 10% uncertainty in the cross section may result in an answer that is "off" by a factor of 7. All cross sections are measured to only a limited accuracy. In particular, some neutron data comes from very approximate modeling of nuclear reactions. Cross sections for gammas are more amenable to modeling and are in much better shape accuracy-wise. Some neutron cross sections are measured very well and others have not been measured at all. In general only a few neutron cross sections have been measured above 20 MeV. If you want a satisfactory estimate of the *accuracy* of your

COG results, you will have to determine the accuracy of the cross sections used in most parts of your problem. This may not be a trivial job, but is absolutely essential for any reasonable accuracy estimate. One of the ironies of the semi-successful cross section modeling programs is that there is now very little activity in measuring cross sections, particularly for neutrons. The accuracy of the present data bases will likely not improve until someone finances the cost of new cross section measurements or until some breakthrough in cross-section modeling is made. • Another possible source of code errors is inadequate code physics models. Inadequacies can occur in the mathematical treatment of the models, the physics approximations employed, or in the coding of the models. As far as we have been able to determine by comparison to other Monte Carlo codes, mathematical models and experimental results, COG has included all the applicable physics to the appropriate level of detail needed to produce accurate results. No large code, such as COG, is bug free. However, COG's constant use over the years has ferreted out the vast majority of coding errors and makes the likelihood of major errors increasingly slight.

- Inadequate physical modeling of Monte Carlo problems by the user is another source of errors. Significant parts of the geometry may be over simplified, mixtures of materials (alloys) may be misrepresented as a single material. The source may be inadequately described in terms of energy, intensity and angular distribution. Discerning what level of detail to include in a given problem comes only with experience.

- Inappropriate use of the code will always be a problem. Variance reduction techniques (biasing) may be used in inappropriate ways. Other problems include: inadequate sampling of phase space, inadequate number of particles run, inappropriate detector specification or choice, inappropriate energy and/or time cut-offs. Undetected input errors can also occur. A thorough understanding of the quantities being calculated is needed. Small detector regions typically have very poor statistics. Detector efficiencies, data reduction, etc. all must be included in any meaningful accuracy estimate.

15.6 Estimated Errors in COG

All standard COG detector results are computed as results per source particle, multiplied by source strength. (Exception: if a STEADY-state source is used, then detector results are given per unit time). All standard COG detector results include the estimated fractional standard deviation (fsd) defined as: $\text{fsd} = s_m / \langle x \rangle$,

where

$\langle x \rangle$ is the mean detector result obtained by summing up each scoring event's contribution, then dividing by the number of source particles:

$$\langle x \rangle = \sum x_i / n ;$$

s_m is the estimate of the standard deviation of the mean result $\langle x \rangle$:

$$s_m = s / \sqrt{n} ;$$

where

s is the sample standard deviation:

$$s = [\sum (x_i - \langle x \rangle)^2 / (n - 1)]^{1/2}$$

The fractional standard deviation (fsd) or relative error is easy to use as it gives statistical **precision** as a fractional result. Guidelines for interpreting the values of fsd are given in the following table:

FSD Guideline Table

Range of FSD	Quality of Result	# of scoring events m
0.5 to 1.	Extremely poor	4 – 1
0.2 to 0.5	Factors of a few (2-5)	25 – 4
0.1 to 0.2	Borderline	100 – 25
< 0.10	Generally reliable (except for POINT detector)	> 100
< 0.05	Generally reliable for POINT detector	> 400

Discussions of precision versus accuracy are given in the previous section and should be referred to in interpreting the given fsd's in terms of accuracy. The number **m** is the estimated number of non-zero scores in the detector that would give the equivalent uncertainty in fsd. In a typical run, $m \ll n$, where n is the number of particles run; often very few particles actually contribute to the detector's score. The table is based on $\text{fsd} \approx 1/\sqrt{m}$, where m is the number of particles that are contributing to the score.

16 Electron Transport: Using the EGS5 Transport Code Kernel

COG solves coupled photon-electron transport problems using the EGS5 transport kernel embedded in COG. The EGS (Electron Gamma-Ray Shower) Code¹ is a widely-used coupled photon-electron code package. The advantage of running EGS inside COG is that the powerful COG Source, 3D Geometry package, and Detector (scoring) routines are available for the electron transport. As users of EGS know, EGS is not a stand-alone application like COG but a "box of cards", requiring the user to write, compile, and link his own source, geometry, and scoring routines to generate a stand-alone code. All this user labor is obviated by embedding the EGS transport kernel in COG.

Electron events are tallied in the COG Region Summaries, and scored by COG boundary-crossing and reaction detectors. Electron and positron sources may be used. However, electron transport calculations are slow, so electron transport should be enabled only for those regions of your problem where it is important for your results.

Whenever an electron-producing photon reaction occurs, COG checks whether the reaction occurred in a region enabled by the user for electron transport. If not (the standard case), then the electron energy is immediately deposited. If enabled, then the electrons are placed on the EGS kernel stack for transport. If secondary photons due to bremsstrahlung or positron annihilation occur during electron transport, these photons are placed on the COG stack for COG to follow.

16.1 Specifying Electron Transport

If you have an existing COG photon-transport problem input file, you can enable electron transport by adding the following lines.

To the **BASIC** Data Block, add the particle type **ELECTRON**.

Example of BASIC Data Block for electron transport:

```
BASIC
      PHOTON ELECTRON
```

This will cause COG to follow electrons in all regions specified as enabled in the EGS Data Block, and to score electrons in all problem detectors.

You can specify an electron source in the Source Data Block, using the *particle-type* **electron** in the definition of the Source ENERGY dependence.

16.1.1 The DNEAR Geometry Option

DNEAR is a geometry speed-up option. If enabled, the code computes the distance to the nearest boundary surface of the current sector containing the electron. The transport routine can then advance the particle for several steps, without making any more calls to the geometry module, until the accumulated track-length approaches this DNEAR distance. This can improve the speed of tracking by about 50%. The default of DNEAR OFF means that the geometry module will check each proposed step to see if it will cross a boundary.

To turn on the DNEAR option, add these words to the **BASIC** Block:

DNEAR ON DNEAR is off by default, because it may not result in a speed up in all cases. The user should determine if this option is useful for his particular job.

Use of DNEAR with the surfaces listed below will usually (but not always) result in a significant decrease of running time. Use of DNEAR with unlisted surfaces may increase running time, as some of the DNEAR option code must be executed before COG realizes that there is no DNEAR calculation available for the surface.

List of DNEAR-capable surfaces:

Axis-aligned planes

General plane

Spheres

Axis-aligned cylinders

Axis-aligned cones

Boxes

Finite cylinders and truncated spheres (arbitrary orientation)

16.2 EGS Data Block

The EGS Data Block tells the code where to find the electron scattering data file, specifies the problem sectors for which electron transport is enabled, and optionally specifies what lower-limit kinetic-energy cutoffs to use. The form of the EGS Data Block is:

EGS

CALLPEGS = *pegsin*

or

PEGSLIB = *pegsfilename*

ESECTORS = *sect-ID#1 sect-ID#2 ...*

{ **ECUT** = *ecut1 ecut2 ...* }

Where:

EGS is the keyword indicating the start of the EGS Data Block.

Option **CALLPEGS**=*pegsin* calls PEGS5 with input file *pegsin* to generate a PEGS library file, while

Option **PEGSLIB**=*pegsfilename* uses the existing PEGS library which must have previously prepared by the user. It should reside in the same directory as the COG input file. It contains electron collision data for all the materials in your COG problem, for which electron transport is enabled. At run time, COG matches the COG material definitions for enabled sectors with material descriptions in the PEGS cross-section file, by elemental composition and by density. Failure to find a match causes the code to terminate. See the COG Introduction: Overview of Electron Transport for information on obtaining the PEGS database generator.

sect-ID#1 sect-ID#2 ... are the ID numbers of the sectors for which the user desires electron transport to be enabled. Primary and secondary electrons will be tracked only in the specified regions. If an electron crosses from an enabled sector into a non-enabled sector, the electron energy is immediately dumped.

ecut1 ecut2 ... are the optional lower-limit kinetic-energy cutoff values for electron transport, in MeV, to be applied in one-to-one correspondence to sectors *sect-ID#1 sect-ID#2 ...*. The actual value which COG will use in a region is the greater of **ecut** and the PEGS lower-limit kinetic-energy cutoff value found in the **PEGSLIB** file. If more **ESECTORS** are specified than **ECUT** values, the last entered **ECUT** value will be applied to all subsequent sectors.

Example of EGS Data Block for electron transport:

```
EGS
PEGS LIB = PEGS1
ESECTORS = 1      10      20      34
ECUT =          0.4      0.7
```

16.2.1 Bremsstrahlung Angular Distribution

Instead of the default EGS5 model, COG uses the Koch and Motz² model for the Bremsstrahlung photon angular distribution.

16.2.2 Known Limitations

COG Limitations:

The only biasing mode available for electron transport is source biasing. Only Reaction and Boundary detectors will score electrons. Error-checking on electron transport inputs is rudimentary. Electron transport is not enabled for geometry units.

EGS Limitations:

COG uses the default EGS5 kernel, which does not include the PRESTA transport correction algorithm. (PRESTA adds lateral displacement to each step and controls step size artifacts.)

16.3 Example Problem

Example of an electron transport problem:

```

BASIC
  photon  electron $ particle types to follow
  cm MeV          $ problem units
  DNEAR ON        $ Turn on geometry speed-up feature
  RN 28172  619    $ Initialize RNG seeds

SURFACES
$  name      R    x1  x2
100 cyl    x 2.0  0.  10.
 23 plane x 3.0
 24 plane x 4.0
 25 plane x 5.0
900 cyl    x 3.0 -1. 11.

GEOMETRY
sector 101 src    -100 -23    $ source in air
sector 102 target -100 +23 -24 $ tungsten target
sector 103 water  -100 +24 -25 $ post-target water sector
sector 104 Air    -100 +25    $ following air sector
sector 900 vac1   +100 -900    $ vacuum surround
boundary vacuum          +900  $ exterior vacuum boundary

MIX
mat=1  N          9.740E-04  $ air (1.29E-03 gm/cc)
      O          2.993E-04
      Ar         1.677E-05
mat=2  W          19.3        $ W (19.3 gm/cc)
mat=3  water      1.0         $ h2o

ASSIGN-ML
1  101 104 /    $ Assign air to sectors
2  102    /    $ Assign tungsten to sector 102
3  103    /    $ Assign water to sector 103
0  900    $ Assign vacuum to sector 900

EGS
  pegslib=egsdat.01 $ PEGS library file containing electron
                    $ scattering data for all problem materials

```



```
esectors= 101 102 103 104 900 $ Designate sectors for electron transport
Ecut = 0.010      $ low-energy electron cutoff (MeV)
```

SOURCE

```
npart = 1000      $ # of source particle
DEFINE P=1        $ Position def. follows
  SS-DISK 0. 0. 0. 1. 0. 0. 0.5 $ Disk source of electrons
DEFINE E=1        $ Energy def. follows
  electron line 20. 1. $ 20 MeV electrons
DEFINE A=1        $ Angle def. follows
  1. 0. 0. FIXED $ pencil beam along +x-axis
```

```
INCREMENT 1.0 P=1 E=1 A=1 $ Defines total source
```

DETECTOR

```
number=deposlp Title="photon energy deposition in water"
REACTION 103 12.566 $ region # and region's volume
drf-e photon energy-deposition 2
```

```
number=deposle Title="electron/positron energy deposition in water"
REACTION 103 12.566 $ region # and region's volume
drf-e electron energy-deposition 2
drf-e positron energy-deposition 2
```

```
END
```

16.4 References

- 1.0 W. R. Nelson, H. Hirayama, and D. W. O. Rogers, The EGS5 Code System, SLAC-Report #265, Stanford University, (December, 1985).
Internet Address: <http://www.slac.stanford.edu/egs/>
- 2.0 H. W. Koch and J. W. Motz, Bremsstrahlung cross-section formulas and related data, Rev. Mod. Phys.,31, (1959).

17 Proton Transport

COG now includes the transport of protons through materials as one of its basic capabilities. Coupled (n,p) problems may also be solved.

The proton transport physics models take into account hadronic collisions with atomic nuclei, multiple small-angle Coulomb scattering, continuous energy loss to the medium via excitation and ionization, energy straggling, and deflection by vacuum magnetic fields, within the constraint that a proton hadronic collision is modeled as an absorption event (see below).

Proton transport uses the method of condensed histories; i.e. single small scattering events are not modeled, but rather the particle is advanced a macrostep in some direction, with a step length large enough to encompass many small scattering events along the way. At the end of the linear step, a multiple small-angle scattering distribution is sampled to obtain the net scattering angle (in Gaussian approximation), and the next step proceeds in this new direction.

Particles lose energy along the step through excitation and ionization of the media, which is modeled using a stopping-power formulation in the Continuous Slowing Down Approximation (CSDA). To account for the stochastic variation in energy loss over a step, an energy-straggling distribution is sampled.

Infrequently, a proton undergoes a hadronic collision with a nucleus. This is currently treated as a pure absorption event.

17.1 Hadronic Collisions

A proton collision with an atomic nucleus is modeled as a terminal event; the proton is absorbed, its energy is deposited, and no secondaries are produced.

The cross section for this event can be chosen to be either:

- The pp inelastic + elastic cross section,¹ scaled by nuclear area ($A^{0.77}$);
- The total inelastic cross section for proton nuclear scattering, as modeled by Letaw et al.² Because the Letaw model fails for $Z=1$, we use the pp inelastic model (above), for this case.

17.2 Multiple Small-angle Coulomb Scattering

The distribution of net scattering angle, after the proton has undergone many small-angle Coulomb scattering events along a step, is treated in Gaussian approximation. The models available are:

- Highland's approximation to the Moliere projected scattering-angle distribution's sigma³;
- The Lynch-Dahl approximation to the Moliere multiple small-angle scattering distribution (the default)³;
- Above model of Lynch and Dahl, but with a fixed number of collisions.³

The Lynch-Dahl default model gives excellent results for a single step, but tends to underestimate the true angle after many steps. This is because the Gaussian approximation ignores the tail of the Moliere distribution, and the errors accumulate. The Lynch-Dahl model with the fixed number of collisions (a parameter of the Lynch-Dahl formula) generally yields better results. Fixing the number of scattering events results in an overestimate of the sigma of the single-step distribution. In light elements at 1 GeV, the error is 10% - 20% for a fractional energy loss Estep of 0.001. This is therefore a poor model for thin foil problems. The advantage of the model is that this overestimate of a single transport step partially compensates for the accumulating Gaussian undershoot after many transport steps.

For example, after 10 transport steps at Estep=0.001 at E=1 GeV, the error in sigma, for a wide range of Z, is better than 5% (compare to 18% for the standard Lynch-Dahl model).

17.3 Continuous Slowing Down Approximation (CSDA)

The CSDA models used are the Anderson-Ziegler model⁴ (for non-relativistic proton energies), and the Bethe-Bloch model⁵ (with Sternheimer density-effect coefficients⁶ for relativistic protons).

17.4 Energy Straggling

All protons of a given energy in a specified medium will experience the same CSDA energy loss. To produce a realistic distribution of proton energies after a step, COG employs the Landau energy-straggling formalism. For this theory to be valid, the ratio of mean energy lost in the step, to the maximum energy transfer to an atomic electron in a single collision, must be < 0.01. This restricts the step size.

17.5 Transport Step Size Control

The transport step size is determined by the code to be the minimum of:

- an energy-loss-constrained step;
- the distance to the nearest boundary along the trajectory;
- the distance to a hadronic collision.

The energy-loss-constrained step is computed from the user-specified maximum fractional energy loss per step (ESTEP) due to excitation and ionization of the media, OR it is the maximum step which still ensures the validity of the Landau energy-straggling distribution.

This procedure is followed:

If ESTEP has been specified, then compute step size from ESTEP and dE/dX tables:

$$\text{stepsize} = E * \text{ESTEP} / (dE(E)/dX).$$

Else if energy-straggling is enabled, compute stepsize using the maximum average energy loss allowed for the Landau straggling distribution to be valid (determined by the Landau parameter Kappa).

Else, determine stepsize by using a default $\text{ESTEP} = 0.001$ (proton will lose 0.1% of its kinetic energy over the step).

17.6 Transport in a Magnetic Field

COG will now transport protons in a static vacuum magnetic field confined to a finite cylinder, or other geometrically-simple volume. Once the proton has entered the field, by passing through one of the specified endplanes which bound the magnet, COG solves the Lorentz force equation for the proton trajectory. When the particle leaves the field by exiting through an endplane, normal transport resumes. The user can choose to check the particle's position after every step, to look for wall collisions (**BBOUNDCHECK** Option). The default, faster method is to only check after the particle has exited the field region. Currently, only a quadrupole magnetic field is supported. See the **BFIELD** Data Block below for a description of the magnetic field inputs.

17.7 Setting Up a Proton Transport Problem

To enable proton transport, the *particle-type* **proton** must appear if the **BASIC** Data Block. To solve (n,p) problems, *particle-type* **neutron** must also appear. You can specify a proton source in the Source Data Block, using the *particle-type* **proton** in the definition of the Source ENERGY dependence.

Additionally, the user may elect to turn various proton physics options ON or OFF, and specify models and their parameters. The following proton-specific options can be specified in the **BASIC** Data Block:

HADRONIC PROTON MODEL *modelname representation* [ON/OFF]

where:

modelname is one of:

PPTOTAL : Total pp inelastic + elastic cross section, scaled by nuclear area.

LETAW : Model of Letaw et al. (**Default** model).

representation is one of:

FUNCTION : Direct call to the functional model.

TABLE : Use the pre-computed table.

Default [ON/OFF] is ON.

MSCAT PROTON [ON/OFF]

Turns on/off multiple small-angle scattering (**default** is ON).

MSCAT PROTON MODEL *modelname*

Selects multiple scattering model, where:

modelname is one of:

HIGHLAND : Highland approximation, updated by Lynch and Dahl.¹

LYNCH : Model of Lynch and Dahl,³ improved single-step error (**default**).

LYNCHFC : Simplified Lynch-Dahl model³ with Fixed number of Collisions.

DEDX PROTON [ON/OFF]

Turns on/off ionization and excitation energy losses. **Default** is **ON**.

ESTRAGGLE PROTON [ON/OFF]

Turns on/off Landau energy straggling. **Default** is **ON**.

ESTEP PROTON *fraceval*

Sets proton transport step size so that the CSDA energy loss over the step is *fraceval**E0, the proton kinetic energy at the start of the step.

If **ESTRAGGLE** is turned **OFF** and **ESTEP** is not specified, the code will assume an **ESTEP** value of 0.001 for step-size calculations.

DNEAR [ON/OFF]

Turns the **DNEAR** geometry option on to speed up transport. **Default** is **OFF**. See the preceding **Electron Transport** section for a discussion of the DNEAR option.

BBOUNDCHECK [ON/OFF] - turns on the magnetic-sector boundary checking feature, for a problem containing a magnetic-field sector, as specified in the BFIELD Block. **Default** is **OFF**.

If **BBOUNDCHECK** is **OFF**, then no boundary checking will be done during the tracking of particles in a magnetic field. The particle will be stepped to the exit endplane, then its position will be checked to see if it has still not penetrated the magnet walls. If it has, then it is just deposited in the wall at the exit plane.

If **BBOUNDCHECK** is **ON**, then we check the particle's position after each step to ensure that the particle has not collided with the walls. The DNEAR geometry feature is automatically turned ON to minimize geometry calls.

17.8 Low-Energy Cutoffs

Transport of protons whose kinetic energy falls below a user-specified threshold can be terminated, thus saving computer time. See the **Walk-ENERGY** Option in the **Monte Carlo Random Walk** Section.

17.9 Detectors

Only BOUNDARY-CROSSING and REACTION detectors may be used with protons. If ENERGY BINS are specified, they yield these results:

- BOUNDARY-CROSSING: the quantity scored is the proton's incident energy at the boundary;
- REACTION: the quantity scored is energy deposited by a proton during each transport step in the detector volume. The bin is selected based on proton kinetic energy at start of the incremental transport step in the volume.

17.10 Known Limitations

Proton transport is not enabled for geometry units.

17.11 References

- 1.0 Baldini, V. Flaminio, and O. Yushchenko, Particle Data Book, Phys. Rev. D 45, III.83, (1992).
- 2.0 J. R. Letaw, R. Silberberg and C. H. Tsao, Proton-nucleus total inelastic cross sections: An empirical formula for $E > 10$ MeV, Astrophysical Journal Suppl. Series, 51, 271, (1983).
- 3.0 G. Lynch and O. Dahl, Approximations to multiple Coulomb scattering, Nucl. Inst. and Meth., B58, 6-10, (1991).
- 4.0 H. H. Anderson and J. F. Ziegler, Hydrogen, Stopping Powers and Ranges in All Elements, in "The Stopping and Ranges of Ions in Matter," Vol. 3, (organized by J. F. Ziegler, Pergamon Press, 1977).
- 5.0 Particle Data Book, Phys. Rev. D, 54, 132, (1996).
- 6.0 R. M. Sternheimer, M. J. Berger, and S. M. Smeltzer, Density effect for the ionization loss of charged particles in various substances, At. Data and Nuc. Data Tables, 30, 261-271, (1984).

17.12 BFIELD Data Block

17.12.1 Magnetic Field Sectors

This Data Block is used to specify static vacuum magnetic field sectors through which protons will be transported.

The format of this Block is:

BFIELD

SECTOR *secno* **MODEL** *model_name*

PMORIGIN *pmox pmoy pmoy*

PMAxis *pmax pmay pmaz*

PBAXIS *pbax pbay pbaz*

ENDS *z1 z2*

model_parm1 model_parm2 ...

where:

secno is the sector # of the COG sector containing the vacuum magnetic field;

model_name is the name of the field type to be generated inside *secno*;

e.g., "quad" for a quadrupole field (only type implemented)

PMORIGIN, **PMAxis**, and **PBAXIS** are three optional points which respectively specify the Magnetic Field Origin, Direction, and Beam Direction in each magnetic sector.

PMORIGIN:

(*pmox, pmoy, pmoy*) are the COG coordinates of the Point representing the Magnetic field Origin in the sector;

Default is (0,0,0) (the COG origin).

PMAxis:

(*pmax, pmay, pmaz*) are the COG coordinates of a Point lying on the Magnetic field Axis in the sector;

Default is (0,0,1) (the COG z-axis).

PBAxis:

(*pbax, pbay, pbaz*) are the COG coordinates of a Point lying on the Beam Axis through the sector;

Default is (0,0,1) (the COG z-axis).

ENDS z1 z2 gives the location of the endplanes, normal to the beam axis, which bound the magnetic sector. Magnetic field transport is only enabled between the **ENDS**.

model_parm1 ... are the parameters needed to generate the field *model_name* specified on the **SECTOR** line ;

For a quadrupole field, the *model_parm1 ...* parameters are:

Pole_tip_field Magnet_length Magnet_radius

where the field is in kiloGauss, and the distances in cm.

See the **BBOUNDCHECK** Option above for a description of options for checking for proton-wall collisions while tracking in magnetic fields.

17.12.2 BFIELD Limitations

Setup will be simpler if the magnetic sector is a z-oriented cylinder, and the magnetic axis and the beam axis lie along z.

A proton detector can't have the magnetic sector as one of its defining sectors.

17.13 Example Problem

Example of a proton transport problem:

```

PROTON PROBLEM: 1 GEV PROTONS INTO WATER
BASIC
  CM          $ UNIT OF LENGTH
  GEV         $ UNIT OF ENERGY
  PROTON      $ PARTICLE TYPE TO BE FOLLOWED
  ESTRAGGLE PROTON ON $ TURN ON THE LANDAU ENERGY STRAGGLING
$DISTRIBUTION
  MSCAT PROTON MODEL LYNCHFC $ USE THE LYNCH_DAHLL FIXED
$COLLISION MULTIPLE-SCATTERING MODEL
  DNEAR ON $ TURN ON OPTION TO REDUCE # OF CALLS TO
$GEOMETRY PACKAGE
  RN 27500      5229 $ SPECIFY THE STARTING RANDOM NUMBER
                  $GENERATOR SEEDS

SURFACES
$ #    TYPE    R      X1    X2
  100  CYL     5.    -0.1 1000.

  200 PLANE X    0.
  201 PLANE X 100.
  202 PLANE X 200.
  203 PLANE X 300.
  204 PLANE X 400.
  205 PLANE X 500.
  206 PLANE X 600.
  207 PLANE X 700.
  208 PLANE X 800.

  900  CYL    22.    -1. 1001.

GEOMETRY
  SECTOR 100 CYL1 -100 +200 -201
  SECTOR 101 CYL2 -100 +201 -202
  SECTOR 102 CYL3 -100 +202 -203
  SECTOR 103 CYL4 -100 +203 -204
  SECTOR 104 CYL5 -100 +204 -205
  SECTOR 105 CYL6 -100 +205 -206
  SECTOR 106 CYL7 -100 +206 -207
  SECTOR 107 CYL8 -100 +207 -208

  SECTOR 201 VOID1 +100 -900 $ VOLUME OUTSIDE OF 100 AND INSIDE
  $900 IS A VOID
  BOUNDARY VACUUM +900 $ VOLUME OUTSIDE SURFACE 900 IS A VACUUM
$BOUNDARY

```

```
SOURCE
NPART = 2000 $ $ # OF SOURCE PARTICLES TO RUN
DEFINE POSITION = 1 $ DEFINITION OF SOURCE POSITION FOLLOWS
  POINT 0. 0. 0. $ POINT AT THE ORIGIN
DEFINE ENERGY = 1 $ DEF. OF SOURCE ENERGY-DEPENDENCE FOLLOWS
  PROTON
  LINE 2.0 1. $ LINE SOURCE AT 1 GEV, STRENGTH =1.
DEFINE ANGLE = 1. $ DEF. OF ANGLE-DEPENDENCE OF SOURCE FOLLOWS
  1. 0. 0. FIXED $ PENCIL BEAM ALONG +X-AXIS
INCREMENT 1. P=1 E=1 A=1 $ DEFINES TOTAL SOURCE

I/O
PRINTPMDEDX PRINTPMXS $ LIST MATERIAL STOPPING POWER AND
$NUCLEAR CROSS SECTION
MIX
MAT=1 H2O 1.0 $ WATER AT UNIT DENSITY

ASSIGN-ML
  1 100 101 102 103 104 105 106 107 \ $ ASSIGN MATERIAL 1
  0 201 $ ASSIGN MATERIAL 0 (VOID) TO SECTOR 101

DETECTOR
$ SCORE # OF PROTONS PASSING THROUGH EACH BOUNDARY DETECTOR,
$ PER SOURCE PARTICLE.
NUMBER = 1
BOUNDARY COUNTS 100 101 1.

NUMBER = 2
BOUNDARY COUNTS 101 102 1.

NUMBER = 3
BOUNDARY COUNTS 102 103 1.

NUMBER = 4
BOUNDARY COUNTS 103 104 1.

NUMBER = 5
BOUNDARY COUNTS 104 105 1.

NUMBER = 6
BOUNDARY COUNTS 105 106 1.

NUMBER = 7
BOUNDARY COUNTS 106 107 1.

END
```

18 Alpha Transport - ALPHATRANS

Alpha transport is handled as a two step process...

Step 1: CSDA. Using data and coding borrowed from the AlfaMC code¹², a Continuous Slowing Down Approximation for alphas has been implemented in COG. Following the guide of the AlfaMC code: the NIST/ASTAR¹³ stopping-power database is used; Gaussian, Vavilov, or Landau distributed energy straggling is performed; and a simple Fermi small-angle multiple scattering model is adopted. Since for the energy range under consideration (~1 to 20 MeV), the mean free path for nuclear interactions (e.g. (alpha,neutron), (alpha,gamma), etc.) is very much greater than the range of the alpha, the CSDA approach alone is used to track the alpha. That is nuclear interactions are ignored in this step.

Step 2: Nuclear interactions. A fraction (see below) of the CSDA steps are sampled for nuclear reactions – that is, for each step if a random number is less than the fraction, then nuclear reactions are included and any (appropriately weighted) secondaries are produced and followed.

How to: In the COG input deck, there is a new block – **alphatrans**.

The inputs are...

matat *mat1 mat2 ...* where *mat1*, *mat2*, ... are the material numbers (from the **mix** block) in which alpha transport is allowed, this input is required and must be the first entry in the block.

adestep *de* where *de* is the fractional energy loss per step, **default** is 0.01

astragflag *flag* where *flag* = 0 for Gaussian only, = 1 for Gaussian, Vavilov, or Landau depending on energy, **default** is 0 (option 1 increases running time)

astepmin *stmin1 stmin2 ...* where *stmin1*, *stmin2*, ... are minimum steps in cm for *mat1*, *mat2*, ..., must be > 1.e-8, **default** is 1.e-6

astepmax *stmax1 stmax2 ...* where *stmax1*, *stmax2*, ... are maximum steps in cm for *mat1*, *mat2*, ..., must be < 10., **default** is 1.e-3

aecut *ecut1 ecut2 ...* where *ecut1*, *ecut2*, ... are cutoff energies in MeV for *mat1*, *mat2*, ..., must be > 0.001, **default** is 0.01

afrac $f1, f2, \dots$ where $f1, f2, \dots$ are the fraction of CSDA steps, on average, with nuclear reactions for $mat1, mat2, \dots$, (this factor should be determined by trial and error, too low and you get few, if any, secondaries, too high and you are swamped with secondaries), **default** is 0.01

Note: If an entry is made for one material it must be entered for each material, otherwise COG doesn't know how to assign values.

18.1 Alpha Transport Example 1

A pencil beam of 5.48 MeV alphas impinging on a 1000 μ thickness of water. Output is energy deposited vs. depth in the water. Figure 1 shows a comparison of COG and AlfaMC.

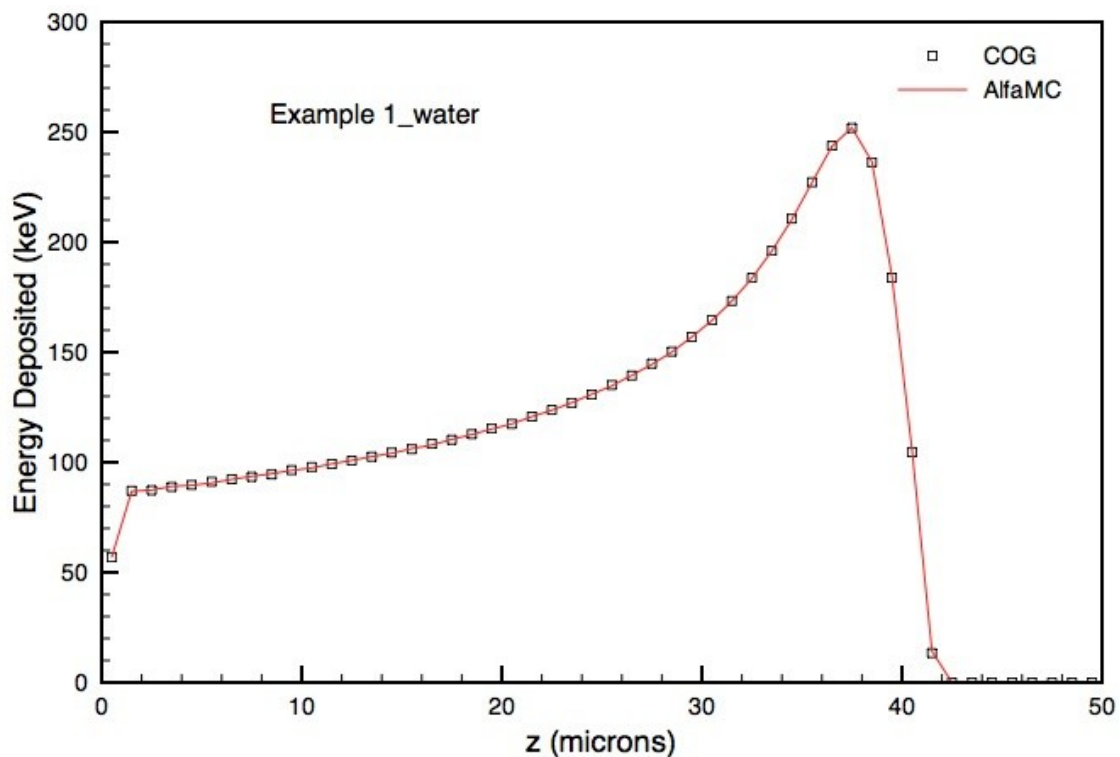


Figure 1

18.2 Example 1 COG input deck:

pencil beam of 5.48 MeV alphas impinging on a 1000 μ thickness of water

\$

basic

alpha

rn 9783 4507

alphatransport

matat 1

geometry

sector 1 h2o -1 2 -3

sector 2 h2o -1 3 -4

sector 3 h2o -1 4 -5

sector 4 h2o -1 5 -6

sector 5 h2o -1 6 -7

sector 6 h2o -1 7 -8

sector 7 h2o -1 8 -9

sector 8 h2o -1 9 -10

sector 9 h2o -1 10 -11

sector 10 h2o -1 11 -12

sector 11 h2o -1 12 -13

sector 12 h2o -1 13 -14

sector 13 h2o -1 14 -15

sector 14 h2o -1 15 -16

sector 15 h2o -1 16 -17

sector 16 h2o -1 17 -18

sector 17 h2o -1 18 -19

sector 18 h2o -1 19 -20

sector 19 h2o -1 20 -21

sector 20 h2o -1 21 -22

sector 21 h2o -1 22 -23

sector 22 h2o -1 23 -24

sector 23 h2o -1 24 -25

sector 24 h2o -1 25 -26

sector 25 h2o -1 26 -27

sector 26 h2o -1 27 -28

sector 27 h2o -1 28 -29

sector 28 h2o -1 29 -30

sector 29 h2o -1 30 -31

sector 30 h2o -1 31 -32

sector 31 h2o -1 32 -33

sector 32 h2o -1 33 -34
sector 33 h2o -1 34 -35
sector 34 h2o -1 35 -36
sector 35 h2o -1 36 -37
sector 36 h2o -1 37 -38
sector 37 h2o -1 38 -39
sector 38 h2o -1 39 -40
sector 39 h2o -1 40 -41
sector 40 h2o -1 41 -42
sector 41 h2o -1 42 -43
sector 42 h2o -1 43 -44
sector 43 h2o -1 44 -45
sector 44 h2o -1 45 -46
sector 45 h2o -1 46 -47
sector 46 h2o -1 47 -48
sector 47 h2o -1 48 -49
sector 48 h2o -1 49 -50
sector 49 h2o -1 50 -51
sector 50 h2o -1 51 -52
boundary vacuum 1 or -1 -2 or -1 52

surfaces

1 c z 1.
2 p z 0.
3 p z 0.0001
4 p z 0.0002
5 p z 0.0003
6 p z 0.0004
7 p z 0.0005
8 p z 0.0006
9 p z 0.0007
10 p z 0.0008
11 p z 0.0009
12 p z 0.001
13 p z 0.0011
14 p z 0.0012
15 p z 0.0013
16 p z 0.0014
17 p z 0.0015
18 p z 0.0016
19 p z 0.0017

20 p z 0.0018
21 p z 0.0019
22 p z 0.002
23 p z 0.0021
24 p z 0.0022
25 p z 0.0023
26 p z 0.0024
27 p z 0.0025
28 p z 0.0026
29 p z 0.0027
30 p z 0.0028
31 p z 0.0029
32 p z 0.003
33 p z 0.0031
34 p z 0.0032
35 p z 0.0033
36 p z 0.0034
37 p z 0.0035
38 p z 0.0036
39 p z 0.0037
40 p z 0.0038
41 p z 0.0039
42 p z 0.004
43 p z 0.0041
44 p z 0.0042
45 p z 0.0043
46 p z 0.0044
47 p z 0.0045
48 p z 0.0046
49 p z 0.0047
50 p z 0.0048
51 p z 0.0049
52 p z 0.005

assign-ml

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

mix

nlib JENDL3.3

alib A.TENDL2013

mat 1 w-p 1. h1 0.111894 o16 0.888106

```
source
npart 1000000
define p 1
  point 0. 0. 0.
define e 1
  alpha line 5.48 1.
define a 1
  0. 0. 1. fixed
inc 1. p 1 e 1 a 1
end
```

19 Deuteron Transport - DEUTTRANS

Deuteron transport is modeled after alpha transport (see Alpha Transport Data Block)

How to: In the COG input deck, there is a new block – **deuttrans**.

The inputs are...

matdt *mat1 mat2 ...* where *mat1, mat2, ...* are the material numbers (from the **mix** block) in which deuteron transport is allowed, this input is required and must be the first entry in the block.

ddestep *de* where *de* is the fractional energy loss per step, **default** is 0.01

dstragflag *flag* where *flag* = 0 for Gaussian only, = 1 for Gaussian, Vavilov, or Landau depending on energy, **default** is 0 (option 1 increases running time)

dstepmin *stmin1 stmin2 ...* where *stmin1, stmin2, ...* are minimum steps in cm for *mat1, mat2, ...*, must be > 1.e-8, **default** is 1.e-6

dstepmax *stmax1 stmax2 ...* where *stmax1, stmax2, ...* are maximum steps in cm for *mat1, mat2, ...*, must be < 10., **default** is 1.e-3

decut *ecut1 ecut2 ...* where *ecut1, ecut2, ...* are cutoff energies in MeV for *mat1, mat2, ...*, must be > 0.001, **default** is 0.01

dfrac *f1, f2, ...* where *f1, f2, ...* are the fraction of CSDA steps, on average, with nuclear reactions for *mat1, mat2, ...*, (this factor should be determined by trial and error, too low and you get few, if any, secondaries, too high and you are swamped with secondaries), **default** is 0.01

Note: If an entry is made for one material it must be entered for each material, otherwise COG doesn't know how to assign values.

19.1 Deuteron Transport Example 1

A pencil beam of 40keV deuterons impinging on a 2.23 μ thickness of titanium tritide. Output is neutrons/deuteron from the $H^3(d,n)He^4$ reaction. Figure 1 shows a comparison of COG calculation with measurement

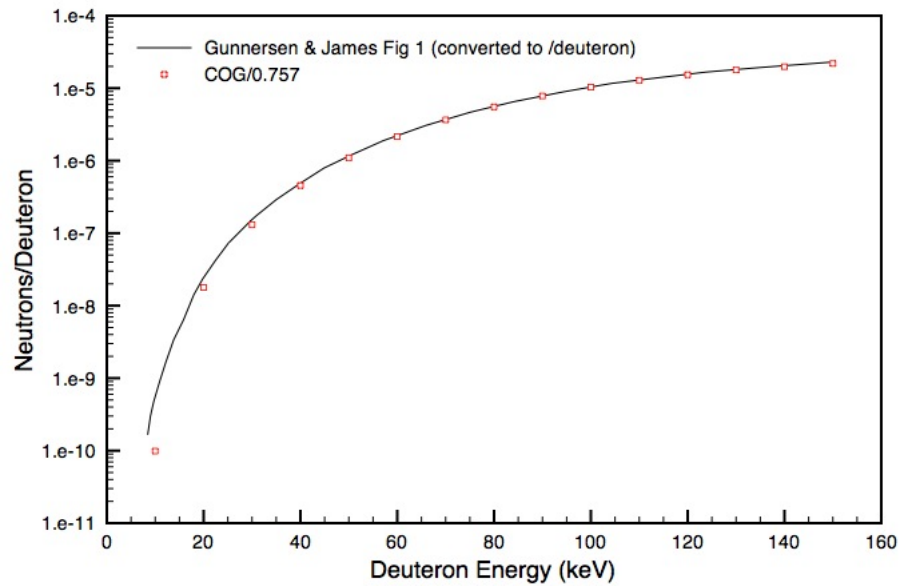


Figure 1

19.2 Example 1 COG input deck

Example 1 - neutrons/deuteron - Gunnarsen& James

\$

basic

neutron

deuteron

rn 7390 10066

deuttransport

MatDT 1

\$ DStepMin 1.e-5

\$ DStepMax 1.e-4

\$ DECut 0.001

\$ DFrac 0.01

geometry

sector 1 foil -1

sector 2 vac 1 -2

sector 3 det 2 -3

boundary vacuum 3

surfaces

1 c z 0.5 -0.0001115 0.0001115
2 s 10.
3 s 11.

mix

nlib ENDFB7R1
dlib D.ENDFB7R1
mat 1 a-f 4.22 Ti48 1. H3 1.5

assign-ml

0 2 3

detector

number 1
boundary 2 3 1.
drf-e neutron number-flux
bin e neutron 101 [0. i 20.]

source

npart 100000
define p 1
point 0. 0. -1.
define e 1
deuteron line 0.04 1.
define a 1
0. 0. 1. fixed
inc 1. p 1 e 1 a 1

end

20 COG Example Problem 1

20.1 Problem 1 Description and Input

This is a very simple shielding problem that illustrates a problem's input and output. Assume we have a cobalt-60 gamma-ray source and a Geiger-Müller tube, and we want to count gamma rays in a particular location. To make it more interesting, there are lead bricks to put between the source and the detector.

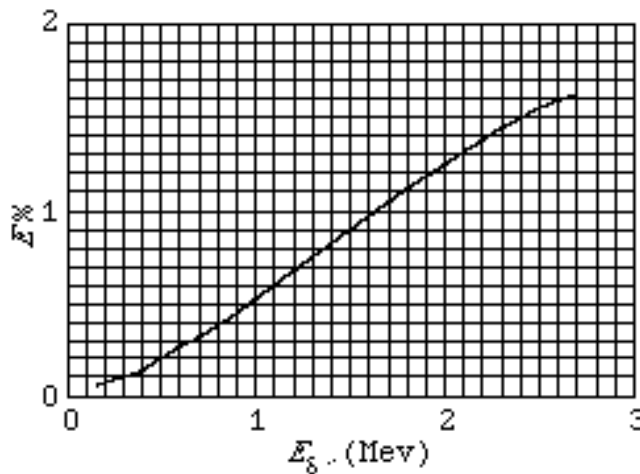
The experiment is in a room 30 feet long by 20 feet wide and with a 10-foot high ceiling. All the walls and the ceiling and the floor are formed of 12-inch thick concrete and, for purposes of calculation, we may forget doors, heating ducts and other minor features.

There is a single table at the center of the room with dimensions of 3 feet wide by 5 feet long and oriented with the longest length coincident with the longer dimension of the room. The table is made of white oak (12% moisture) 3 inches thick and with its top 36 inches from the floor. Four wood legs, each 4 inches square, are located at the table's corners.

The gamma-ray source is enclosed in a cylindrical aluminum holder 1 inch in outside diameter and 1 inch in outside length and with 1/16-inch thick walls. This is oriented with the axis perpendicular to the table top and is held by a 1/4-inch diameter aluminum rod so the center of the cylinder is 12 inches above the table and 24 inches to the left of the center of the table.

The cylindrical holder contains cobalt metal that has been activated so as to have a source strength of 1 millicurie (3.7×10^7 disintegrations per second). Each disintegration yields one 1.17-MeV and one 1.33-MeV gamma ray.

A Geiger-Müller (G-M) counter is located 24 inches to the right of the center of the table—i.e. there are 4 feet between the counter and the source. The counter has an aluminum cathode and a response efficiency given by the curve below. The count rate of the detector is given by the product of the gamma flux at the detector, the response efficiency, and the detector area. We will assume that the area is 1 square centimeter. Note that we are roughly talking about 1 count per second for a flux of 500 gamma rays/square centimeter-second. Dead time corrections need not be considered, and the detector need not be modeled in detail because its response includes the actual detector geometry.



Lead bricks are piled in the center of the table to give a stack 16 inches high by 8 inches wide and 1.5 inches thick (the thickness is measured along the direction measured between the source and detector).

Forget all other items in the room.

What count rate does the G-M counter register?

Now if you really want to see if you understand the COG input, go ahead and set the problem up and try running it without looking at the rest of this chapter. The concrete's specification can be represented by the material CNCRT2 in the COG library. To save even more time, we will give you the most difficult piece of needed information—the composition of the wood in the table:

carbon	0.32 grams/cm ³
hydrogen	0.04
oxygen	0.30
nitrogen	0.07
water	0.04

If you succeed in running the problem you should get an answer of about 0.29 counts/second. If you got near this, give yourself a gold star and skip to the next example.

COG input file for Example Problem 1 ...

sample shielding problem

\$

basic

photon

in

mix

mat 1 cncrt2 2.51

mat 2 c 0.32 h 0.04 o 0.03 n 0.07 h2o 0.04 \$oak 12% moisture

mat 3 al 2.7

mat 4 co 8.85

mat 5 pb 11.4

mat 6 air 0.00129

geometry

sector 1 walls -1 +2

sector 2 table -3 or -4 or -5 or -6 or -7

sector 3 holder -8 or -9 +10

sector 4 cobalt -10

sector 5 lead -11

fill 6

boundary vacuum +1

picture cs m color -36. 0. 12. -36. 0. -49. 36. 0. -49.

surfaces

1 box 384 264 144 tr 0 0 12 \$ outside room walls

2 box 360 240 120 tr 0 0 12 \$ inside room walls

3 box 60 36 3 tr 0 0 -13.5 \$ table top

4 box 4 4 33 tr -28 -16 -31.5 \$ table leg

5 box 4 4 33 tr -28 16 -31.5 \$ table leg

6 box 4 4 33 tr 28 16 -31.5 \$ table leg

7 box 4 4 33 tr 28 -16 -31.5 \$ table leg

8 cyl 0.125 0.5 12 tr -24.0 0.0 0.0 -24.0 0.0 -120.0 0 -1 0 \$ rod

9 cyl 0.5 -0.5 0.5 tr -24.0 0.0 0.0 -24.0 0.0 -120 0 -1 0 \$ container

10 cyl 0.4375 -0.4375 0.4375 tr -24.0 0.0 0.0 -24.0 0.0 -120.0 0 -1 0 \$ src

11 box 1.5 8 16 tr 0.0 0.0 -4.0 \$ lead bricks

```

detector
number 1
point 24. 0. 0.
drf-e photon $ gm tube sens for al cathode
0.1 0. 0.2 8.0e-4 0.3 1.0e-3 0.4 1.6e-3
0.5 2.1e-3 0.6 2.6e-3 0.7 3.3e-3 0.8 4.0e-3
0.9 4.5e-3 1.0 5.1e-3 1.1 5.9e-3 1.2 6.6e-3
1.3 7.2e-3 1.4 8.0e-3
bin e photon
16 [0. i 1.5]

source
npart 50000
define p 1
cylinder -24. 0. -0.4375 -24. 0. 0.4375 0.4375
define t 1
steady
define e 1
photon line 1.17 1. 1.33 1.
inc 7.4e7 e 1 p 1 t 1

walk-collision
photon region all energy 0. to 0.1 0.

End

```

A brief explanation of the input ...

- The first line is the problem title.
- BASIC Data Block: Photon only problem. Dimensions are in inches, energy and time units default to MeV and seconds respectively.
- MIX Data Block: Material 1 is concrete (cnct2) at density 2.51 gm/cm³, material 2 is white oak at 12% moisture, etc. Since there are no ASSIGN Data Blocks, the defaults are in force and sector 1 contains material 1, sector 2 contains material 2, etc.
- GEOMETRY Data Block: Sector 1 is the concrete walls of the room, sector 2 is the oak table top and 4 oak table legs, sector 3 is the aluminum source holder, sector 4 is the cobalt source, sector 5 is the lead bricks, and air fills the undefined space. The vacuum boundary tells the code not to track particles which penetrate through the room walls. A cross sectional picture of the geometry is requested.

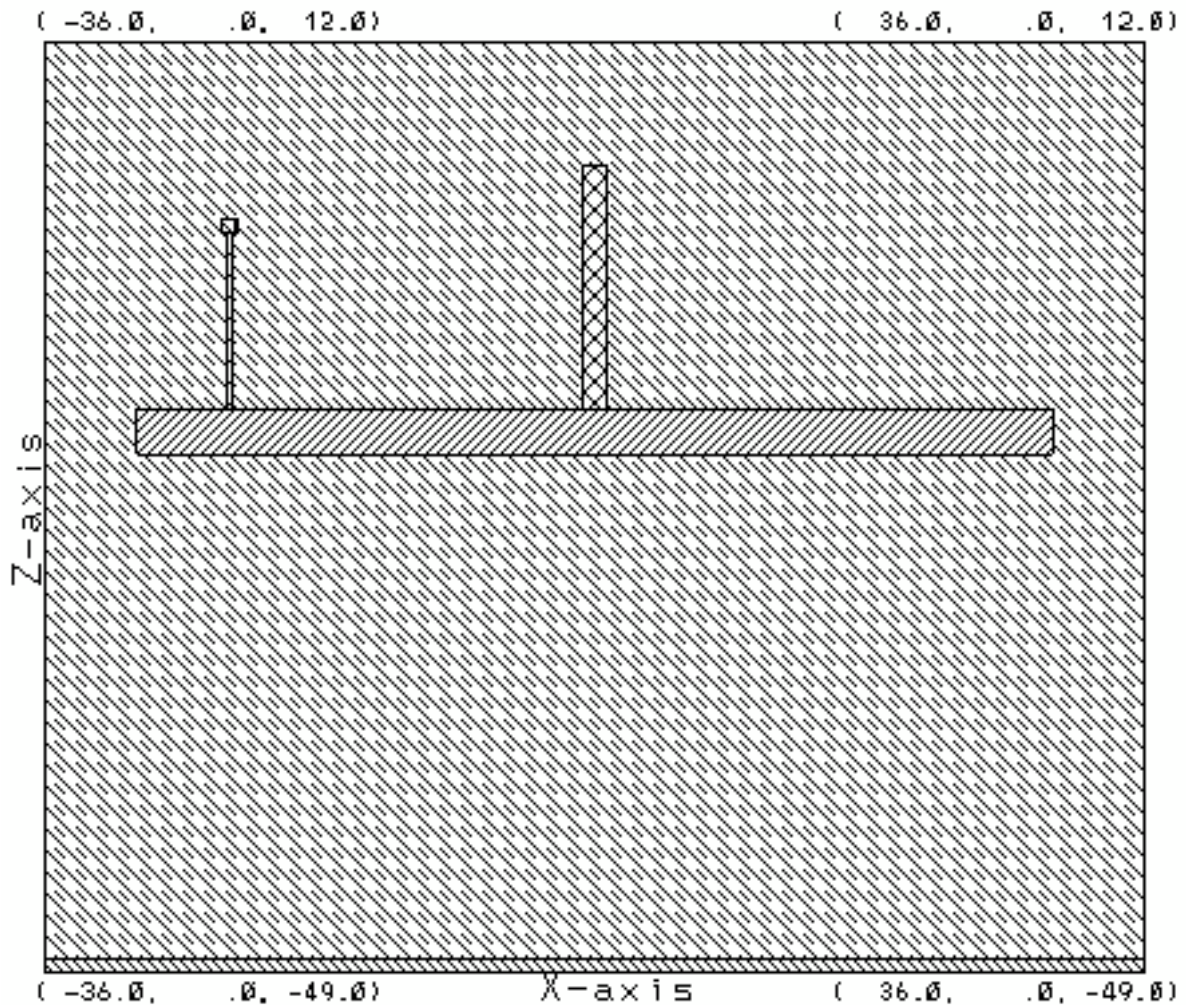
- **SURFACE Data Block:** Describes the various surfaces which make up the geometry. For example the room walls are defined as inside surface 1 (a box with x-,y-,z-sides of 384, 264, and 144 respectively and with center at $z = 12$) and outside surface 2 (a box with x-,y-,z-sides of 360, 240, and 120 respectively and with center at $z = 12$). All dimensions are in inches.
- **DETECTOR Data Block:** The single detector is a point detector located at (24.,0.,0.), all dimensions are in inches. The detector has the given response function (taken from the above graph). Differential results, in the form of 15 uniform bins running from 0. to 1.5 MeV, are requested.
- **SOURCE Data Block:** The source consists of 50000 particles taken from a distribution whose position dependence is a cylinder of radius 0.4375 and whose axis extends from (-24.,0.,-0.4375) to (-24.,0.,0.4375) (all dimensions are in inches), whose time dependence is steady state, and whose energy dependence is equal intensity, photon lines at 1.17 and 1.33 MeV. The source is normalized to 7.4×10^7 (3.7×10^7 disintegrations per second times 2 lines).
- **WALK-COLLISION Data Block:** Low energy (below 0.1 MeV) photons in all regions are killed. Since there are no ASSIGN Data Blocks, the defaults are in force and region 1 is sector 1, region 2 is sector 2, etc.
- **END Data Block:** End of input.

20.2 Problem 1 Results

20.2.1 The ps file for Problem 1

What follows are the various output plots along with a brief description of each.
sample shielding problem

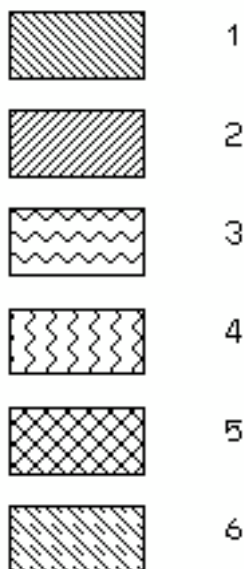
picture cs m -36.0, 0.0, 12.0, -36.0, 0.0, -49.0, 36.0, 0.0, -49.0
12:57:36 15-Aug-94



This cross section picture shows the table top (material 2), source holder (material 3), lead bricks (material 5), air (material 6) and the concrete floor (material 1). Materials are identified by the key given in the next figure.

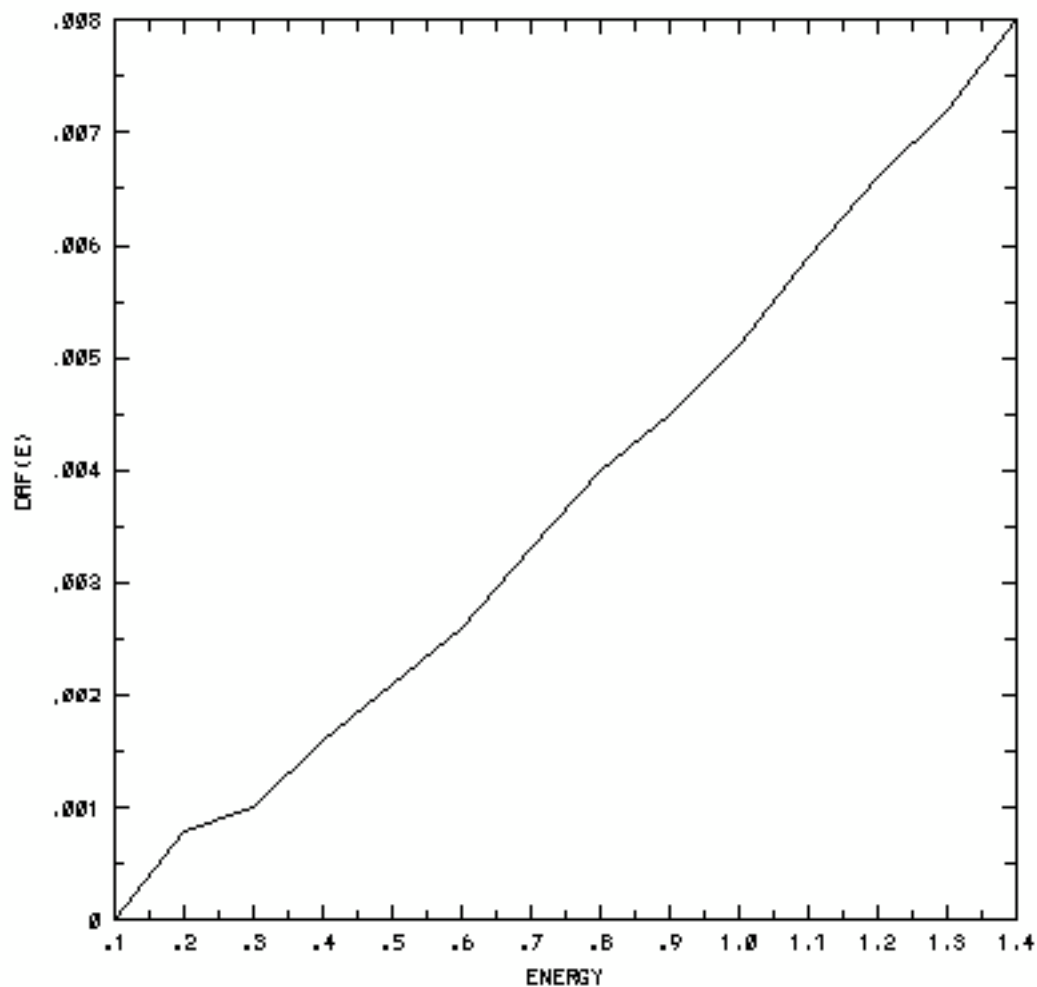
sample shielding problem

picture cs m -36. 0 12. -36. 0 -49. 36. 0 -49. 12:57:36 15-Aug-94
Sector/Media Key



This is the pattern-material key for the preceding cross sectional picture.

Energy Dependent Response Function for Detector Number 1
energy in units of mev

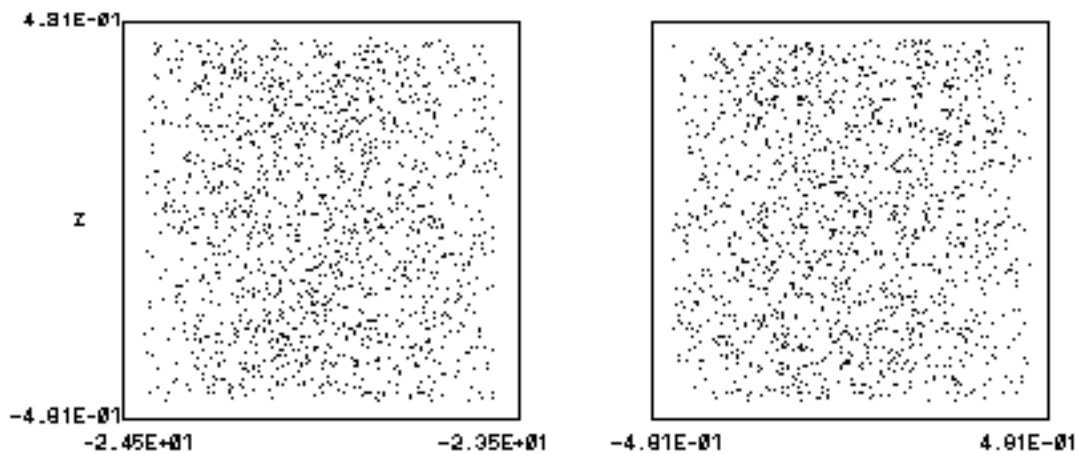
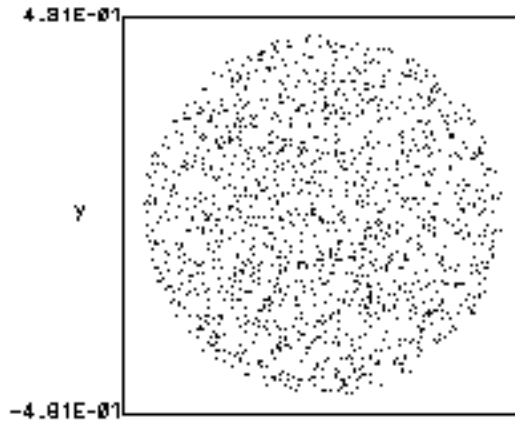


A graphical representation of the detector energy response function. Compare this to the graph from which the data was originally taken.

```

SOURCE--POSITION. Definition of description    1
  the unit of length is one inch
VOLUME of CYLINDER with center of base 1-2.4000E+01, .0000E+00,-4.3750E-01)
  center of top                          1-2.4000E+01, .0000E+00, 4.3750E-01)
  radius                                  4.3750E-01

```



location of 1500 points picked at random from this distribution

A representation of the source position dependence. This source is uniform within a cylindrical.

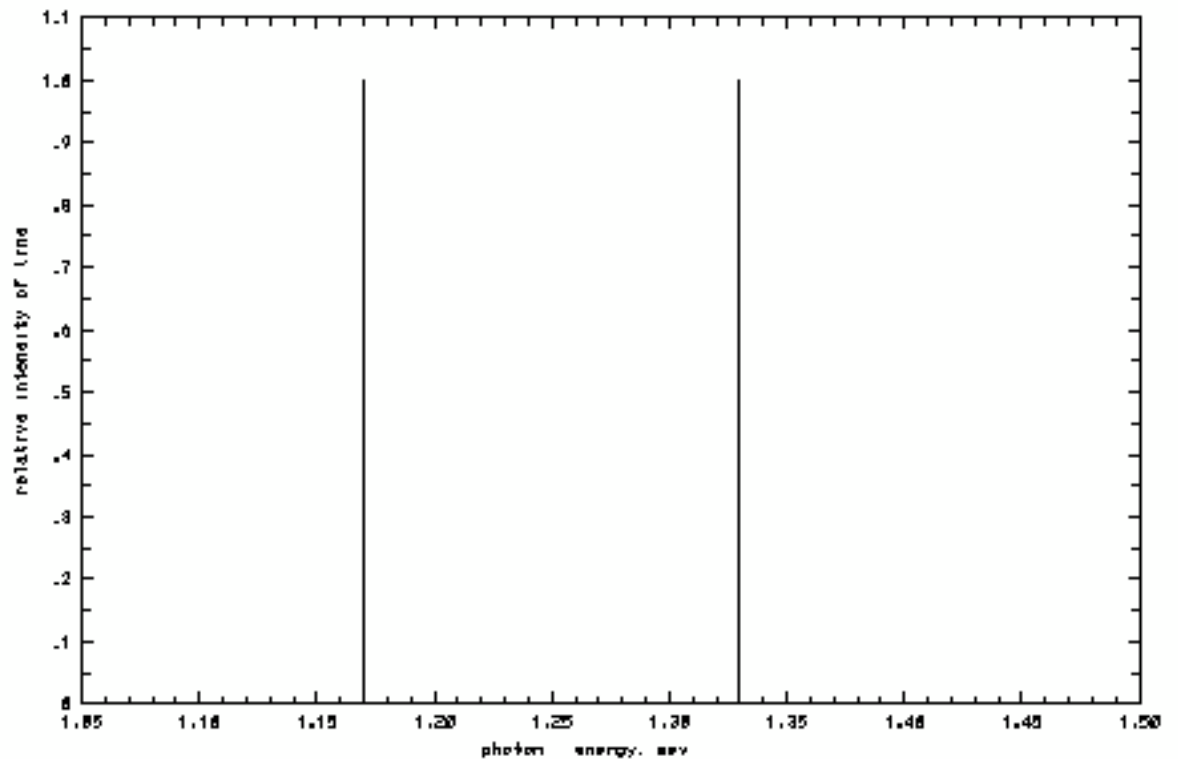
```

SOURCE--TIME. Definition of description    1
  the unit of time is one second
STEADY

```

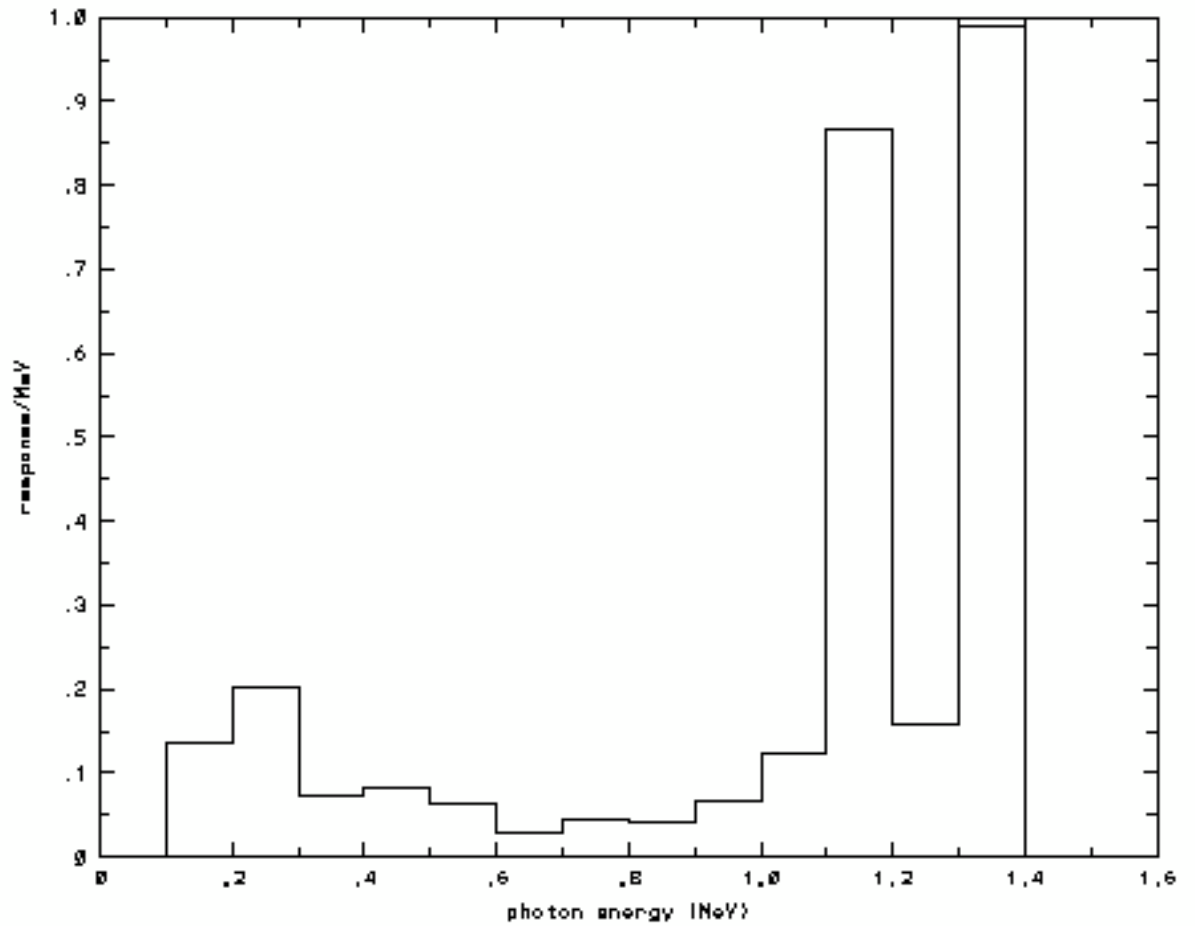
The time-dependence of this source is steady state.

```
SOURCE--ENERGY. Definition of description    1
  the unit of energy is mev
  photon    source
LINE
  energy    intensity
  1.170E+00  1.000E+00
  1.330E+00  1.000E+00
```



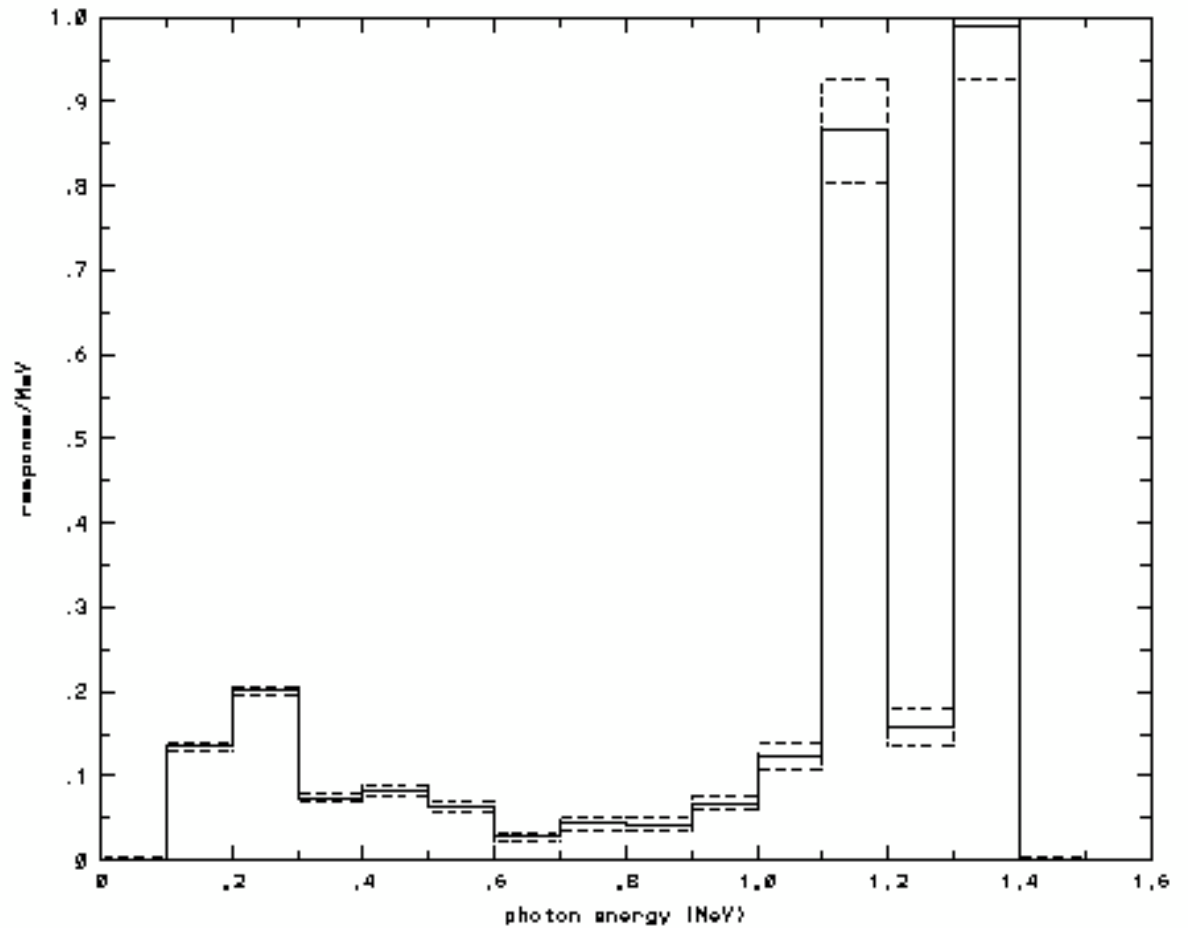
A representation of the source energy dependence. This source is a photon line source with equal intensity lines at 1.17 and 1.33 MeV.

DIFFERENTIAL RESULTS FOR DETECTOR NUMBER 1



Differential results showing the response/MeV vs. the photon energy in MeV for the given bin structure.

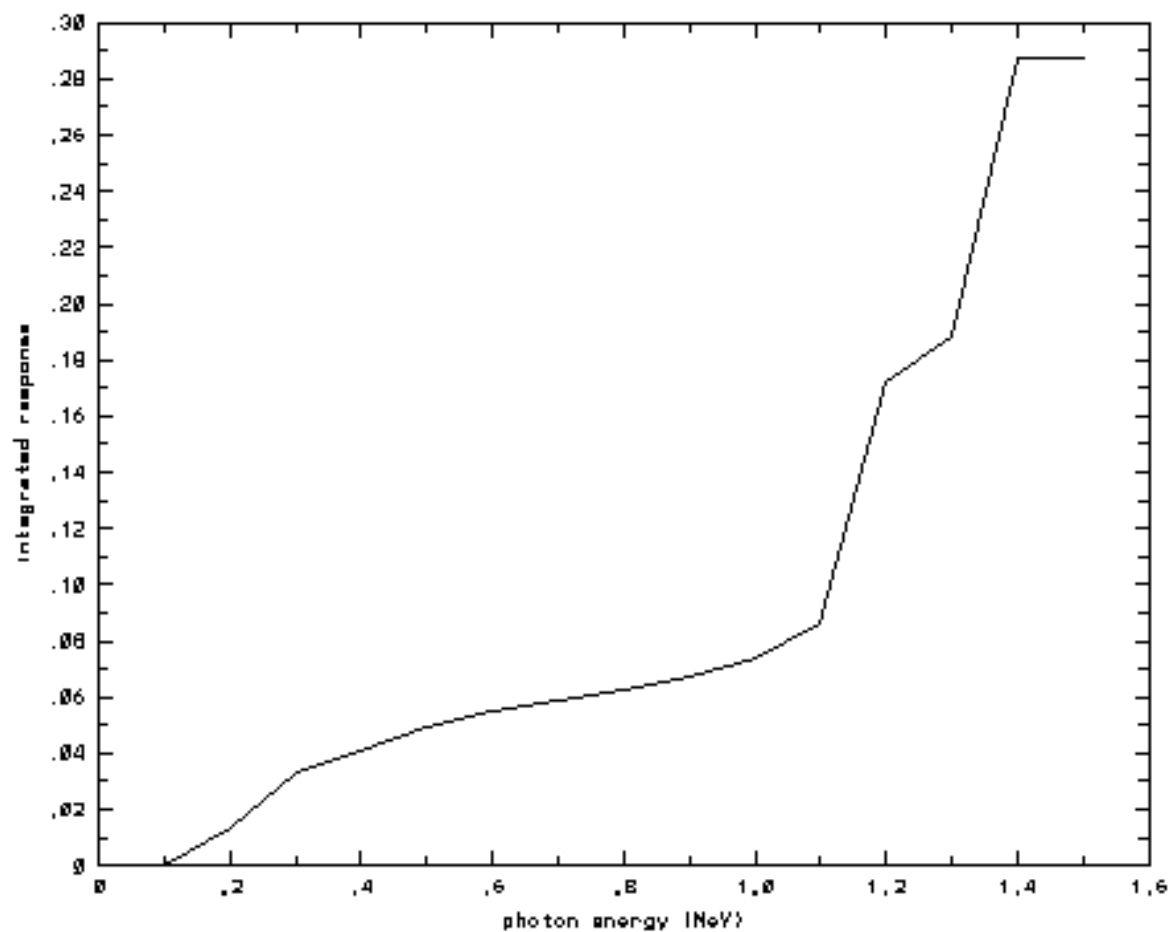
DIFFERENTIAL RESULTS FOR DETECTOR NUMBER 1



dotted lines indicate one standard deviation

Detector response as shown previously, with $\pm$ one standard deviation envelope.

DIFFERENTIAL RESULTS FOR DETECTOR NUMBER 1



INTEGRAL REPRESENTATION OF THE RESULTS

Running sum of the detector differential results.

20.2.2 The out file for Problem 1

- General information concerning code version and machine dependencies.

**** COG11.3 DESKTOP-LEE12 EGS5 -- Release 11.3 Oct 3 2022 ****

Please report all errors to the COG Developers:

Richard Buck

Edward Lent at COG@llnl.gov

COG Input file: shielding_problem

In Directory: /home/lee12/sample/Example_problems

COG -- LISTINGS OF INPUT LINES Wed Nov 2 13:52:57 2022

>> sample shielding problem

>> \$

>> basic

>> photon

>> in

>> mix

>> mat 1 cnrt2 2.51

>> mat 2 c 0.32 h 0.04 o 0.03 n 0.07 h2o 0.04 \$soak 12% moist

>> mat 3 al 2.7

>> mat 4 co 8.85

>> mat 5 pb 11.4

>> mat 6 air 0.00129

>>

>> geometry

>> sector 1 walls -1 +2

>> sector 2 table -3 or -4 or -5 or -6 or -7

>> sector 3 holder -8 or -9 +10

>> sector 4 cobalt -10

>> sector 5 lead -11

>> fill 6

>> boundary vacuum +1

>>

>> picture cs m color -36. 0. 12. -36. 0. -49. 36. 0. -49.

>>

```
>> surfaces
>> 1 box 384 264 144 tr 0 0 12      $ outside room walls
>> 2 box 360 240 120 tr 0 0 12      $ inside room walls
>> 3 box 60 36 3 tr 0 0 -13.5        $ table top
>> 4 box 4 4 33 tr -28 -16 -31.5 $ table leg
>> 5 box 4 4 33 tr -28 16 -31.5      $ table leg
>> 6 box 4 4 33 tr 28 16 -31.5      $ table leg
>> 7 box 4 4 33 tr 28 -16 -31.5      $ table leg
>> 8 cyl 0.125 0.5 12 tr -24.0 0.0 0.0 -24.0 0.0 -120.0 0 -1 0 $ rod
>> 9 cyl 0.5 -0.5 0.5 tr -24.0 0.0 0.0 -24.0 0.0 -120 0 -1 0 $ container
>> 10 cyl 0.4375 -0.4375 0.4375 tr -24.0 0.0 0.0 -24.0 0.0 -120.0 0 -1 0 $ src
>> 11 box 1.5 8 16 tr 0.0 0.0 -4.0 $ lead bricks
>>
>> detector
>> number 1
>> point 24. 0. 0.
>> drf-e photon $ gm tube sens for al cathode
>> 0.1 0. 0.2 8.0e-4 0.3 1.0e-3 0.4 1.6e-3
>> 0.5 2.1e-3 0.6 2.6e-3 0.7 3.3e-3 0.8 4.0e-3
>> 0.9 4.5e-3 1.0 5.1e-3 1.1 5.9e-3 1.2 6.6e-3
>> 1.3 7.2e-3 1.4 8.0e-3
>> bin e photon
>> 16 [0. i 1.5]
>>
>> source
>> npart 50000
>> define p 1
>> cylinder -24. 0. -0.4375 -24. 0. 0.4375 0.4375
>> define t 1
>> steady
>> define e 1
>> photon line 1.17 1. 1.33 1.
>> inc 7.4e7 e 1 p 1 t 1
>>
>> walk-collision
>> photon region all energy 0. to 0.1 0.
>>
>> End
```

- COG's interpretation of the BASIC Data Block—units, starting random number seed, particle types to be followed, etc.

```

BASIC PARAMETERS
Units of length = inch
Units of time = second
Units of energy = mev
Maximum running time = 1.0000E+08 minutes
Random number seeds = 26173 18852
Sequence number = 1
Particle types to be followed: photon

```

- COG's interpretation of the MIX Data Block. The total density for each material is shown so that input errors can be more easily detected.

MIXTURE DEFINITIONS

Material Number	Physical Temp	Component Name	Density (gm/cc)	Number Fraction	Weight Percent	atoms/ barn-cm
1	293.6°K	cncrt2	2.510E+00			
2	293.6°K	c	3.200E-01			
		h	4.000E-02			
		o	3.000E-02			
		n	7.000E-02			
		h2o	4.000E-02			
3	293.6°K	al	2.700E+00			
4	293.6°K	co	8.850E+00			
5	293.6°K	pb	1.140E+01			
6	293.6°K	air	1.290E-03			

- COG's interpretation of the SURFACE Data Block—the input "shorthand" is expanded. Check to make sure the surfaces are as intended.

```

GEOMETRICAL SURFACE SPECIFICATIONS:          unit length is one inch

Surface Number      1          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' direction  3.84000E+02  2.64000E+02  1.44000E+02
  T/R data
    center of (x',y',z') system  ( .00000E+00, .00000E+00, 1.20000E+01)

Surface Number      2          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  3.60000E+02  2.40000E+02  1.20000E+02
  T/R data
    center of (x',y',z') system  ( .00000E+00, .00000E+00, 1.20000E+01)

Surface Number      3          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  6.00000E+01  3.60000E+01  3.00000E+00
  T/R data
    center of (x',y',z') system  ( .00000E+00, .00000E+00, -1.35000E+01)

Surface Number      4          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  4.00000E+00  4.00000E+00  3.30000E+01
  T/R data
    center of (x',y',z') system  (-2.80000E+01, -1.60000E+01, -3.15000E+01)

Surface Number      5          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  4.00000E+00  4.00000E+00  3.30000E+01
  T/R data
    center of (x',y',z') system  (-2.80000E+01, 1.60000E+01, -3.15000E+01)

Surface Number      6          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  4.00000E+00  4.00000E+00  3.30000E+01
  T/R data
    center of (x',y',z') system  ( 2.80000E+01, 1.60000E+01, -3.15000E+01)

```

```
Surface Number      7          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  4.00000E+00  4.00000E+00  3.30000E+01
  T/R data
    center of (x',y',z') system  ( 2.80000E+01, -1.60000E+01, -3.15000E+01)

Surface Number      8          right circular cylinder
  Radius            1.25000E-01
  Surface bounded by planes at x'=  5.00000E-01 and x'=  1.20000E+01
  T/R data
    center of (x',y',z') system  (-2.40000E+01,  .00000E+00,  .00000E+00)
    point on x'-axis             (-2.40000E+01,  .00000E+00, -1.20000E+02)

Surface Number      9          right circular cylinder
  Radius            5.00000E-01
  Surface bounded by planes at x'= -5.00000E-01 and x'=  5.00000E-01
  T/R data
    center of (x',y',z') system  (-2.40000E+01,  .00000E+00,  .00000E+00)
    point on x'-axis             (-2.40000E+01,  .00000E+00, -1.20000E+02)

Surface Number     10          right circular cylinder
  Radius            4.37500E-01
  Surface bounded by planes at x'= -4.37500E-01 and x'=  4.37500E-01
  T/R data
    center of (x',y',z') system  (-2.40000E+01,  .00000E+00,  .00000E+00)
    point on x'-axis             (-2.40000E+01,  .00000E+00, -1.20000E+02)

Surface Number     11          box
  center of box is at x',y',z' coordinate center
  length in x', y', and z' directions  1.50000E+00  8.00000E+00  1.60000E+01
  T/R data
    center of (x',y',z') system  ( .00000E+00,  .00000E+00, -4.00000E+00)
```


- COG's interpretation of the GEOMETRY Data Block—essentially an echo of the input, with material, region, and density factors included. Note that we have requested a picture of the geometry. It is only during picture production that COG checks for certain geometry definition errors. *Pictures should be used extensively to "debug" the geometry.*

BASIC GEOMETRICAL DESCRIPTION

```

      No.   Name   Med   Reg   Df   --Surface relationships--
sector  1  walls    1     1  1.000E+00      -1      +2
sector  2  table    2     2  1.000E+00      -3
                                     or      -4
                                     or      -5
                                     or      -6
                                     or      -7
sector  3  holder    3     3  1.000E+00      -8
                                     or      -9      +10
sector  4  cobalt    4     4  1.000E+00     -10
sector  5  lead      5     5  1.000E+00     -11
boundary vacuum                                +1
fill     6                                6     6  1.000E+00

picture cs m -36. 0 12. -36. 0 -49. 36. 0 -49.

```

- COG's interpretation of the DETECTOR Data Block.

DETECTOR DESCRIPTIONS

```

DETECTOR NUMBER = 1

POINT

position - x = 2.400000E+01
          y =  .000000E+00
          z =  .000000E+00 (in units of a in)

```

energy dependent detector response function

photon

energy in units of mev

energy	DRF
1.0000E-01	.0000E+00
2.0000E-01	8.0000E-04
3.0000E-01	1.0000E-03
4.0000E-01	1.6000E-03
5.0000E-01	2.1000E-03
6.0000E-01	2.6000E-03
7.0000E-01	3.3000E-03
8.0000E-01	4.0000E-03
9.0000E-01	4.5000E-03
1.0000E+00	5.1000E-03
1.1000E+00	5.9000E-03
1.2000E+00	6.6000E-03
1.3000E+00	7.2000E-03
1.4000E+00	8.0000E-03

BIN definitions for differential output

energy-bin structure --- photon

energy in units of mev

.0000E+00	1.0000E-01	2.0000E-01	3.0000E-01	4.0000E-01
5.0000E-01	6.0000E-01	7.0000E-01	8.0000E-01	9.0000E-01
1.0000E+00	1.1000E+00	1.2000E+00	1.3000E+00	1.4000E+00
1.5000E+00				

- COG's interpretation of the SOURCE Data Block—a reiteration of the source definitions.

DEFINITION OF FIXED SOURCE

SOURCE--POSITION: Definition of description 1

the unit of length is one inch

VOLUME of CYLINDER with center of base(-2.4000E+01, .0000E+00,-4.3750E-01)

center of top (-2.4000E+01, .0000E+00, 4.3750E-01)

radius 4.3750E-01

```

SOURCE--TIME: Definition of description      1
    the unit of time is one second

STEADY

SOURCE--ENERGY: Definition of description    1
    the unit of energy is mev
    photon    source

LINE

                                energy      intensity
                                1.1700E+00    1.0000E+00
                                1.3300E+00    1.0000E+00

COMPLETE SOURCE DESCRIPTION

total number of source particles to be generated = 50000

increment    source    strength    description of separable parts    relative
position    energy    time    angle    importance
1          7.4000E+07    p/t          1          1          1          0    1.0000E+00

                                totals
                                7.4000E+07    p/t

```

• COG's interpretation of the WALK-COLLISION Data Block.

Modifications to the random WALK-SPLITTING/RUSSIAN ROULETTE at a COLLISION site

```

particle    ----regions----    -----energy-----    #-out / #-in
type
photon      all                .00E+00 to 1.00E-01    .0000E+00

```

• The names of the dictionary and all cross section files, plus the starting random number seeds and the starting sequence number.

```

dictionary file .....2/ 7/ 0 /usr/gapps/cog/Dec/data/COGLEX
Standard Dictionary
gamma cross section file .....7/ 7/98 /usr/gapps/cog/Dec/data/COGGXS

Starting Random Number Seeds for Transport Phase =    26173    18852
Starting Sequence Number =                                1

```

- The average source parameters—these should be compared to the desired source. The minimum, maximum, and average x, y, and z are consistent with the cylindrical volume position dependence; the minimum, maximum, and average u, v, and w are consistent with the (default) isotropic angle dependence; etc.

AVERAGE SOURCE PARAMETERS

	minimum	maximum	average	
photon				
x	-2.4437E+01	-2.3563E+01	-2.4001E+01	in
y	-4.3729E-01	4.3630E-01	3.3249E-04	in
z	-4.3750E-01	4.3748E-01	-1.9139E-04	in
u	-9.9997E-01	1.0000E+00	2.5009E-03	
v	-1.0000E+00	9.9998E-01	-2.6508E-03	
w	-9.9994E-01	9.9999E-01	-3.0175E-06	
t	.0000E+00	.0000E+00	.0000E+00	sec
E	1.1700E+00	1.3300E+00	1.2504E+00	MeV
wt	1.0000E+00	1.0000E+00	1.0000E+00	

- The Summary Table. The summary table contains, for each region, a list of events that occurred in the region, the particle types associated with each event, the number of such events, the removal (from the region) weight for each event, the addition (to the region) weight for each event, and the energy deposited (in the region) by each event. For example, in region 2, the oak table, there were 210 Russian roulette events, removing a weight of 3.1080E+05 and depositing 2.7831E+04 MeV; 17 Rayleigh events, with no net weight gain or loss and no energy deposited; etc. The table also shows that most of the deposited energy ($\sim 7.0 \times 10^7$ out of $\sim 9.3 \times 10^7$) ends up in the concrete walls, which should not be surprising since they subtend by far the largest solid angle. For shielding problems the weights and energy deposition are normalized to one source particle, i.e. the *Addition Weight* for the *Fixed source* event in the *Sum over all regions* should be equal to the (sum of) the source strength(s) on the SOURCE INCREMENTS(s).

The Energy Balance column tracks the change of energy in a particle population in a region due to the Events. The sums of all the gains and losses in a region (shown in the Totals rows) should be zero for a particle type. The lack of closure of this number indicates the accumulated round-off error resulting from summing over all particle events in the region.

COG Example Problem 1

10/30/2024

SUMMARY OF RANDOM-WALK EVENTS FOR 50000 SOURCE PARTICLES

Removal and Addition Weights and Deposited Energy are normalized
to one source particle

Region number 0

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
-----	----	-----	-----	-----	-----	-----

Region number 1 Region name walls

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Current in	pho	60296	0.0000E+00	8.9322E+07	0.0000E+00	7.4174E+07
Current out	pho	13766	2.0403E+07	0.0000E+00	0.0000E+00	-4.4959E+06
Leakage	pho	1402	2.0750E+06	0.0000E+00	0.0000E+00	-1.2582E+06
Russ roulette	pho	34842	5.1621E+07	0.0000E+00	4.3901E+06	-4.3901E+06
Rayleigh	pho	7979	1.1819E+07	1.1819E+07	0.0000E+00	0.0000E+00
Compton	pho	263963	3.9101E+08	3.9101E+08	6.1450E+07	-6.1450E+07
Photoelectric	pho	12073	1.7887E+07	2.6462E+06	2.5749E+06	-2.5749E+06
Pair prod	pho	12	1.7760E+04	3.5520E+04	4.9965E+03	-4.9965E+03
-----	----	-----	-----	-----	-----	-----
Totals	pho		4.9484E+08	4.9484E+08	6.8420E+07	1.2160E+00

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0009E+00

Region number 2 Region name table

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Current in	pho	10823	0.0000E+00	1.6028E+07	0.0000E+00	1.5872E+07
Current out	pho	10609	1.5712E+07	0.0000E+00	0.0000E+00	-1.3520E+07
Russ roulette	pho	210	3.1080E+05	0.0000E+00	2.7831E+04	-2.7831E+04
Rayleigh	pho	17	2.5160E+04	2.5160E+04	0.0000E+00	0.0000E+00
Compton	pho	5092	7.5436E+06	7.5436E+06	2.3231E+06	-2.3231E+06
Photoelectric	pho	4	5.9200E+03	0.0000E+00	7.3493E+02	-7.3493E+02
-----	----	-----	-----	-----	-----	-----
Totals	pho		2.3597E+07	2.3597E+07	2.3517E+06	-2.3280E-01

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0010E+00

Region number	3	Region name holder				
		# of	Weight		Energy (MeV)	
Event	Part	Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Current in	pho	48578	0.0000E+00	7.1952E+07	0.0000E+00	7.6167E+07
Current out	pho	48537	7.1892E+07	0.0000E+00	0.0000E+00	-7.4810E+07
Russ roulette	pho	32	4.7360E+04	0.0000E+00	4.3435E+03	-4.3435E+03
Rayleigh	pho	17	2.5160E+04	2.5160E+04	0.0000E+00	0.0000E+00
Compton	pho	2245	3.3270E+06	3.3270E+06	1.3497E+06	-1.3497E+06
Photoelectric	pho	10	1.4800E+04	1.4800E+03	2.1553E+03	-2.1553E+03
Pair prod	pho	1	1.4800E+03	2.9600E+03	4.5584E+02	-4.5584E+02
-----	----	-----	-----	-----	-----	-----
Totals	pho		7.5308E+07	7.5308E+07	1.3567E+06	3.3483E+00

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0013E+00

Region number	4	Region name cobalt				
		# of	Weight		Energy (MeV)	
Event	Part	Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Fixed source	pho	50000	0.0000E+00	7.4000E+07	0.0000E+00	9.2550E+07
Current in	pho	457	0.0000E+00	6.7636E+05	0.0000E+00	1.9436E+05
Current out	pho	48176	7.1354E+07	0.0000E+00	0.0000E+00	-7.5564E+07
Russ roulette	pho	1046	1.5525E+06	0.0000E+00	5.7420E+04	-5.7420E+04
Rayleigh	pho	747	1.1056E+06	1.1056E+06	0.0000E+00	0.0000E+00
Compton	pho	26713	3.9584E+07	3.9584E+07	1.6424E+07	-1.6424E+07
Photoelectric	pho	1909	2.8283E+06	9.9900E+05	6.8065E+05	-6.8065E+05
Pair prod	pho	40	5.9200E+04	1.1840E+05	1.7997E+04	-1.7997E+04
-----	----	-----	-----	-----	-----	-----
Totals	pho		1.1648E+08	1.1648E+08	1.7180E+07	3.5977E+00

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0012E+00

Region number 5 Region name lead

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Current in	pho	910	0.0000E+00	1.3468E+06	0.0000E+00	1.3528E+06
Current out	pho	151	2.2348E+05	0.0000E+00	0.0000E+00	-2.2387E+05
Russ roulette	pho	817	1.2136E+06	0.0000E+00	7.2155E+04	-7.2155E+04
Rayleigh	pho	88	1.3024E+05	1.3024E+05	0.0000E+00	0.0000E+00
Compton	pho	811	1.2077E+06	1.2077E+06	5.1134E+05	-5.1134E+05
Photoelectric	pho	759	1.1292E+06	1.2136E+06	5.4357E+05	-5.4357E+05
Pair prod	pho	4	5.9200E+03	1.1840E+04	1.8234E+03	-1.8234E+03
	pho					
-----	----	-----	-----	-----	-----	-----
Totals	pho		3.9102E+06	3.9102E+06	1.1289E+06	7.0643E-02

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0054E+00

Region number 6

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Current in	pho	72532	0.0000E+00	1.0745E+08	0.0000E+00	9.2764E+07
Current out	pho	72357	1.0719E+08	0.0000E+00	0.0000E+00	-9.1909E+07
Russ roulette	pho	169	2.5012E+05	0.0000E+00	2.2128E+04	-2.2128E+04
Rayleigh	pho	23	3.4040E+04	3.4040E+04	0.0000E+00	0.0000E+00
Compton	pho	2066	3.0621E+06	3.0621E+06	8.3151E+05	-8.3151E+05
Photoelectric	pho	6	8.8800E+03	0.0000E+00	1.0347E+03	-1.0347E+03
-----	----	-----	-----	-----	-----	-----
Totals	pho		1.1054E+08	1.1054E+08	8.5467E+05	-3.8396E+00

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0014E+00

Sum over all regions

Event	Part	# of Events	Weight		Energy (MeV)	
			Removal	Addition	Deposited	Balance
Fixed source	pho	50000	0.0000E+00	7.4000E+07	0.0000E+00	9.2550E+07
Leakage	pho	1402	2.0750E+06	0.0000E+00	0.0000E+00	-1.2582E+06
Russ roulette	pho	37116	5.4995E+07	0.0000E+00	4.5740E+06	-4.5740E+06
Rayleigh	pho	8871	1.3139E+07	1.3139E+07	0.0000E+00	0.0000E+00
Compton	pho	300890	4.4574E+08	4.4574E+08	8.2889E+07	-8.2889E+07
Photoelectric	pho	14761	2.1874E+07	4.8603E+06	3.8030E+06	-3.8030E+06
Pair prod	pho	57	8.4360E+04	1.6872E+05	2.5272E+04	-2.5272E+04
Totals	pho		5.3791E+08	5.3791E+08	9.1292E+07	4.5318E+00

Statistics on Pre-collision Weights

	Minimum	Maximum	Average
pho	1.0000E+00	2.0000E+00	1.0010E+00

- The detector results, showing the total response, standard deviation, fractional standard deviation and figure-of-merit.

RESULTS FOR DETECTOR 1

sample shielding problem

TYPE -- point

TOTAL RESPONSE PER SOURCE PARTICLE

(Integrated over all energies, times and angles and with all masks and response functions included. Units of response are those of the response function (if any) times particles/cm². fsd is the fractional standard deviation = std. dev./response.)

particle	response	std. dev.	fsd	fom
photon	2.8257E-01	5.5071E-03	1.9490E-02	3.6194E+03

- The detector results, showing the particle number and the % contribution for the 10 largest contributors to the total response. If you wished to rerun this problem to get better overall statistics, then it might be wise to first run a RETRACE on the top few particles in this scoring list to get information on how these particles came to contribute so much to the answer. If you find that these heavily-contributing particles all come along a certain source-to-detector path, you might wish to bias the problem to increase the number of particles which traverse this path, thus obtaining a statistically-improved answer.

SOURCE PARTICLES MAKING THE TEN LARGEST CONTRIBUTIONS TO TOTAL RESPONSE
event histories for these may be obtained by re-running with RETRACE
option

Source Particle	Score as	Cumulative Score as
RNG Sequence #	% of total result	% of total result
11366902	0.5932	0.5932
5794778	0.4718	1.0650
4092535	0.4587	1.5237
9883980	0.3602	1.8839
12567980	0.3504	2.2344
14055473	0.3504	2.5847
3432425	0.3261	2.9108
3915209	0.3255	3.2363
14114877	0.3137	3.5500
1069626	0.3022	3.8523

- The detector results, as a function of the number of scatterings since source generation. About half the result comes from uncollided (0 scatterings) particles. This part of the total response could have been calculated by hand, quite accurately, using the usual point-kernel techniques.

TOTAL RESPONSE AS A FUNCTION OF SCATTERING EVENTS SINCE SOURCE
GENERATION

scatterings	% total result	accumulative %
0	48.9836	48.9836
1	31.5610	80.5446
2	11.1687	91.7133
3	4.2358	95.9491
4	2.0812	98.0304
5	1.0564	99.0868
6	.4982	99.5850
7	.2672	99.8522
8	.0925	99.9447
9	.0327	99.9774
10	.0136	99.9910
11	.0043	99.9953
12	.0017	99.9970
13	.0022	99.9992
14	.0005	99.9997
15	.0003	100.0000
16	3.0809E-05	100.0000
17	8.1585E-08	100.0000
18	5.9071E-10	100.0000
19	3.0408E-10	100.0000

• The detector results, listed by the region in which the scatterings occurred. About 53% of the total response comes from region 4—the source region, while < 8% comes from scatterings in the concrete walls—region 1. Comparing regions 1 and 5 we see that we have spent ~75% (283902 collisions out of a total of 374828 collisions) of the computational effort to very accurately (fsd of ~1%) determine ~8% of the total response, while spending ~0.4% of the time to rather poorly (fsd of ~12%) determine ~26% of the total response. This is a problem where the LIMIT option for POINT detectors could be used to advantage. You would set up this option to limit the number of point flux estimates made from region 1, while making more estimates from region 5.

LISTED BY THE REGION IN WHICH SCATTERINGS OCCURRED

region	particles	source + collisions	%response	Fsd
4	50000	79800	53.072783	.003
5	50000	1621	26.310876	.124
2	50000	5243	10.455555	.048
1	50000	283902	7.862106	.009
6	50000	2023	1.966539	.176
3	50000	2239	.332142	.103

- The detector results, presented as differential results within the given bin structure.

DIFFERENTIAL RESULTS FOR DETECTOR NUMBER 1

bin specifications		response/MeV			response	integral
photon energy (MeV)		value	std.dev.	fsd		response
0.00E+00	1.00E-01	0.000E+00	0.00E+00	0.000	0.000E+00	0.000E+00
1.00E-01	2.00E-01	1.382E-01	3.56E-03	0.026	1.382E-02	1.382E-02
2.00E-01	3.00E-01	1.995E-01	4.63E-03	0.023	1.995E-02	3.376E-02
3.00E-01	4.00E-01	7.377E-02	4.19E-03	0.057	7.377E-03	4.114E-02
4.00E-01	5.00E-01	1.033E-01	1.78E-02	0.173	1.033E-02	5.147E-02
5.00E-01	6.00E-01	7.364E-02	7.51E-03	0.102	7.364E-03	5.883E-02
6.00E-01	7.00E-01	3.015E-02	5.30E-03	0.176	3.015E-03	6.185E-02
7.00E-01	8.00E-01	3.234E-02	4.63E-03	0.143	3.234E-03	6.508E-02
8.00E-01	9.00E-01	4.399E-02	6.19E-03	0.141	4.399E-03	6.948E-02
9.00E-01	1.00E+00	7.994E-02	1.18E-02	0.148	7.994E-03	7.747E-02
1.00E+00	1.10E+00	1.737E-01	2.01E-02	0.116	1.737E-02	9.484E-02
1.10E+00	1.20E+00	7.877E-01	2.67E-02	0.034	7.877E-02	1.736E-01
1.20E+00	1.30E+00	1.500E-01	2.20E-02	0.146	1.500E-02	1.886E-01
1.30E+00	1.40E+00	9.396E-01	1.95E-02	0.021	9.396E-02	2.826E-01
1.40E+00	1.50E+00	0.000E+00	0.00E+00	0.000	0.000E+00	2.826E-01

Some restart, timing and miscellaneous memory information.

To continue this calculation, restart with RNG Sequence Number = 15421771
for Starting Seeds 26173 18852

sample shielding problem #1

COG Problem Began at: Wed Jul 17 14:45:26 2002

COG Problem Completed at: Wed Jul 17 14:46:19 2002

Total Problem CPU Time Used (hh:mm:ss) = 0:00:43

*****Memory Manager Report*****

Number of System memory blocks allocated= 4

Total number of Megabytes allocated= 4.0

Min. block size = 1.0 Max. block size= 1.0 Megabytes

21 COG Example Problem 2

21.1 Problem 2 Description and Input

This is a very simple criticality problem that illustrates COG's input and output for a criticality calculation. The problem is to determine k_{eff} for a bare ball of plutonium; the ball has a radius of 6.3849 cm.

COG input file for Example Problem 2 ...

```

PU-MET-FAST-001:  JEZEBEL  (17.020  kg  Pu(95.48)-1.02Ga  @ 15.61

g/cc)  BASIC
      neutron  delayedn

SURFACES
      1 sphere 6.3849 $ per Section 3.2

GEOMETRY
      sector 1 alloy -
      1 boundary vacuum
      1
      picture cs material -7 0 7 -7 0 -7 7 0 -7
      volume -7 -7 -7 7 -7 -7 -7 7 -7 14 14 14

CRITICALITY
      npart=5000 nbatch=5005 sdt=0.0001 nfirst=6
      norm=1. nsource=1 0. 0. 0.

MIX
      nlib=ENDFB6R7 $ Atom Densities per Table 3
      mat=1 bunches ga 1.3752-3 pu239 3.7047-2 pu240 1.7512-3 pu241 1.1674-4

END

```

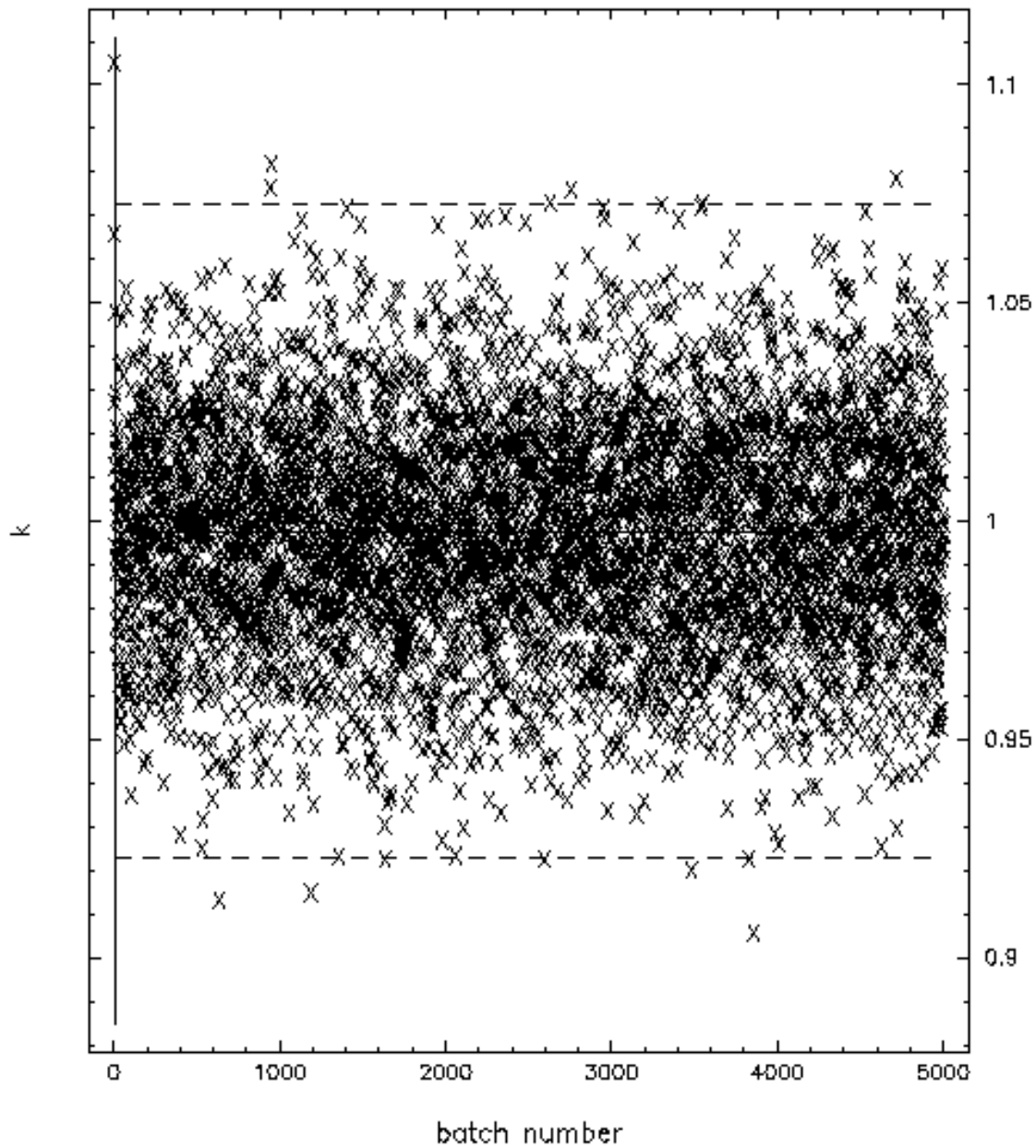
A brief explanation of the input ...

- The first line is the problem title.
- BASIC Data Block: Neutron only problem. Units default to cm, MeV, and seconds. “Delayedn” must be specified to get delayed fission neutrons created.
- SURFACE Data Block: One sphere
- GEOMETRY Data Block: Sector 1 is inside surface 1, i.e. a spherical volume with radius 6.3849 cm. There is a vacuum boundary outside surface 1, i.e. particles that get the sphere are no longer tracked.

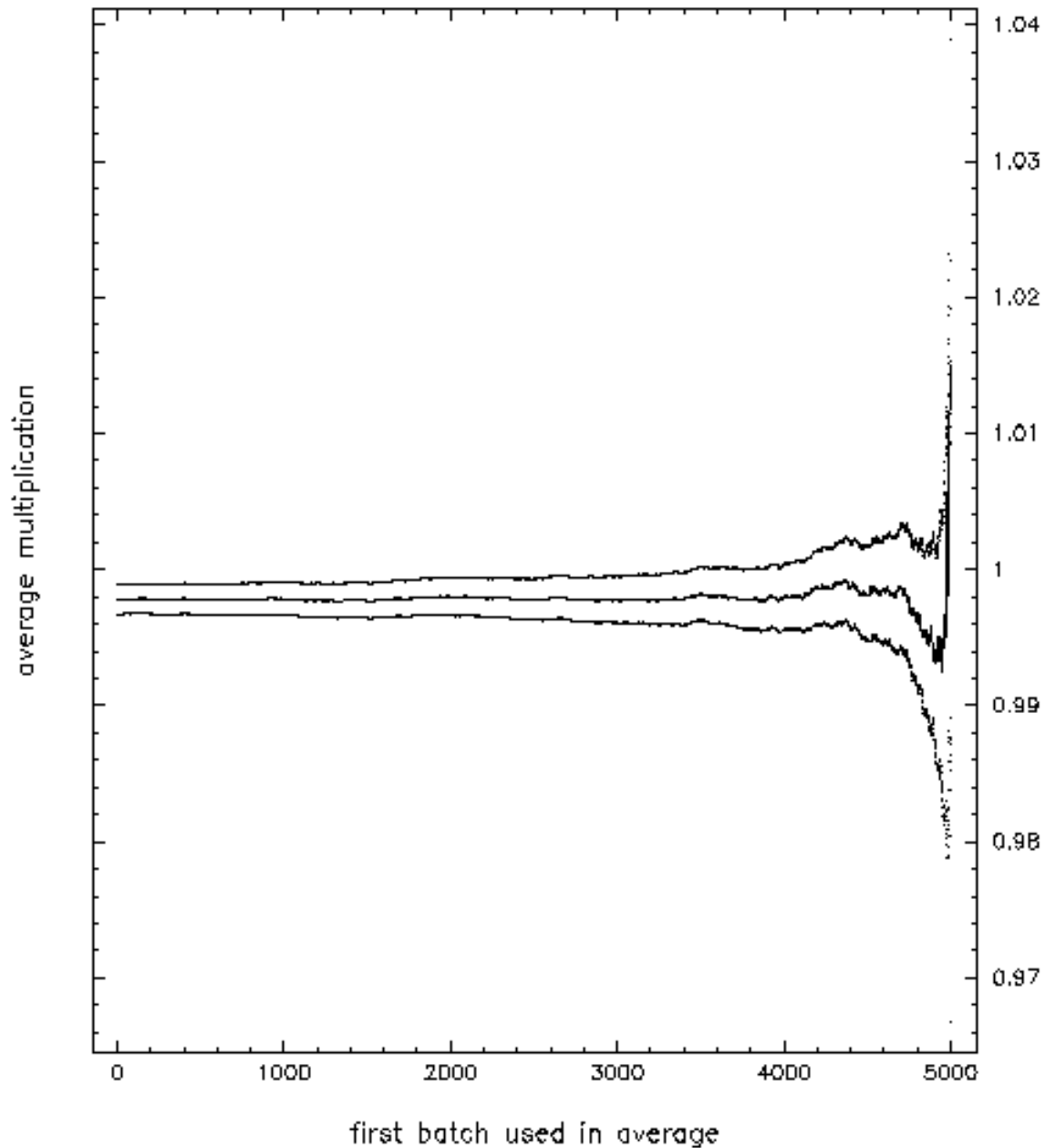
- MIX Data Block: Material 1 is a mixture Pu and Ga. Composition is specified by atom densities (bunches = atoms/barn-cm).
- CRITICALITY Data Block: Each batch will consist of ~5000 particles (npart 5000). Convergence occurs when 10 consecutive batches have a standard deviation of the mean less than 0.0001 (sdt 0.0001), or when nbatch = 5005 batches have been run. The first 5 batches are not averaged into the means (nfirst 6). Finally, the initial source distribution consists of a single point at the origin (nsourc 1 0. 0. 0.).
- END Data Block: End of input.

21.2 Problem 2 Results

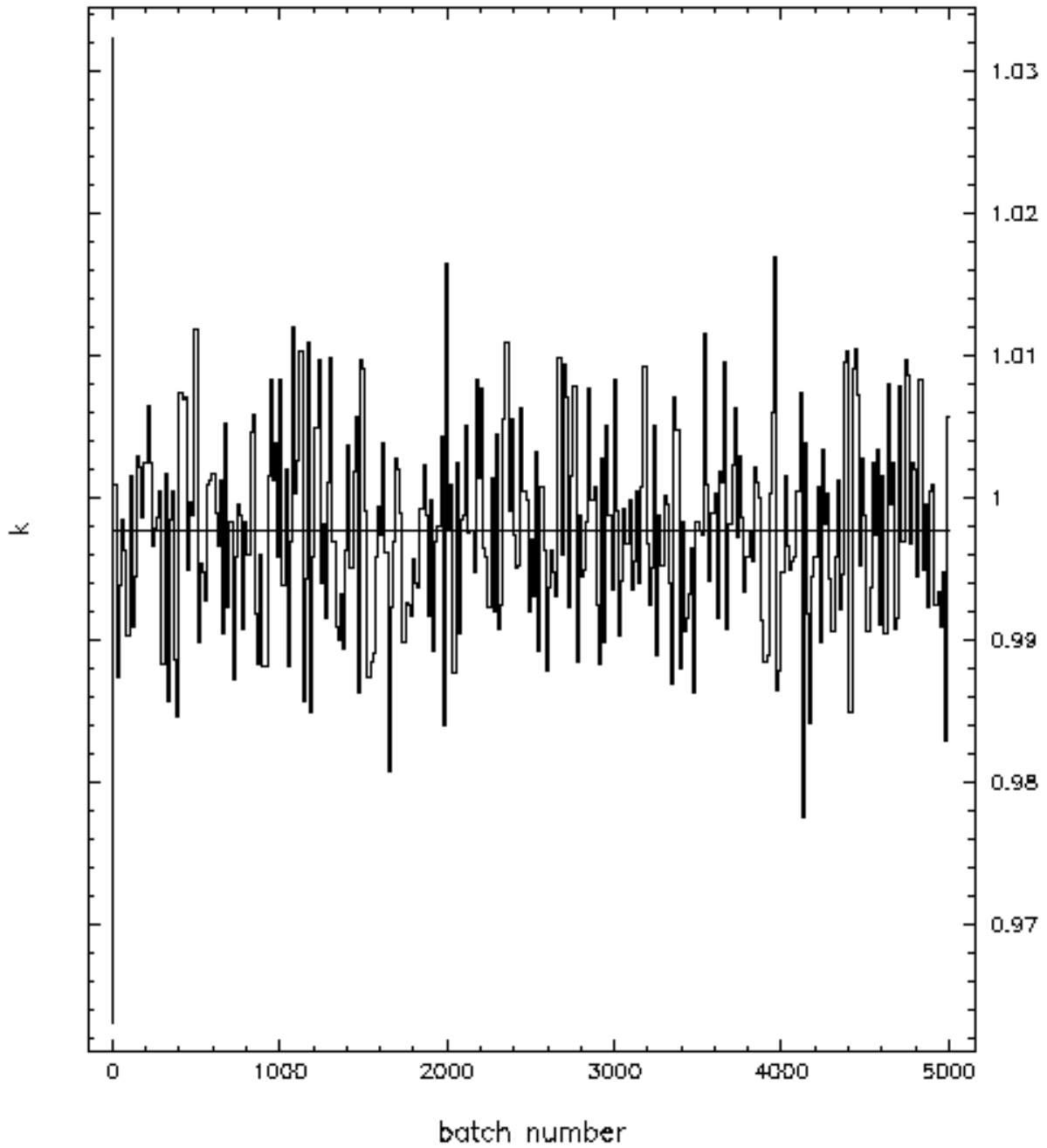
21.2.1 The....ps file for Problem 2



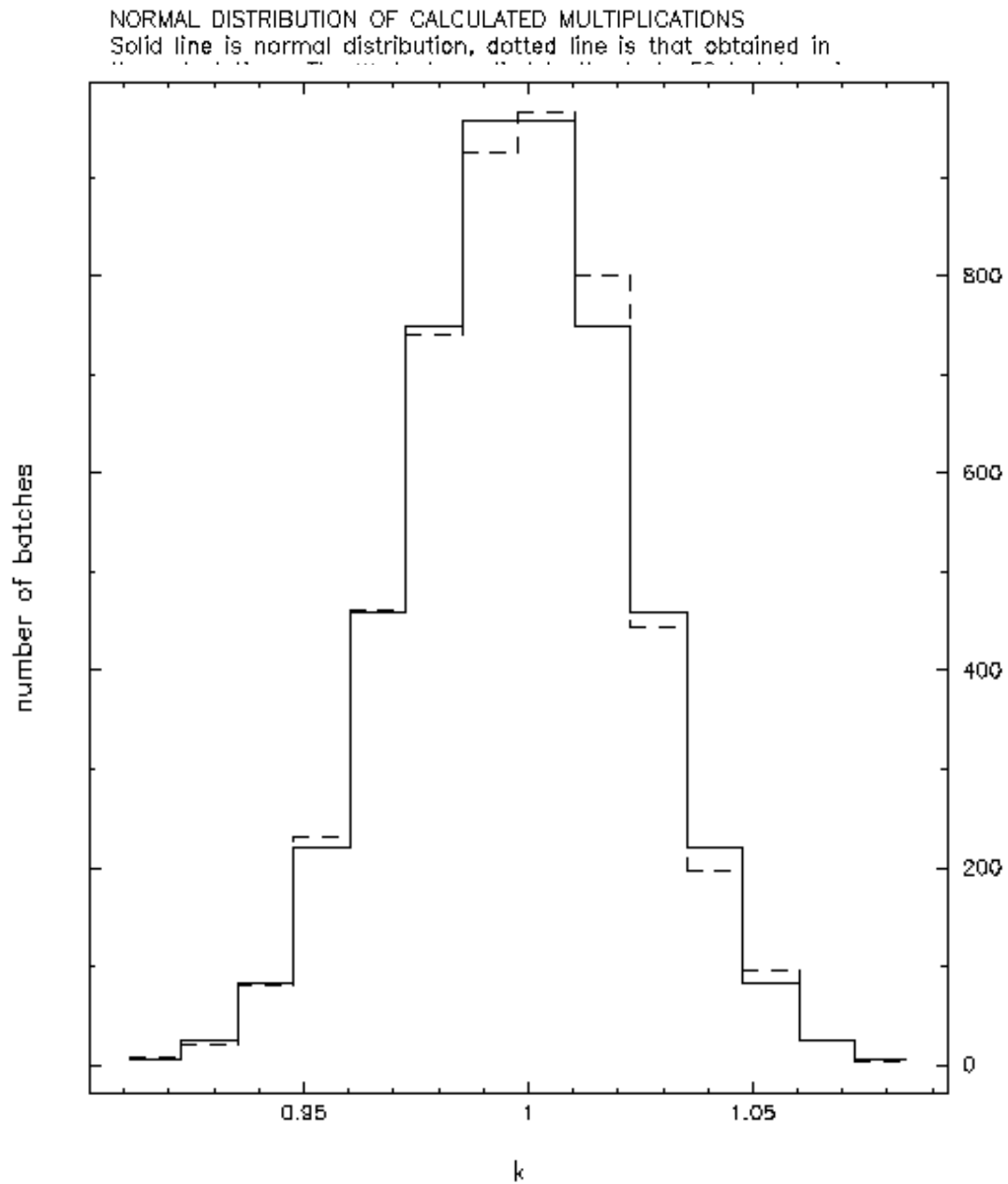
dotted lines are the three-sigma limits for the distribution of k -values.



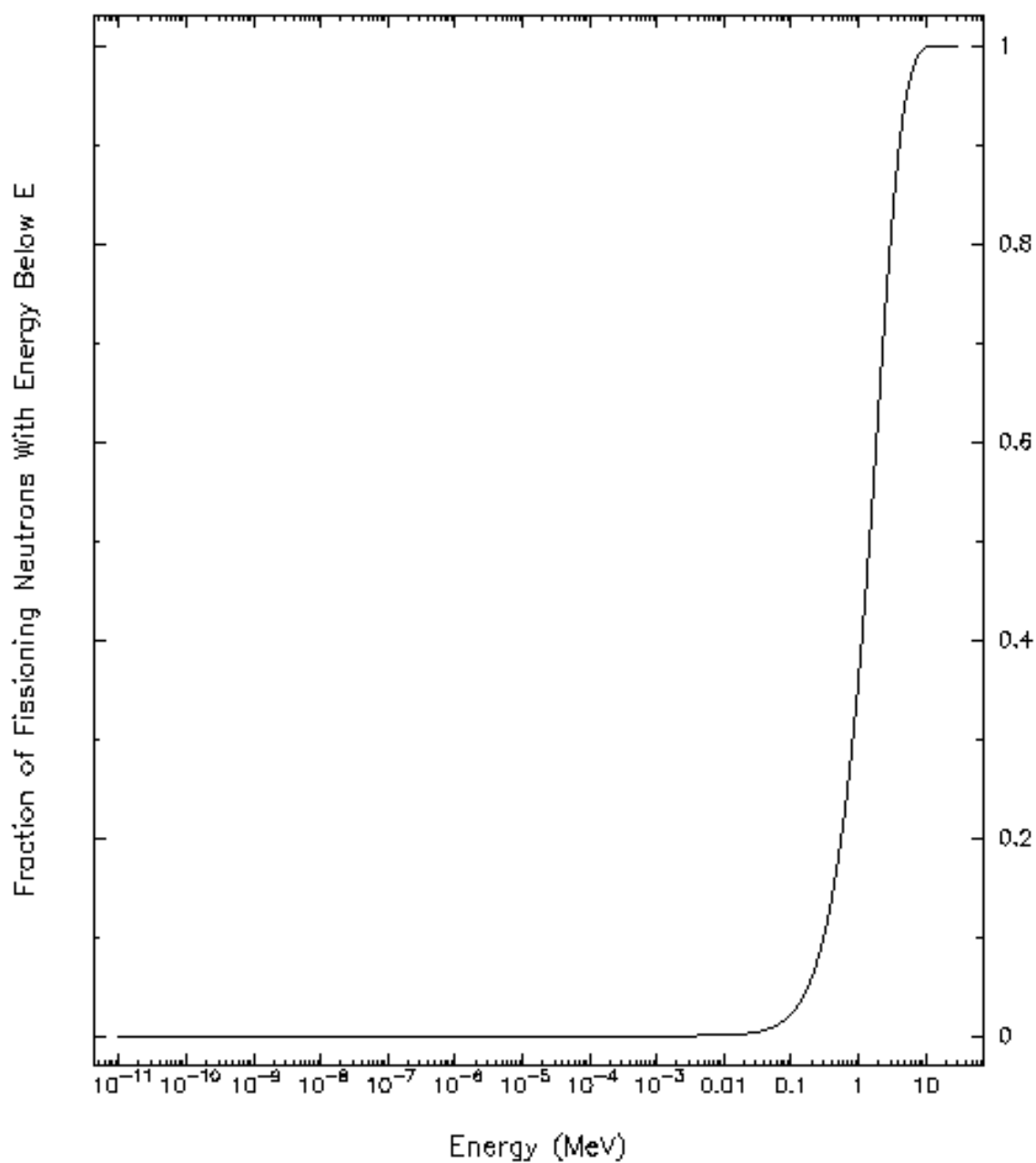
This is a plot of the average neutron multiplication, where the average is taken from the last batch towards the first batch. That is, the value at batch 2000 is the average of the batch results from batch 2000 through the last batch.



The plotted values correspond to the average of the last 15 batches, the average of the next to last 15 batches, etc. This plot is intended to show trends in the neutron multiplication.



This plot shows the actual distribution of k's compared to a normal distribution (solid line). The plot also indicates the results of a W-test (a test of normality) applied to the "settled" batches.



A plot of the "Fraction of Fissioning Neutrons With Energy Below E vs. Energy".

21.2.2 The out file for Problem 2

- General information concerning code version and machine dependencies.

```
** COG11.3          sora.llnl.gov          EGS5 -- Release  11.3   Sep 19 2022  **
```

Please report all errors to the COG Developers:

Edward Lent at COG@llnl.gov

COG Input file: example2

In Directory: /home/lent1/COG11.3-22.09.18/LOON

- Echo of the input file.

```
COG  -- LISTINGS OF INPUT LINES      Thu Sep 22      10:38:57 2022
>>PU-MET-FAST-001:  JEZEBEL (17.020 kg  Pu(95.48)-1.02Ga @ 15.61 g/cc )
>>
>>BASIC
>>neutrongdelayedn
>>
>>SURFACES
>>1 sphere 6.3849 $per Section 3.2
>>
>>GEOMETRY
>>sector 1 alloy -1
>>boundary vacuum 1
>>picture cs material -7  0  7  -7  0 -7      7  0 -7
>>volume          -7 -7 -7   7 -7 -7   -7  7 -7   14  14  14
>>
>>CRITICALITY
>>npart=5000
>>nbatch=5005
>>nfirst=6
>>sdt=0.0001
>>norm=1.
>>nsource=1   0.0.0.
>>
>>MIX
>>nlib=ENDFB6R7      $ Atom Densities per Table 3
>>mat=1 bunches ga 1.3752-3 pu239 3.7047-2 pu240 1.7512-3 pu241 1.1674-4
>>
>>END
```

- COG's interpretation of the BASIC Data Block—units, starting random number seed, particle types to be followed, etc.

BASIC PARAMETERS

```

Units of length = centimeter
Units of time = second
Units of energy = mev
Maximum running time = 1.0000E+08 minutes
Random number seeds = 27906 3945
Sequence number = 1
Particle types to be followed: neutron

Electron-positron pair-production will result in one annihilation
gammas produced and transported

Secondary neutron production is in the analog mode,
i.e. multiple neutrons of unit weight, rather than a single
neutron with non-unit weight, are produced for (n,2n), etc.

Delayed neutrons are being followed

Restart dumps will be taken every 60. minutes.
```

- COG's interpretation of the MIX Data Block. The total density for each material is shown so that input errors can be more easily detected.

MIXTURE DEFINITIONS

```
Definition of cross-section file: nlib = ENDFB6R7
```

Material Number	Physical Temp	Component Name	Density (gm/cc)	Number Fraction	Weight Percent	atoms/ barn-cm
1	293.6°K	ga				1.375E-03
		pu239				3.705E-02
		pu240				1.751E-03
		pu241				1.167E-04

- COG's interpretation of the SURFACE Data Block—the input "shorthand" is expanded. Check to make sure the surfaces are as intended.

```

GEOMETRICAL SURFACE SPECIFICATIONS:           unit length is one centimeter
Surface Number      1           sphere
Radius              6.38490E+00
  
```

- COG's interpretation of the GEOMETRY Data Block—essentially an echo of the input, with material, region, and density factors included.

```

BASIC GEOMETRICAL DESCRIPTION
      No.   Name   Mat   Reg   Df   --Surface relationships--
sector     1   alloy   1     1   1.000E+00   -1
boundary           vacuum                   +1
  
```

- COG's interpretation of the CRITICALITY Data Block—a reiteration of the criticality parameters.

```

CRITICALITY CALCULATION PARAMETERS
Number of particles per batch =      5000
Maximum number of batches =      5005
Terminating standard deviation =    1.00E-04
First batch used to obtain fluxes =    6
Number of fissions per second =    1.0000E+00
Position of initial point sources
      .0000E+00   .0000E+00   .0000E+00
  
```

- The names of the dictionary and all cross section files, plus the starting random number seeds.

```

dictionary file ..... 7/27/2022 COGLEX11.2
Standard Dictionary (COGLEX11.2 - for COG versions 11.2 and up)

neutron cross section file . 10/17/2012 /opt/apps/cog/COG11/datax11/ENDFB6R7
ENDF/B-VI Release 7

Starting Random Number Seeds for Transport Phase =    7406    8922
Starting Sequence Number =    1
  
```

- Results showing the individual batch k's and various average k's, where the averages are from the last batch towards the first batch. Repeated, for emphasis, are the average multiplication and its standard deviation.

PU-MET-FAST-001: JEZEBEL (17.020 kg Pu(95.48)-1.02Ga @ 15.61 g/cc)

K CALCULATION RESULTS

(Averaged from the last batch towards the first batch.)

Batch	Batch k	Average to this batch		AverageAverage of last 5		of last 10	
-----	-----	-----	-----	-----	-----	-----	-----
5005	0.97654	0.97654	0.97654				
5004	0.97974	0.97814	0.00160				
5003	0.99556	0.98395	0.00588				
5002	1.00464	0.98912	0.00664				
5001	1.00169	0.99163	0.00572	0.99163	0.00572		
5000	0.98735	0.99092	0.00473	0.99379	0.00459		
4999	0.98809	0.99051	0.00401	0.99546	0.00349		
4998	1.01513	0.99359	0.00464	0.99938	0.00526		
4997	1.00898	0.99530	0.00444	1.00025	0.00554		
4996	1.00567	0.99634	0.00410	1.00104	0.00565	0.99634	0.00410
4995	0.97456	0.99436	0.00421	0.99848	0.00748	0.99614	0.00421
...							
11	1.03674	0.99796	0.00033	1.00760	0.00774	1.00383	0.00761
10	1.04137	0.99797	0.00033	1.01403	0.01031	1.00931	0.00818
9	0.98609	0.99797	0.00033	1.01090	0.01164	1.00685	0.00850
8	1.00262	0.99797	0.00033	1.01226	0.01127	1.00353	0.00787
7	1.00258	0.99797	0.00033	1.01388	0.01073	1.00839	0.00565
6	0.98501	0.99797	0.00033	1.00353	0.01020	1.00557	0.00607
5	1.00569						
4	1.05493						
3	1.03769						
2	1.14190						
1	1.31764						

Average Multiplication = 0.9979670
Standard Deviation = 0.0003250

- A table showing the fraction of fissioning neutrons with energy below a given energy, i.e. effectively, the integrated spectrum of fissioning neutrons. This table is in MeV, regardless of the energy units set in the BASIC Data Block.

FRACTION OF FISSION, ABSORPTION, LEAKAGE, AND PRODUCTION NEUTRONS WITH ENERGY < E
 Median Energy = 1.4742E+00 1.4280E+00 1.4227E+00 1.5844E+00

Energy (MeV)	Fission	Fraction Below			Production
		Absorption	Leakage		
1.0000E-11	0.00000	0.00000	0.00000		0.00000
2.3259E-04	0.00000	0.00000	0.00000		0.00000
2.6871E-04	0.00000	0.00001	0.00000		0.00000
3.1045E-04	0.00001	0.00001	0.00000		0.00001
3.5867E-04	0.00001	0.00001	0.00000		0.00001
...					
4.0750E+00	0.89848	0.89860	0.89461		0.87857
4.6996E+00	0.93132	0.93029	0.92980		0.91500
5.4198E+00	0.95546	0.95357	0.95659		0.94255
6.2505E+00	0.97262	0.97026	0.97548		0.96289
7.2084E+00	0.98528	0.98350	0.98748		0.97907
8.3132E+00	0.99324	0.99228	0.99426		0.98995
9.5873E+00	0.99732	0.99688	0.99770		0.99582
1.1057E+01	0.99908	0.99893	0.99920		0.99851
1.2751E+01	0.99974	0.99970	0.99977		0.99956
1.4706E+01	0.99995	0.99994	0.99995		0.99990
1.6959E+01	0.99999	0.99999	0.99999		0.99998
1.9559E+01	1.00000	1.00000	1.00000		1.00000
3.0000E+01	1.00000	1.00000	1.00000		1.00000

- Various criticality output parameters. Mean time to fission—time from neutron birth from fission to capture producing fission. Mean time to absorption—time from neutron birth from fission to neutron absorption. Mean time to escape—time from neutron birth from fission to neutron escape. Optical path by region—average distance between collisions in the various regions. These times are in seconds and the optical path is in cm, regardless of the units set in the BASIC Data Block.

OPTICAL PATH (cm) BY REGION

 3.1940E+00 ± 3.8695E-04 1

TOTALS FOR THE LAST 5000 BATCHES NORMALIZED TO 1 FISSION

Event	Totals	# of Events
Production	3.164136E+00 ± 9.697764E-05	7922870
Removal	3.170567E+00 ± 2.148293E-05	25028503
Capture	3.441593E-02 ± 3.229432E-06	271343
Leakage	2.132539E+00 ± 1.762580E-05	16834290
Fission	1.000000E+00 ± 1.184937E-05	7894317

RATES AND MEAN-TIMES FOR THE LAST 5000 BATCHES

Event	Rates (1/sec)	Mean-Times (sec)
Production	3.264993E+08 ± 1.992534E+05	3.062794E-09 ± 1.869137E-12
Removal	3.271629E+08 ± 1.996584E+05	3.056581E-09 ± 1.865346E-12
Capture	3.551294E+06 ± 2.167255E+03	2.815875E-07 ± 1.718450E-10
Leakage	2.200513E+08 ± 1.342911E+05	4.544395E-09 ± 2.773317E-12
Fission	1.031875E+08 ± 6.297244E+04	9.691097E-09 ± 5.914205E-12

k = ProductionRate/RemovalRate= 0.9979717 ± 0.0008613

Effective delayed neutron fraction, beta-eff...

as fissions by delayed neutrons/all fissions = 0.00201 ± 0.00002

as fission neutrons by delayed neutrons/all fission neutrons = 0.00187 ± 0.00001

as 1 - kp/k = -.00018 ± 0.00122

alpha(gen/microsec) = 1.e-6*betaeff/RemovalTime= 6.13056E-01 ± 2.86401E-03

- The Summary Table. The summary table contains, for each region, a list of events that occurred in the region, the particle types associated with each event, the number of such events, the removal (from the region) weight for each event, the addition (to the region) weight for each event, and the energy deposited (in the region) by each event. For example, in the uranium ball the deposited energy due to fissions is 181.43 MeV, the (averaged over energy) $\bar{\nu}$ is 2.60 (Addition Weight), etc. For criticality problems the weights and energy deposition are normalized to one fission event, i.e. the *Removal Weight* for the *Fission* event in the *Sum over all regions* should be unity.

SUMMARY OF RANDOM-WALK EVENTS FOR THE LAST 5000 BATCHES

The Weights and Energies are normalized to one fission event,
(Neutrons from fission are missing a factor of k).

Region number 0

Event	Part	# of Events	Weight		Energy (MeV)	
			Removal	Addition	Deposited	Balance
-----	-----	-----	-----	-----	-----	-----

Region number 1		Region name alloy				
Event	Part	# of Events	Weight		Energy (MeV)	
			Removal	Addition	Deposited	Balance
-----	----	-----	-----	-----	-----	-----
Leakage	neut	16834290	2.13254E+00	0.00000E+00	0.00000E+00	-4.05495E+00
(n,n)	neut	22218427	2.81573E+00	2.81573E+00	1.10301E-02	-1.10301E-02
(n,n'g)	neut	6920023	8.76479E-01	8.76479E-01	4.96548E-01	-7.86697E-01
(n,2ng)	neut	28513	3.60791E-03	7.21583E-03	3.41690E-03	-2.38650E-02
(n,3ng)	neut	40	5.06377E-06	1.51913E-05	5.96988E-06	-6.97839E-05
(n,fission)	neut	7894317	1.00000E+00	3.16334E+00	2.01828E+02	4.89324E+00
(n,pg)	neut	22	2.78151E-06	0.00000E+00	2.17635E-05	-2.20597E-05
(n,ag)	neut	257	3.25215E-05	0.00000E+00	2.56219E-04	-1.72314E-04
(n,g)	neut	271064	3.43806E-02	0.00000E+00	2.38027E-01	-1.64339E-02
-----	----	-----	-----	-----	-----	-----
Totals	neut		6.86278E+00	6.86278E+00	2.02578E+02	-5.23019E-13

Statistics on Pre-collision Weights			
	Minimum	Maximum	Average
neut	9.90688E-01	3.01205E+00	1.00136E+00

Sum over all regions

Event	Part	# of	Weight		Energy (MeV)	
		Events	Removal	Addition	Deposited	Balance
-----		-----	-----	-----	-----	-----
Leakage	neut	16834290	2.13254E+00	0.00000E+00	0.00000E+00	-4.05495E+00
(n,n)	neut	22218427	2.81573E+00	2.81573E+00	1.10301E-02	-1.10301E-02
(n,n'g)	neut	6920023	8.76479E-01	8.76479E-01	4.96548E-01	-7.86697E-01
(n,2ng)	neut	28513	3.60791E-03	7.21583E-03	3.41690E-03	-2.38650E-02
(n,3ng)	neut	40	5.06377E-06	1.51913E-05	5.96988E-06	-6.97839E-05
(n,fission)	neut	7894317	1.00000E+00	3.16334E+00	2.01828E+02	4.89324E+00
(n,pg)	neut	22	2.78151E-06	0.00000E+00	2.17635E-05	-2.20597E-05
(n,ag)	neut	257	3.25215E-05	0.00000E+00	2.56219E-04	-1.72314E-04
(n,g)	neut	271064	3.43806E-02	0.00000E+00	2.38027E-01	-1.64339E-02
-----		-----	-----	-----	-----	-----
Totals	neut		6.86278E+00	6.86278E+00	2.02578E+02	-5.23019E-13

Statistics on Pre-collision Weights			
	Minimum	Maximum	Average
neut	9.90688E-01	3.01205E+00	1.00136E+00

•Some restart and timing information.

To continue this calculation...

```

restart with RNG Sequence Number = 253202261
for Starting Seeds 7406 8922

```

```

PU-MET-FAST-001: JEZEBEL (17.020 kg Pu(95.48)-1.02Ga @ 15.61 g/cc)
COG Problem Began at: Thu Sep 22 10:38:57 2022
COG Problem Completed at: Thu Sep 22 10:49:02 2022

```

```

Total Problem CPU Time Used (hh:mm:ss) = 0:09:56

```

22 COG COGLEX11.2 Dictionary Listing

The following pages list the materials available in the standard COG dictionary, COGLEX. COGLEX lists the elements, isotopes, compounds, and mixtures to be found in ALL of the cross section databases which COG uses. The user should carefully note that not every COGLEX entry can be found in a particular database. For example, the ENDFB6R7 neutron database contains the oxygen isotopes 8016 and 8017, while ENDL90 contain just 8016. None of the neutron libraries contains Ge, although it is available on the EPDL photon library. Requesting a material entry not in the specified database causes COG to issue an error message and stop. We plan to make available database query tools which will allow the user to review the contents of a given database.

Some databases have entries for “natural” elements, such as C (6000). Other databases may not, but may have the isotopic constituents (6012 and 6013) and COG will mix up cross sections as directed by a database of natural abundances. If neither of these cases obtains for a specified natural element, COG halts with an error message.

The listing consists, for each material, of the material name (and aliases), the density (in gm/cc), and pairs of (isotope, weight fraction) for the isotopes composing the material. Material names can contain at most 8 characters, and consist of the chemical symbol (Li, Li6, etc.), the full name (Lithium, Lithium6, etc., where possible), Z or 1000Z (3, 3000, etc.) for naturally occurring abundances, or 1000Z + A (3006, etc.) for specific isotopes. The code input is case insensitive, so Li, li, lI, and LI are all equivalent. The material definition is given in terms of isotope, weight fraction pairs, i.e. naturally occurring Lithium is defined to be 6.4% Lithium6 and 93.6% Lithium7.

The entries are self-explanatory with the following

exceptions. FF or 99125 represents a composite of fission fragments.

The materials which are all numbers (e.g., 1801, 4809) are special S(a,b) materials.

The materials at the end of the listing (e.g., c*3, c*9) are special analytic cross sections for code testing.

Calculations using the S(α,β) scattering formalism (described in the MIX section) can make use of a special set of materials. In describing them, the following notation is used: (A.B) indicates that material A in the material is treated as if it were bound to material B. For example, the MIX specification (H.H2O) or 301001 causes COG to treat the material's hydrogen as if it were bound in water.

The available S(α,β) materials are listed in Appendix.

At the end of the listing is an alphabetical index of all the entries, with page numbers corresponding to the definitions. These page numbers refer to the listing pages, they are not page numbers for the Manual.

These densities and definitions are intended to be an aid to COG users. This is not meant to be an authoritative compilation.

• COGLEX11.2 Listing

Material: Dictionary entry (maximum 8 characters)

ZA: 100000# + 1000Z + A, # = 0 for usual (non-thermal) materials

= instance number for thermal materials

WF: Weight fraction

Material	ZA	WF	ZA	WF	ZA	WF
H	1000	1.000000				
Hydrogen	1000	1.000000				
H1	1001	1.000000				
(H.Li7H-mixed)	2401001	1.000000	-1001	1.000000		
(H.C5O2H8)	1001001	1.000000	-1001	1.000000		
(H.CaH2)	101001	1.000000	-1001	1.000000		
(H.Li6H)	1101001	1.000000	-1001	1.000000		
(C6H6)	501001	1.000000	-1001	0.077000	-6000	0.923000
(Benzene)	501001	1.000000	-1001	0.077000	-6000	0.923000
(H.C6H6)	1301001	1.000000	-1001	1.000000		
(H.C2H5OH)	1401001	1.000000	-1001	1.000000		
(H.C9H12)	1501001	1.000000	-1001	1.000000		
(H.C8H10)	1601001	1.000000	-1001	1.000000		
(H.C7H8)	1701001	1.000000	-1001	1.000000		
(H.C19H16)	1801001	1.000000	-1001	1.000000		
(H.CH2)	201001	1.000000	-1001	1.000000		
(H.CeH2)	601001	1.000000	-1001	1.000000		
(H.C8H8)	1901001	1.000000	-1001	1.000000		
(H.H2O)	301001	1.000000	-1001	1.000000		
(H.TiH2)	701001	1.000000	-1001	1.000000		
(H.YH2)	801001	1.000000	-1001	1.000000		
(H.CH4)	901001	1.000000	-1001	1.000000		
(H.ZrH)	401001	1.000000	-1001	1.000000		
(H.ZrH2)	2201001	1.000000	-1001	1.000000		
(H.ZrHx)	2301001	1.000000	-1001	1.000000		
(H.nCH2H)	2001001	1.000000	-1001	1.000000		
(H.UH3)	2101001	1.000000	-1001	1.000000		
H2	1002	1.000000				
D	1002	1.000000				
Deuterium	1002	1.000000				
(D.Li7D-mixed)	201002	1.000000	-1002	1.000000		
(D.D2O)	101002	1.000000	-1002	1.000000		
(D.CaH2)	301002	1.000000	-1002	1.000000		
H3	1003	1.000000				
T	1003	1.000000				
Tritium	1003	1.000000				
He	2000	1.000000				
Helium	2000	1.000000				
He3	2003	1.000000				
Helium3	2003	1.000000				
He4	2004	1.000000				
Helium4	2004	1.000000				
Li	3000	1.000000				
Lithium	3000	1.000000				
Li6	3006	1.000000				

COG COGLEX Dictionary Listing

10/30/2024

Lithium6	3006	1.000000			
(Li6.Li6H)	103006	1.000000	-3006	1.000000	
Li7	3007	1.000000			
Lithium7	3007	1.000000			
(Li7.Li7H-mixed)	103007	1.000000	-3006	1.000000	
(Li7.Li7D-mixed)	203007	1.000000	-3006	1.000000	
Be	4000	1.000000			
Beryllium	4000	1.000000			
Be7	4007	1.000000			
Be9	4009	1.000000			
Be10	4010	1.000000			
Be11	4011	1.000000			
(Be)	104009	1.000000	-4009	1.000000	
(Beryllium)	104009	1.000000	-4009	1.000000	
(BeO)	204009	1.000000	-4009	0.360000	-8016 0.640000
(Be.Be2C)	404009	1.000000	-4009	1.000000	
(Be.BeO)	304009	1.000000	-4009	1.000000	
B	5000	1.000000			
Boron	5000	1.000000			
B10	5010	1.000000			
Boron10	5010	1.000000			
B11	5011	1.000000			
Boron11	5011	1.000000			
C	6000	1.000000			
Carbon	6000	1.000000			
C10	6010	1.000000			
Carbon10	6010	1.000000			
C11	6011	1.000000			
Carbon11	6011	1.000000			
C12	6012	1.000000			
Carbon12	6012	1.000000			
C13	6013	1.000000			
Carbon13	6013	1.000000			
C15	6015	1.000000			
Carbon15	6015	1.000000			
(C)	106012	1.000000	-6000	1.000000	
(Graphite)	106012	1.000000	-6000	1.000000	
(C-10P)	306012	1.000000	-6000	1.000000	
(C-20P)	1506012	1.000000	-6000	1.000000	
(C-30P)	406012	1.000000	-6000	1.000000	
(C.Be2C)	1206012	1.000000	-6000	1.000000	
(C.C5O2H8)	1306012	1.000000	-6000	1.000000	
(C.C6H6)	506012	1.000000	-6000	1.000000	
(C.C8H8)	1406012	1.000000	-6000	1.000000	
(C.SiC)	206012	1.000000	-6000	1.000000	
(C.C2H5OH)	606012	1.000000	-6000	1.000000	
(C.C9H12)	706012	1.000000	-6000	1.000000	
(C.C8H10)	806012	1.000000	-6000	1.000000	
(C.C7H8)	906012	1.000000	-6000	1.000000	
(C.C19H16)	1006012	1.000000	-6000	1.000000	
(C.CH4)	1106012	1.000000	-6000	1.000000	
(C.UC-100P)	1606012	1.000000	-6000	1.000000	
(C.UC-10P)	1706012	1.000000	-6000	1.000000	
(C.UC-5P)	1806012	1.000000	-6000	1.000000	
(C.UC)	1906012	1.000000	-6000	1.000000	
(C.UC-HALEU)	2006012	1.000000	-6000	1.000000	
(C.UC-HUE)	2106012	1.000000	-6000	1.000000	
(C.ZrC)	2206012	1.000000	-6000	1.000000	
N	7000	1.000000			
Nitrogen	7000	1.000000			
N13	7013	1.000000			
N14	7014	1.000000			
(N.UN)	107014	1.000000	-7014	1.000000	
(N.UN-100P)	207014	1.000000	-7014	1.000000	
(N.UN-10P)	307014	1.000000	-7014	1.000000	
(N.UN-5P)	407014	1.000000	-7014	1.000000	
(N.UN-HALEU)	507014	1.000000	-7014	1.000000	
(N.UN-HEU)	607014	1.000000	-7014	1.000000	

N15	7015	1.000000		
N17	7017	1.000000		
O	8000	1.000000		
Oxygen	8000	1.000000		
O14	8014	1.000000		
Oxygen14	8014	1.000000		
O15	8015	1.000000		
Oxygen15	8015	1.000000		
O16	8016	1.000000		
Oxygen16	8016	1.000000		
O17	8017	1.000000		
Oxygen17	8017	1.000000		
O18	8018	1.000000		
Oxygen18	8018	1.000000		
O19	8019	1.000000		
Oxygen19	8019	1.000000		
O20	8020	1.000000		
Oxygen20	8020	1.000000		
(O.Al2O3)	508016	1.000000	-8016	1.000000
(O.BeO)	208016	1.000000	-8016	1.000000
(O.C2H5OH)	708016	1.000000	-8016	1.000000
(O.UO2)	108016	1.000000	-8016	1.000000
(O.UO2-100P)	1108016	1.000000	-8016	1.000000
(O.UO2-10P)	1208016	1.000000	-8016	1.000000
(O.UO2-5P)	1308016	1.000000	-8016	1.000000
(O.UO2-HALEU)	1408016	1.000000	-8016	1.000000
(O.UO2-HEU)	1508016	1.000000	-8016	1.000000
(O.D2O)	408016	1.000000	-8016	1.000000
(O.C5O2H8)	808016	1.000000	-8016	1.000000
(O.PuO2)	908016	1.000000	-8016	1.000000
(O.H2O)	608016	1.000000	-8016	1.000000
(O.SiO2-alpha)	1008016	1.000000	-8016	1.000000
(Pu.PuO2)	194000	1.000000	-8016	1.000000
F	9000	1.000000		
Fluorine	9000	1.000000		
F17	9017	1.000000		
F18	9018	1.000000		
F19	9019	1.000000		
F20	9020	1.000000		
F21	9021	1.000000		
(F.CF2)	109019	1.000000	-1001	1.000000
(H.HF)	1201001	1.000000	-1001	1.000000
Ne	10000	1.000000		
Neon	10000	1.000000		
Ne18	10018	1.000000		
Neon18	10018	1.000000		
Ne19	10019	1.000000		
Neon19	10019	1.000000		
Ne20	10020	1.000000		
Neon20	10020	1.000000		
Ne21	10021	1.000000		
Neon21	10021	1.000000		
Ne22	10022	1.000000		
Neon22	10022	1.000000		
Ne23	10023	1.000000		
Neon23	10023	1.000000		
Ne24	10024	1.000000		
Neon24	10024	1.000000		
Na	11000	1.000000		
Sodium	11000	1.000000		
Na21	11021	1.000000		
Sodium21	11021	1.000000		
Na22	11022	1.000000		
Sodium22	11022	1.000000		
Na23	11023	1.000000		
Sodium23	11023	1.000000		
Na24	11024	1.000000		
Sodium24	11024	1.000000		

COG COGLEX Dictionary Listing

10/30/2024

Na25	11025	1.000000			
Sodium25	11025	1.000000			
Mg	12000	1.000000			
Magnesium	12000	1.000000			
Mg22	12022	1.000000			
Mg23	12023	1.000000			
Mg24	12024	1.000000			
Mg25	12025	1.000000			
Mg26	12026	1.000000			
Mg27	12027	1.000000			
Mg28	12028	1.000000			
(Mg24)	112024	1.000000	-12024	1.000000	
Al	13000	1.000000			
Aluminum	13000	1.000000			
Al24	13024	1.000000			
Al25	13025	1.000000			
Al26	13026	1.000000			
Al27	13027	1.000000			
Al28	13028	1.000000			
Al29	13029	1.000000			
(Al)	113027	1.000000	-13027	1.000000	
(Aluminum)	113027	1.000000	-13027	1.000000	
(Al.Al2O3)	213027	1.000000	-13027	1.000000	
Si	14000	1.000000			
Silicon	14000	1.000000			
Si26	14026	1.000000			
Si27	14027	1.000000			
Si28	14028	1.000000			
Si29	14029	1.000000			
Si30	14030	1.000000			
Si31	14031	1.000000			
Si32	14032	1.000000			
Si33	14033	1.000000			
Si34	14034	1.000000			
(SiO2)	114000	1.000000	-14028	0.430700	-14029 0.021900
	-14030	0.014400	-8016	0.533000	
(Si.SiC)	214000	1.000000	-14028	0.922300	-14029 0.046900
	-14030	0.030800			
(Si)	314000	1.000000	-14028	0.922300	-14029 0.046900
	-14030	0.030800			
(Si.SiO2-alpha)	414000	1.000000	-14028	0.922300	-14029 0.046900
	-14030	0.030800			
P	15000	1.000000			
Phosphorus	15000	1.000000			
P29	15029	1.000000			
P30	15030	1.000000			
P31	15031	1.000000			
P32	15032	1.000000			
P33	15033	1.000000			
S	16000	1.000000			
Sulfur	16000	1.000000			
S30	16030	1.000000			
Sulfur30	16030	1.000000			
S31	16031	1.000000			
Sulfur31	16031	1.000000			
S32	16032	1.000000			
Sulfur32	16032	1.000000			
S33	16033	1.000000			
Sulfur33	16033	1.000000			
S34	16034	1.000000			
Sulfur34	16034	1.000000			
S35	16035	1.000000			
Sulfur35	16035	1.000000			
S36	16036	1.000000			
Sulfur36	16036	1.000000			
S37	16037	1.000000			
Sulfur37	16037	1.000000			
S38	16038	1.000000			

Sulfur38	16038	1.000000		
Cl	17000	1.000000		
Chlorine	17000	1.000000		
Cl33	17033	1.000000		
Cl34	17034	1.000000		
Cl35	17035	1.000000		
Cl36	17036	1.000000		
Cl37	17037	1.000000		
Cl38	17038	1.000000		
Cl39	17039	1.000000		
Ar	18000	1.000000		
Argon	18000	1.000000		
Ar34	18034	1.000000		
Argon34	18034	1.000000		
Ar35	18035	1.000000		
Argon35	18035	1.000000		
Ar36	18036	1.000000		
Argon36	18036	1.000000		
Ar38	18038	1.000000		
Argon38	18038	1.000000		
Ar40	18040	1.000000		
Argon40	18040	1.000000		
Ar41	18041	1.000000		
Argon41	18041	1.000000		
K	19000	1.000000		
Potassium	19000	1.000000		
K37	19037	1.000000		
K38	19038	1.000000		
K39	19039	1.000000		
K40	19040	1.000000		
K41	19041	1.000000		
K42	19042	1.000000		
K43	19043	1.000000		
Ca	20000	1.000000		
Calcium	20000	1.000000		
Ca40	20040	1.000000		
Ca41	20041	1.000000		
Ca42	20042	1.000000		
Ca43	20043	1.000000		
Ca44	20044	1.000000		
Ca45	20045	1.000000		
Ca46	20046	1.000000		
Ca47	20047	1.000000		
Ca48	20048	1.000000		
Ca49	20049	1.000000		
Ca50	20050	1.000000		
(Ca.CaH2)	120040	1.000000	-20040	1.000000
Sc	21000	1.000000		
Scandium	21000	1.000000		
Sc41	21041	1.000000		
Sc42	21042	1.000000		
Sc43	21043	1.000000		
Sc44	21044	1.000000		
Sc45	21045	1.000000		
Sc46	21046	1.000000		
Sc47	21047	1.000000		
Sc48	21048	1.000000		
Sc49	21049	1.000000		
Sc50	21050	1.000000		
Ti	22000	1.000000		
Titanium	22000	1.000000		
Ti44	22044	1.000000		
Ti45	22045	1.000000		
Ti46	22046	1.000000		
Ti47	22047	1.000000		
Ti48	22048	1.000000		
Ti49	22049	1.000000		
Ti50	22050	1.000000		

Ti51	22051	1.000000		
Ti52	22052	1.000000		
V	23000	1.000000		
Vanadium	23000	1.000000		
V46	23046	1.000000		
V47	23047	1.000000		
V48	23048	1.000000		
V49	23049	1.000000		
V50	23050	1.000000		
V51	23051	1.000000		
V52	23052	1.000000		
V53	23053	1.000000		
Cr	24000	1.000000		
Chromium	24000	1.000000		
Cr47	24047	1.000000		
Cr48	24048	1.000000		
Cr49	24049	1.000000		
Cr50	24050	1.000000		
Cr51	24051	1.000000		
Cr52	24052	1.000000		
Cr53	24053	1.000000		
Cr54	24054	1.000000		
Cr55	24055	1.000000		
Cr56	24056	1.000000		
Mn	25000	1.000000		
Manganese	25000	1.000000		
Mn50	25050	1.000000		
Mn51	25051	1.000000		
Mn52	25052	1.000000		
Mn53	25053	1.000000		
Mn54	25054	1.000000		
Mn55	25055	1.000000		
Mn56	25056	1.000000		
Mn57	25057	1.000000		
Fe	26000	1.000000		
Iron	26000	1.000000		
Fe52	26052	1.000000		
Iron52	26052	1.000000		
Fe53	26053	1.000000		
Iron53	26053	1.000000		
Fe54	26054	1.000000		
Iron54	26054	1.000000		
Fe55	26055	1.000000		
Iron55	26055	1.000000		
Fe56	26056	1.000000		
Iron56	26056	1.000000		
Fe57	26057	1.000000		
Iron57	26057	1.000000		
Fe58	26058	1.000000		
Iron58	26058	1.000000		
Fe59	26059	1.000000		
Iron59	26059	1.000000		
Fe60	26060	1.000000		
Iron60	26060	1.000000		
Fe61	26061	1.000000		
Iron61	26061	1.000000		
Fe62	26062	1.000000		
Iron62	26062	1.000000		
(Fe56)	126056	1.000000	-26056	1.000000
(Iron56)	126056	1.000000	-26056	1.000000
Co	27000	1.000000		
Cobalt	27000	1.000000		
Co57	27057	1.000000		
Cobalt57	27057	1.000000		
Co58	27058	1.000000		
Cobalt58	27058	1.000000		
Co58m	-27058	1.000000		
Co59	27059	1.000000		

Cobalt59	27059	1.000000
Co60	27060	1.000000
Cobalt60	27060	1.000000
Co61	27061	1.000000
Cobalt61	27061	1.000000
Ni	28000	1.000000
Nickel	28000	1.000000
Ni56	28056	1.000000
Nickel56	28056	1.000000
Ni57	28057	1.000000
Nickel57	28057	1.000000
Ni58	28058	1.000000
Nickel58	28058	1.000000
Ni59	28059	1.000000
Nickel59	28059	1.000000
Ni60	28060	1.000000
Nickel60	28060	1.000000
Ni61	28061	1.000000
Nickel61	28061	1.000000
Ni62	28062	1.000000
Nickel62	28062	1.000000
Ni63	28063	1.000000
Nickel63	28063	1.000000
Ni64	28064	1.000000
Nickel64	28064	1.000000
Ni65	28065	1.000000
Nickel65	28065	1.000000
Ni66	28066	1.000000
Nickel66	28066	1.000000
Ni67	28067	1.000000
Nickel67	28067	1.000000
Ni68	28068	1.000000
Nickel68	28068	1.000000
Cu	29000	1.000000
Copper	29000	1.000000
Cu59	29059	1.000000
Copper59	29059	1.000000
Cu60	29060	1.000000
Copper60	29060	1.000000
Cu61	29061	1.000000
Copper61	29061	1.000000
Cu62	29062	1.000000
Copper62	29062	1.000000
Cu63	29063	1.000000
Copper63	29063	1.000000
Cu64	29064	1.000000
Copper64	29064	1.000000
Cu65	29065	1.000000
Copper65	29065	1.000000
Cu66	29066	1.000000
Copper66	29066	1.000000
Cu67	29067	1.000000
Copper67	29067	1.000000
Cu68	29068	1.000000
Copper68	29068	1.000000
Cu69	29069	1.000000
Copper69	29069	1.000000
Zn	30000	1.000000
Zinc	30000	1.000000
Zn60	30060	1.000000
Zinc60	30060	1.000000
Zn61	30061	1.000000
Zinc61	30061	1.000000
Zn62	30062	1.000000
Zinc62	30062	1.000000
Zn63	30063	1.000000
Zinc63	30063	1.000000
Zn64	30064	1.000000

Zinc64	30064	1.000000
Zn65	30065	1.000000
Zinc65	30065	1.000000
Zn66	30066	1.000000
Zinc66	30066	1.000000
Zn67	30067	1.000000
Zinc67	30067	1.000000
Zn68	30068	1.000000
Zinc68	30068	1.000000
Zn69	30069	1.000000
Zinc69	30069	1.000000
Zn70	30070	1.000000
Zinc70	30070	1.000000
Zn71	30071	1.000000
Zinc71	30071	1.000000
Zn72	30072	1.000000
Zinc72	30072	1.000000
Zn73	30073	1.000000
Zinc73	30073	1.000000
Ga	31000	1.000000
Galium	31000	1.000000
Ga67	31067	1.000000
Galium67	31067	1.000000
Ga68	31068	1.000000
Galium68	31068	1.000000
Ga69	31069	1.000000
Galium69	31069	1.000000
Ga70	31070	1.000000
Galium70	31070	1.000000
Ga71	31071	1.000000
Galium71	31071	1.000000
Ga72	31072	1.000000
Galium72	31072	1.000000
Ga73	31073	1.000000
Galium73	31073	1.000000
Ge	32000	1.000000
Germanium	32000	1.000000
Ge68	32068	1.000000
Ge69	32069	1.000000
Ge70	32070	1.000000
Ge71	32071	1.000000
Ge72	32072	1.000000
Ge73	32073	1.000000
Ge74	32074	1.000000
Ge75	32075	1.000000
Ge76	32076	1.000000
Ge77	32077	1.000000
Ge78	32078	1.000000
Ge79	32079	1.000000
As	33000	1.000000
Arsenic	33000	1.000000
As72	33072	1.000000
As73	33073	1.000000
As74	33074	1.000000
As75	33075	1.000000
As76	33076	1.000000
As77	33077	1.000000
As79	33079	1.000000
Se	34000	1.000000
Selenium	34000	1.000000
Se72	34072	1.000000
Se73	34073	1.000000
Se74	34074	1.000000
Se75	34075	1.000000
Se76	34076	1.000000
Se77	34077	1.000000
Se78	34078	1.000000
Se79	34079	1.000000

Se80	34080	1.000000
Se81	34081	1.000000
Se82	34082	1.000000
Se83	34083	1.000000
Se84	34084	1.000000
Br	35000	1.000000
Bromine	35000	1.000000
Br75	35075	1.000000
Br76	35076	1.000000
Br77	35077	1.000000
Br78	35078	1.000000
Br79	35079	1.000000
Br80	35080	1.000000
Br81	35081	1.000000
Br82	35082	1.000000
Br83	35083	1.000000
Kr	36000	1.000000
Krypton	36000	1.000000
Kr76	36076	1.000000
Kr77	36077	1.000000
Kr78	36078	1.000000
Kr79	36079	1.000000
Kr80	36080	1.000000
Kr81	36081	1.000000
Kr82	36082	1.000000
Kr83	36083	1.000000
Kr84	36084	1.000000
Kr85	36085	1.000000
Kr86	36086	1.000000
Kr87	36087	1.000000
Kr88	36088	1.000000
Rb	37000	1.000000
Rubidium	37000	1.000000
Rb77	37077	1.000000
Rb78	37078	1.000000
Rb79	37079	1.000000
Rb80	37080	1.000000
Rb81	37081	1.000000
Rb82	37082	1.000000
Rb83	37083	1.000000
Rb84	37084	1.000000
Rb85	37085	1.000000
Rb86	37086	1.000000
Rb87	37087	1.000000
Rb88	37088	1.000000
Rb89	37089	1.000000
Sr	38000	1.000000
Strontium	38000	1.000000
Sr82	38082	1.000000
Sr83	38083	1.000000
Sr84	38084	1.000000
Sr85	38085	1.000000
Sr86	38086	1.000000
Sr87	38087	1.000000
Sr88	38088	1.000000
Sr89	38089	1.000000
Sr90	38090	1.000000
Sr91	38091	1.000000
Sr92	38092	1.000000
Y	39000	1.000000
Yttrium	39000	1.000000
Y84	39084	1.000000
Y85	39085	1.000000
Y86	39086	1.000000
Y87	39087	1.000000
Y88	39088	1.000000
Y89	39089	1.000000
Y90	39090	1.000000

Y91	39091	1.000000			
Y92	39092	1.000000			
Y93	39093	1.000000			
(Y.YH2)	139089	1.000000	-39089	1.000000	
Zr	40000	1.000000			
Zirconium	40000	1.000000			
Zr86	40086	1.000000			
Zr87	40087	1.000000			
Zr88	40088	1.000000			
Zr89	40089	1.000000			
Zr90	40090	1.000000			
Zr91	40091	1.000000			
Zr92	40092	1.000000			
Zr93	40093	1.000000			
Zr94	40094	1.000000			
Zr95	40095	1.000000			
Zr96	40096	1.000000			
Zr97	40097	1.000000			
Zr98	40098	1.000000			
(Zr.ZrH)	140000	1.000000	-40090	0.507080	-40091 0.111840
	-40092	0.172740	-40094	0.178910	-40096 0.029430
(Zr.ZrHx	240000	1.000000	-40090	0.507080	-40091 0.111840
	-40092	0.172740	-40094	0.178910	-40096 0.029430
(Zr.ZrC)	340000	1.000000	-40090	0.507080	-40091 0.111840
	-40092	0.172740	-40094	0.178910	-40096 0.029430
Nb	41000	1.000000			
Niobium	41000	1.000000			
Nb87	41087	1.000000			
Nb88	41088	1.000000			
Nb89	41089	1.000000			
Nb90	41090	1.000000			
Nb91	41091	1.000000			
Nb92	41092	1.000000			
Nb93	41093	1.000000			
Nb94	41094	1.000000			
Nb95	41095	1.000000			
Nb96	41096	1.000000			
Nb97	41097	1.000000			
Nb98	41098	1.000000			
Nb99	41099	1.000000			
Nb100	41100	1.000000			
Mo	42000	1.000000			
Molybdenum	42000	1.000000			
Mo90	42090	1.000000			
Mo91	42091	1.000000			
Mo92	42092	1.000000			
Mo93	42093	1.000000			
Mo94	42094	1.000000			
Mo95	42095	1.000000			
Mo96	42096	1.000000			
Mo97	42097	1.000000			
Mo98	42098	1.000000			
Mo99	42099	1.000000			
Mo100	42100	1.000000			
Mo101	42101	1.000000			
Mo102	42102	1.000000			
Tc	43000	1.000000			
Technetium	43000	1.000000			
Tc91	43091	1.000000			
Tc92	43092	1.000000			
Tc93	43093	1.000000			
Tc94	43094	1.000000			
Tc95	43095	1.000000			
Tc96	43096	1.000000			
Tc97	43097	1.000000			
Tc98	43098	1.000000			
Tc99	43099	1.000000			
Tc100	43100	1.000000			

Tc101	43101	1.000000
Tc102	43102	1.000000
Tc103	43103	1.000000
Tc104	43104	1.000000
Ru	44000	1.000000
Ruthenium	44000	1.000000
Ru94	44094	1.000000
Ru95	44095	1.000000
Ru96	44096	1.000000
Ru97	44097	1.000000
Ru98	44098	1.000000
Ru99	44099	1.000000
Ru100	44100	1.000000
Ru101	44101	1.000000
Ru102	44102	1.000000
Ru103	44103	1.000000
Ru104	44104	1.000000
Ru105	44105	1.000000
Ru106	44106	1.000000
Rh	45000	1.000000
Rhodium	45000	1.000000
Rh096	45096	1.000000
Rh097	45097	1.000000
Rh098	45098	1.000000
Rh099	45099	1.000000
Rh100	45100	1.000000
Rh101	45101	1.000000
Rh102	45102	1.000000
Rh103	45103	1.000000
Rh104	45104	1.000000
Rh105	45105	1.000000
Rh106	45106	1.000000
Rh105	45105	1.000000
Pd	46000	1.000000
Palladium	46000	1.000000
Pd100	46100	1.000000
Pd101	46101	1.000000
Pd102	46102	1.000000
Pd103	46103	1.000000
Pd104	46104	1.000000
Pd105	46105	1.000000
Pd106	46106	1.000000
Pd107	46107	1.000000
Pd108	46108	1.000000
Pd109	46109	1.000000
Pd110	46110	1.000000
Pd111	46111	1.000000
Pd112	46112	1.000000
Ag	47000	1.000000
Silver	47000	1.000000
Ag105	47105	1.000000
Ag106	47106	1.000000
Ag107	47107	1.000000
Ag108	47108	1.000000
Ag109	47109	1.000000
Ag110	47110	1.000000
Ag110m	-47110	1.000000
Ag111	47111	1.000000
Cd	48000	1.000000
Cadmium	48000	1.000000
Cd104	48104	1.000000
Cd105	48105	1.000000
Cd106	48106	1.000000
Cd107	48107	1.000000
Cd108	48108	1.000000
Cd109	48109	1.000000
Cd110	48110	1.000000
Cd111	48111	1.000000

Cd112	48112	1.000000
Cd113	48113	1.000000
Cd114	48114	1.000000
Cd115	48115	1.000000
Cd115m	-48115	1.000000
Cd116	48116	1.000000
Cd117	48117	1.000000
Cd118	48118	1.000000
In	49000	1.000000
Indium	49000	1.000000
In111	49111	1.000000
In112	49112	1.000000
In113	49113	1.000000
In114	49114	1.000000
In115	49115	1.000000
In116	49116	1.000000
In117	49117	1.000000
Sn	50000	1.000000
Tin	50000	1.000000
Sn110	50110	1.000000
Tin110	50110	1.000000
Sn111	50111	1.000000
Tin111	50111	1.000000
Sn112	50112	1.000000
Tin112	50112	1.000000
Sn113	50113	1.000000
Tin113	50113	1.000000
Sn114	50114	1.000000
Tin114	50114	1.000000
Sn115	50115	1.000000
Tin115	50115	1.000000
Sn116	50116	1.000000
Tin116	50116	1.000000
Sn117	50117	1.000000
Tin117	50117	1.000000
Sn118	50118	1.000000
Tin118	50118	1.000000
Sn119	50119	1.000000
Tin119	50119	1.000000
Sn120	50120	1.000000
Tin120	50120	1.000000
Sn121	50121	1.000000
Tin121	50121	1.000000
Sn122	50122	1.000000
Tin122	50122	1.000000
Sn123	50123	1.000000
Tin123	50123	1.000000
Sn124	50124	1.000000
Tin124	50124	1.000000
Sn125	50125	1.000000
Tin125	50125	1.000000
Sn126	50126	1.000000
Tin126	50126	1.000000
Sb	51000	1.000000
Antimony	51000	1.000000
Sb119	51119	1.000000
Sb120	51120	1.000000
Sb121	51121	1.000000
Sb122	51122	1.000000
Sb123	51123	1.000000
Sb124	51124	1.000000
Sb125	51125	1.000000
Sb126	51126	1.000000
Te	52000	1.000000
Tellurium	52000	1.000000
Te118	52118	1.000000
Te119	52119	1.000000
Te120	52120	1.000000

Te121	52121	1.000000
Te122	52122	1.000000
Te123	52123	1.000000
Te124	52124	1.000000
Te125	52125	1.000000
Te126	52126	1.000000
Te127	52127	1.000000
Te127m	-52127	1.000000
Te128	52128	1.000000
Te129	52129	1.000000
Te129m	-52129	1.000000
Te130	52130	1.000000
Te131	52131	1.000000
Te132	52132	1.000000
Te133	52133	1.000000
I	53000	1.000000
Iodine	53000	1.000000
I122	53122	1.000000
I123	53123	1.000000
I124	53124	1.000000
I125	53125	1.000000
I126	53126	1.000000
I127	53127	1.000000
I128	53128	1.000000
I129	53129	1.000000
I130	53130	1.000000
I131	53131	1.000000
I132	53132	1.000000
I133	53133	1.000000
I134	53134	1.000000
I135	53135	1.000000
Xe	54000	1.000000
Xenon	54000	1.000000
Xe122	54122	1.000000
Xenon122	54122	1.000000
Xe123	54123	1.000000
Xenon123	54123	1.000000
Xe124	54124	1.000000
Xenon124	54124	1.000000
Xe125	54125	1.000000
Xenon125	54125	1.000000
Xe126	54126	1.000000
Xenon126	54126	1.000000
Xe127	54127	1.000000
Xenon127	54127	1.000000
Xe128	54128	1.000000
Xenon128	54128	1.000000
Xe129	54129	1.000000
Xenon129	54129	1.000000
Xe130	54130	1.000000
Xenon130	54130	1.000000
Xe131	54131	1.000000
Xenon131	54131	1.000000
Xe132	54132	1.000000
Xenon132	54132	1.000000
Xe133	54133	1.000000
Xenon133	54133	1.000000
Xe134	54134	1.000000
Xenon134	54134	1.000000
Xe135	54135	1.000000
Xenon135	54135	1.000000
Xe136	54136	1.000000
Xenon136	54136	1.000000
Xe137	54137	1.000000
Xenon137	54137	1.000000
Xe138	54138	1.000000
Xenon138	54138	1.000000
Cs	55000	1.000000

Cesium	55000	1.000000
Cs131	55131	1.000000
Cs132	55132	1.000000
Cs133	55133	1.000000
Cs134	55134	1.000000
Cs135	55135	1.000000
Cs136	55136	1.000000
Cs137	55137	1.000000
Ba	56000	1.000000
Barium	56000	1.000000
Ba128	56128	1.000000
Ba129	56129	1.000000
Ba130	56130	1.000000
Ba131	56131	1.000000
Ba132	56133	1.000000
Ba133	56132	1.000000
Ba134	56134	1.000000
Ba135	56135	1.000000
Ba136	56136	1.000000
Ba137	56137	1.000000
Ba138	56138	1.000000
Ba139	56139	1.000000
Ba140	56140	1.000000
La	57000	1.000000
Lanthanum	57000	1.000000
La135	57135	1.000000
La136	57136	1.000000
La137	57137	1.000000
La138	57138	1.000000
La139	57139	1.000000
La140	57140	1.000000
La141	57141	1.000000
Ce	58000	1.000000
Cerium	58000	1.000000
Ce134	58134	1.000000
Ce135	58135	1.000000
Ce136	58136	1.000000
Ce137	58137	1.000000
Ce138	58138	1.000000
Ce139	58139	1.000000
Ce140	58140	1.000000
Ce141	58141	1.000000
Ce142	58142	1.000000
Ce143	58143	1.000000
Ce144	58144	1.000000
Pr	59000	1.000000
Praseodymium	59000	1.000000
Pr139	59139	1.000000
Pr140	59140	1.000000
Pr141	59141	1.000000
Pr142	59142	1.000000
Pr143	59143	1.000000
Nd	60000	1.000000
Neodymium	60000	1.000000
Nd140	60140	1.000000
Nd141	60141	1.000000
Nd142	60142	1.000000
Nd143	60143	1.000000
Nd144	60144	1.000000
Nd145	60145	1.000000
Nd146	60146	1.000000
Nd147	60147	1.000000
Nd148	60148	1.000000
Nd149	60149	1.000000
Nd150	60150	1.000000
Nd151	60151	1.000000
Nd152	60152	1.000000
Pm	61000	1.000000

Promethium	61000	1.000000
Pm147	61147	1.000000
Pm148	61148	1.000000
Pm148m	-61148	1.000000
Pm149	61149	1.000000
Pm150	61150	1.000000
Pm151	61151	1.000000
Sm	62000	1.000000
Samarium	62000	1.000000
Sm142	62142	1.000000
Sm143	62143	1.000000
Sm144	62144	1.000000
Sm145	62145	1.000000
Sm146	62146	1.000000
Sm147	62147	1.000000
Sm148	62148	1.000000
Sm149	62149	1.000000
Sm150	62150	1.000000
Sm151	62151	1.000000
Sm152	62152	1.000000
Sm153	62153	1.000000
Sm154	62154	1.000000
Sm145	62145	1.000000
Sm146	62146	1.000000
Eu	63000	1.000000
Europium	63000	1.000000
Eu145	63145	1.000000
Eu146	63146	1.000000
Eu147	63147	1.000000
Eu148	63148	1.000000
Eu149	63149	1.000000
Eu150	63150	1.000000
Eu151	63151	1.000000
Eu152	63152	1.000000
Eu153	63153	1.000000
Eu154	63154	1.000000
Eu155	63155	1.000000
Eu156	63156	1.000000
Eu157	63157	1.000000
Gd	64000	1.000000
Gadolinium	64000	1.000000
Gd146	64146	1.000000
Gd147	64147	1.000000
Gd148	64148	1.000000
Gd149	64149	1.000000
Gd150	64150	1.000000
Gd151	64151	1.000000
Gd152	64152	1.000000
Gd153	64153	1.000000
Gd154	64154	1.000000
Gd155	64155	1.000000
Gd156	64156	1.000000
Gd157	64157	1.000000
Gd158	64158	1.000000
Gd159	64159	1.000000
Gd160	64160	1.000000
Gd161	64161	1.000000
Gd162	64162	1.000000
Tb	65000	1.000000
Terbium	65000	1.000000
Tb156	65156	1.000000
Tb157	65157	1.000000
Tb158	65158	1.000000
Tb159	65159	1.000000
Tb160	65160	1.000000
Tb161	65161	1.000000
Dy	66000	1.000000
Dysprosium	66000	1.000000

Dy154	66154	1.000000
Dy155	66155	1.000000
Dy156	66156	1.000000
Dy157	66157	1.000000
Dy158	66158	1.000000
Dy159	66159	1.000000
Dy160	66160	1.000000
Dy161	66161	1.000000
Dy162	66162	1.000000
Dy163	66163	1.000000
Dy164	66164	1.000000
Dy166	66166	1.000000
Ho	67000	1.000000
Holmium	67000	1.000000
Ho163	67163	1.000000
Ho164	67164	1.000000
Ho165	67165	1.000000
Ho166	67166	1.000000
Ho166m	-67166	1.000000
Ho167	67167	1.000000
Er	68000	1.000000
Erbium	68000	1.000000
Er160	68160	1.000000
Er161	68161	1.000000
Er162	68162	1.000000
Er163	68163	1.000000
Er164	68164	1.000000
Er165	68165	1.000000
Er166	68166	1.000000
Er167	68167	1.000000
Er168	68168	1.000000
Er169	68169	1.000000
Er170	68170	1.000000
Er171	68171	1.000000
Er172	68172	1.000000
Tm	69000	1.000000
Thulium	69000	1.000000
Tm167	69167	1.000000
Tm168	69168	1.000000
Tm169	69169	1.000000
Tm170	69170	1.000000
Tm171	69171	1.000000
Yb	70000	1.000000
Ytterbium	70000	1.000000
Yb166	70166	1.000000
Yb167	70167	1.000000
Yb168	70168	1.000000
Yb169	70169	1.000000
Yb170	70170	1.000000
Yb171	70171	1.000000
Yb172	70172	1.000000
Yb173	70173	1.000000
Yb174	70174	1.000000
Yb175	70175	1.000000
Yb176	70176	1.000000
Yb177	70177	1.000000
Yb178	70178	1.000000
Lu	71000	1.000000
Lutetium	71000	1.000000
Lu173	71173	1.000000
Lu174	71174	1.000000
Lu175	71175	1.000000
Lu176	71176	1.000000
Lu177	71177	1.000000
Lu178	71178	1.000000
Hf	72000	1.000000
Hafnium	72000	1.000000
Hf172	72172	1.000000

Hf173	72173	1.000000
Hf174	72174	1.000000
Hf175	72175	1.000000
Hf176	72176	1.000000
Hf177	72177	1.000000
Hf178	72178	1.000000
Hf179	72179	1.000000
Hf180	72180	1.000000
Hf181	72181	1.000000
Hf182	72182	1.000000
Hf183	72183	1.000000
Hf184	72184	1.000000
Ta	73000	1.000000
Tantalum	73000	1.000000
Ta178	73178	1.000000
Ta179	73179	1.000000
Ta180	73180	1.000000
Ta181	73181	1.000000
Ta182	73182	1.000000
Ta183	73183	1.000000
W	74000	1.000000
Tungsten	74000	1.000000
W178	74178	1.000000
W179	74179	1.000000
W180	74180	1.000000
W181	74181	1.000000
W182	74182	1.000000
W183	74183	1.000000
W184	74184	1.000000
W185	74185	1.000000
W186	74186	1.000000
W187	74187	1.000000
W188	74188	1.000000
Re	75000	1.000000
Rhenium	75000	1.000000
Re183	75183	1.000000
Re184	75184	1.000000
Re185	75185	1.000000
Re186	75186	1.000000
Re187	75187	1.000000
Re188	75188	1.000000
Re189	75189	1.000000
Os	76000	1.000000
Osmium	76000	1.000000
Os182	76182	1.000000
Os183	76183	1.000000
Os184	76184	1.000000
Os185	76185	1.000000
Os186	76186	1.000000
Os187	76187	1.000000
Os188	76188	1.000000
Os189	76189	1.000000
Os190	76190	1.000000
Os191	76191	1.000000
Os192	76192	1.000000
Os193	761893	1.000000
Os194	76194	1.000000
Ir	77000	1.000000
Iridium	77000	1.000000
Ir184	77184	1.000000
Ir185	77185	1.000000
Ir186	77186	1.000000
Ir187	77187	1.000000
Ir188	77188	1.000000
Ir189	77189	1.000000
Ir190	77190	1.000000
Ir191	77191	1.000000
Ir192	77192	1.000000

Ir193	77193	1.000000
Ir194	77194	1.000000
Ir195	77195	1.000000
Ir196	77196	1.000000
Ir197	77197	1.000000
Ir198	77198	1.000000
Pt	78000	1.000000
Platinum	78000	1.000000
Pt188	78188	1.000000
Pt189	78189	1.000000
Pt190	78190	1.000000
Pt191	78191	1.000000
Pt192	78192	1.000000
Pt193	78190	1.000000
Pt194	78194	1.000000
Pt195	78195	1.000000
Pt196	78196	1.000000
Pt197	78197	1.000000
Pt198	78198	1.000000
Pt199	78199	1.000000
Pt200	78200	1.000000
Au	79000	1.000000
Gold	79000	1.000000
Au193	79193	1.000000
Gold193	79193	1.000000
Au194	79194	1.000000
Gold194	79194	1.000000
Au195	79195	1.000000
Gold195	79195	1.000000
Au196	79196	1.000000
Gold196	79196	1.000000
Au197	79197	1.000000
Gold197	79197	1.000000
Au198	79198	1.000000
Gold198	79198	1.000000
Au199	79199	1.000000
Gold199	79199	1.000000
Hg	80000	1.000000
Mercury	80000	1.000000
Hg194	80194	1.000000
Hg195	80195	1.000000
Hg196	80196	1.000000
Hg197	80197	1.000000
Hg198	80198	1.000000
Hg199	80199	1.000000
Hg200	80200	1.000000
Hg201	80201	1.000000
Hg202	80202	1.000000
Hg203	80203	1.000000
Hg204	80204	1.000000
Hg205	80205	1.000000
Hg206	80206	1.000000
Tl	81000	1.000000
Thallium	81000	1.000000
Tl201	81201	1.000000
Tl202	81202	1.000000
Tl203	81203	1.000000
Tl204	81204	1.000000
Tl205	81205	1.000000
Tl206	81206	1.000000
Tl207	81207	1.000000
Pb	82000	1.000000
Lead	82000	1.000000
Pb200	82200	1.000000
Lead200	82200	1.000000
Pb201	82201	1.000000
Lead201	82201	1.000000
Pb202	82202	1.000000

Lead202	82202	1.000000
Pb203	82203	1.000000
Lead203	82203	1.000000
Pb204	82204	1.000000
Lead204	82204	1.000000
Pb205	82205	1.000000
Lead205	82205	1.000000
Pb206	82206	1.000000
Lead206	82206	1.000000
Pb207	82207	1.000000
Lead207	82207	1.000000
Pb208	82208	1.000000
Lead208	82208	1.000000
Pb209	82209	1.000000
Lead209	82209	1.000000
Pb210	82210	1.000000
Lead210	82210	1.000000
Bi	83000	1.000000
Bismuth	83000	1.000000
Bi206	83206	1.000000
Bi207	83207	1.000000
Bi208	83208	1.000000
Bi209	83209	1.000000
Bi210	83210	1.000000
Bi211	83211	1.000000
Po	84000	1.000000
Polonium	84000	1.000000
Po209	84209	1.000000
At	85000	1.000000
Astatine	85000	1.000000
Rn	86000	1.000000
Radon	86000	1.000000
Rn219	86219	1.000000
Rn220	86220	1.000000
Rn221	86221	1.000000
Rn222	86222	1.000000
Rn223	86223	1.000000
Rn224	86224	1.000000
Fr	87000	1.000000
Francium	87000	1.000000
Ra	88000	1.000000
Radium	88000	1.000000
Ra223	88223	1.000000
Ra224	88224	1.000000
Ra225	88225	1.000000
Ra226	88226	1.000000
Ra227	88227	1.000000
Ra228	88228	1.000000
Ac	89000	1.000000
Actinium	89000	1.000000
Ac225	89225	1.000000
Ac226	89226	1.000000
Ac227	89227	1.000000
Th	90000	1.000000
Thorium	90000	1.000000
Th227	90227	1.000000
Th228	90228	1.000000
Th229	90229	1.000000
Th230	90230	1.000000
Th231	90231	1.000000
Th232	90232	1.000000
Th233	90233	1.000000
Th234	90234	1.000000
Pa	91000	1.000000
Protactinium	91000	1.000000
Pa229	91229	1.000000
Pa230	91230	1.000000
Pa231	91231	1.000000

Pa232	91232	1.000000		
Pa233	91233	1.000000		
U	92000	1.000000		
Uranium	92000	1.000000		
U230	92230	1.000000		
U231	92231	1.000000		
U232	92232	1.000000		
U233	92233	1.000000		
U234	92234	1.000000		
U235	92235	1.000000		
U236	92236	1.000000		
U237	92237	1.000000		
U238	92238	1.000000		
U239	92239	1.000000		
U240	92240	1.000000		
U241	92241	1.000000		
(U)	392238	1.000000	-92238	1.000000
(U-10P)	492238	1.000000	-92238	1.000000
(U-5P)	592238	1.000000	-92238	1.000000
(U-HALEU)	692238	1.000000	-92238	1.000000
(U-HEU)	792238	1.000000	-92238	1.000000
(U.UC)	892238	1.000000	-92238	1.000000
(U.UC-100P)	992238	1.000000	-92238	1.000000
(U.UC-10P)	1092238	1.000000	-92238	1.000000
(U.UC-5P)	1192238	1.000000	-92238	1.000000
(U.UC-HALEU)	1292238	1.000000	-92238	1.000000
(U.UC-HEU)	1392238	1.000000	-92238	1.000000
(U.UN)	292238	1.000000	-92238	1.000000
(U.UN-100P)	1492238	1.000000	-92238	1.000000
(U.UN-10P)	1592238	1.000000	-92238	1.000000
(U.UN-5P)	1692238	1.000000	-92238	1.000000
(U.UN-HALEU)	1792238	1.000000	-92238	1.000000
(U.UN-HEU)	1892238	1.000000	-92238	1.000000
(U.UO2)	192238	1.000000	-92238	1.000000
(U.UO2-100P)	1992238	1.000000	-92238	1.000000
(U.UO2-10P)	2092238	1.000000	-92238	1.000000
(U.UO2-5)	2192238	1.000000	-92238	1.000000
(U.UO2-HALEU)	2292238	1.000000	-92238	1.000000
(U.UO2-HUE)	2392238	1.000000	-92238	1.000000
Np	93000	1.000000		
Neptunium	93000	1.000000		
Np234	93234	1.000000		
Np235	93235	1.000000		
Np236	93236	1.000000		
Np237	93237	1.000000		
Np238	93238	1.000000		
Np239	93239	1.000000		
Pu	94000	1.000000		
Plutonium	94000	1.000000		
Pu236	94236	1.000000		
Pu237	94237	1.000000		
Pu238	94238	1.000000		
Pu239	94239	1.000000		
Pu240	94240	1.000000		
Pu241	94241	1.000000		
Pu242	94242	1.000000		
Pu243	94243	1.000000		
Pu244	94244	1.000000		
Pu245	94245	1.000000		
Pu246	94246	1.000000		
Am	95000	1.000000		
Americium	95000	1.000000		
Am239	95239	1.000000		
Am240	95240	1.000000		
Am241	95241	1.000000		
Am242	95242	1.000000		
Am242m	-95242	1.000000		
Am243	95243	1.000000		

Am244	95244	1.000000				
Am244m	-95244	1.000000				
Am245	95245	1.000000				
Cm	96000	1.000000				
Curium	96000	1.000000				
Cm240	96240	1.000000				
Cm241	96241	1.000000				
Cm242	96242	1.000000				
Cm243	96243	1.000000				
Cm244	96244	1.000000				
Cm245	96245	1.000000				
Cm246	96246	1.000000				
Cm247	96247	1.000000				
Cm248	96248	1.000000				
Cm249	96249	1.000000				
Cm250	96250	1.000000				
Cm251	96251	1.000000				
Bk	97000	1.000000				
Berkelium	97000	1.000000				
Bk245	97245	1.000000				
Bk246	97246	1.000000				
Bk247	97247	1.000000				
Bk248	97248	1.000000				
Bk249	97249	1.000000				
Bk250	97250	1.000000				
Cf	98000	1.000000				
Californium	98000	1.000000				
Cf246	98246	1.000000				
Cf247	98247	1.000000				
Cf248	98248	1.000000				
Cf249	98249	1.000000				
Cf250	98250	1.000000				
Cf251	98251	1.000000				
Cf252	98252	1.000000				
Cf253	98253	1.000000				
Cf254	98254	1.000000				
Es	99000	1.000000				
Einsteinium	99000	1.000000				
FF	99125	1.000000				
FissionF	99125	1.000000				
Es253	99253	1.000000				
Es254	99254	1.000000				
Es255	99255	1.000000				
Fm	100000	1.000000				
Fermium	100000	1.000000				
Fm255	100255	1.000000				
A-150	1000	0.101000	6000	0.777000	7000	0.035000
	8016	0.052000	9000	0.017000	20000	0.018000
TissueEq	1000	0.101000	6000	0.777000	7000	0.035000
	8016	0.052000	9000	0.017000	20000	0.018000
(A-150)	301001	0.101000	-1001	0.101000	6000	0.777000
	7000	0.035000	8016	0.052000	9000	0.017000
	20000	0.018000				
(TissueEq)	301001	0.101000	-1001	0.101000	6000	0.777000
	7000	0.035000	8016	0.052000	9000	0.017000
	20000	0.018000				
Acetone	1000	0.104000	6000	0.621000	8016	0.275000
C3H6O	1000	0.104000	6000	0.621000	8016	0.275000
(Acetone)	201001	0.104000	-1001	0.104000	6000	0.621000
	8016	0.275000				
(C3H6O)	201001	0.104000	-1001	0.104000	6000	0.621000
	8016	0.275000				
Acetylene	1000	0.077000	6000	0.923000		
C2H2	1000	0.077000	6000	0.923000		
(Acetylene)	201001	0.077000	-1001	0.077000	6000	0.923000
(C2H2)	201001	0.077000	-1001	0.077000	6000	0.923000
Adenine	1000	0.037000	6000	0.445000	7000	0.518000
C5H5N5	1000	0.037000	6000	0.445000	7000	0.518000

COG COGLEX Dictionary Listing

10/30/2024

(Adenine)	201001	0.037000	-1001	0.037000	6000	0.445000
	7000	0.518000				
(C5H5N5)	201001	0.037000	-1001	0.037000	6000	0.445000
	7000	0.518000				
Air	7000	0.755000	8016	0.232000	18000	0.013000
Alanine	1000	0.079000	6000	0.405000	7000	0.157000
	8016	0.359000				
C3H7NO2	1000	0.079000	6000	0.405000	7000	0.157000
	8016	0.359000				
(Alanine)	201001	0.079000	-1001	0.079000	6000	0.405000
	7000	0.157000	8016	0.359000		
(C3H7NO2)	201001	0.079000	-1001	0.079000	6000	0.405000
	7000	0.157000	8016	0.359000		
AlOxide	8016	0.471000	13000	0.529000		
Al2O3	8016	0.471000	13000	0.529000		
Amber	1000	0.105000	6000	0.790000	8016	0.105000
C10H16O	1000	0.105000	6000	0.790000	8016	0.105000
(Amber)	201001	0.105000	-1001	0.105000	6000	0.790000
	8016	0.105000				
(C10H16O)	201001	0.105000	-1001	0.105000	6000	0.790000
	8016	0.105000				
Ammonia	1000	0.178000	7000	0.822000		
NH3	1000	0.178000	7000	0.822000		
Aniline	1000	0.078000	6000	0.772000	7000	0.150000
C6H5NH2	1000	0.078000	6000	0.772000	7000	0.150000
(Aniline)	201001	0.078000	-1001	0.078000	6000	0.772000
	7000	0.150000				
(C6H5NH2)	201001	0.078000	-1001	0.078000	6000	0.772000
	7000	0.150000				
Anthracene	1000	0.057000	6000	0.943000		
C14H10	1000	0.057000	6000	0.943000		
(Anthracene)	201001	0.057000	-1001	0.057000	6000	0.943000
(C14H10)	201001	0.057000	-1001	0.057000	6000	0.943000
B-100	1000	0.065000	6000	0.537000	7000	0.022000
	8016	0.032000	9000	0.167000	20000	0.177000
BoneEq	1000	0.065000	6000	0.537000	7000	0.022000
	8016	0.032000	9000	0.167000	20000	0.177000
(B-100)	301001	0.065000	-1001	0.065000	6000	0.537000
	7000	0.022000	8016	0.032000	9000	0.167000
	20000	0.177000				
(BoneEq)	301001	0.065000	-1001	0.065000	6000	0.537000
	7000	0.022000	8016	0.032000	9000	0.167000
	20000	0.177000				
Bakelite	1000	0.057000	6000	0.775000	8016	0.168000
C43H38O7	1000	0.057000	6000	0.775000	8016	0.168000
(Bakelite)	201001	0.057000	-1001	0.057000	6000	0.775000
	8016	0.168000				
(C43H38O)	201001	0.057000	-1001	0.057000	6000	0.775000
	8016	0.168000				
BaF2	9000	0.217000	56138	0.783000		
BariumFluoride	9000	0.217000	56138	0.783000		
BaFluoride	9000	0.217000	56138	0.783000		
BaSO4	8016	0.274000	16032	0.137000	56138	0.589000
BariumSulfate	8016	0.274000	16032	0.137000	56138	0.589000
BaSulfate	8016	0.274000	16032	0.137000	56138	0.589000
Benzene	1001	0.077000	6000	0.923000		
C6H6	1001	0.077000	6000	0.923000		
BeOxide	4000	0.360000	8016	0.640000		
BeO	4000	0.360000	8016	0.640000		
Blood	1000	0.101000	6000	0.100000	7000	0.030000
	8016	0.760000	11000	0.002000	16032	0.002000
	17000	0.003000	19000	0.002000		
(Blood)	301001	0.101000	-1001	0.101000	6000	0.100000
	7000	0.030000	8016	0.760000	11000	0.002000
	16032	0.002000	17000	0.003000	19000	0.002000
BoronCarbide	5000	0.783000	6000	0.217000		
B4C	5000	0.783000	6000	0.217000		
BCarbide	5000	0.783000	6000	0.217000		

COG COGLEX Dictionary Listing

10/30/2024

BoronOxide	5000	0.311000	8016	0.689000		
B2O3	5000	0.311000	8016	0.689000		
BOxide	5000	0.311000	8016	0.689000		
Butane	1000	0.173000	6000	0.827000		
Isobutane	1000	0.173000	6000	0.827000		
C4H10	1000	0.173000	6000	0.827000		
(Butane)	201001	0.173000	-1001	0.173000	6000	0.827000
(Isobuta	201001	0.173000	-1001	0.173000	6000	0.827000
(C4H10)	201001	0.173000	-1001	0.173000	6000	0.827000
ButylAlcohol	1000	0.137000	6000	0.646000	8016	0.217000
C4H9OH	1000	0.137000	6000	0.646000	8016	0.217000
(ButylAlcohol)	201001	0.137000	-1001	0.137000	6000	0.646000
	8016	0.217000				
(C4H9OH)	201001	0.137000	-1001	0.137000	6000	0.646000
	8016	0.217000				
C-552	1000	0.025000	6000	0.501000	8016	0.005000
	9000	0.465000	14000	0.004000		
AirEq	1000	0.025000	6000	0.501000	8016	0.005000
	9000	0.465000	14000	0.004000		
CdTe	48000	0.468000	52000	0.532000		
CadmiumTelluride	48000	0.468000	52000	0.532000		
CdTelluride	48000	0.468000	52000	0.532000		
CdWO4	8016	0.178000	48000	0.312000	74000	0.510000
CadmiumTungstate	8016	0.178000	48000	0.312000	74000	0.510000
CdTungstate	8016	0.178000	48000	0.312000	74000	0.510000
CaCO3	6000	0.120000	8016	0.480000	20000	0.400000
CalciumCarbonate	6000	0.120000	8016	0.480000	20000	0.400000
CaCarbonate	6000	0.120000	8016	0.480000	20000	0.400000
CaF2	9000	0.487000	20000	0.513000		
CalciumFluoride	9000	0.487000	20000	0.513000		
CaFluoride	9000	0.487000	20000	0.513000		
CaO	8016	0.285000	20000	0.715000		
CalciumOxide	8016	0.285000	20000	0.715000		
CaOxide	8016	0.285000	20000	0.715000		
CaSO4	8016	0.471000	16032	0.235000	20000	0.294000
CalciumSulfate	8016	0.471000	16032	0.235000	20000	0.294000
CaSulfate	8016	0.471000	16032	0.235000	20000	0.294000
CaWO4	8016	0.222000	20000	0.139000	74000	0.639000
CalciumTungstate	8016	0.222000	20000	0.139000	74000	0.639000
CaTungstate	8016	0.222000	20000	0.139000	74000	0.639000
CO2	6000	0.273000	8016	0.727000		
CarbonDioxide	6000	0.273000	8016	0.727000		
CDioxide	6000	0.273000	8016	0.727000		
CCl4	6000	0.078000	17000	0.922000		
CarbonTetraCl	6000	0.078000	17000	0.922000		
Cellophane	1000	0.062000	6000	0.445000	8016	0.493000
C6H10O5	1000	0.062000	6000	0.445000	8016	0.493000
(Cellophane)	201001	0.062000	-1001	0.062000	6000	0.445000
	8016	0.493000				
(C6H10O5)	201001	0.062000	-1001	0.062000	6000	0.445000
	8016	0.493000				
CAB	1000	0.067000	6000	0.545000	8016	0.388000
C15H22O8	1000	0.067000	6000	0.545000	8016	0.388000
(CAB)	201001	0.067000	-1001	0.067000	6000	0.545000
	8016	0.388000				
(C15H22O8)	201001	0.067000	-1001	0.067000	6000	0.545000
	8016	0.388000				
CsF	9000	0.125000	55000	0.875000		
CesiumFluoride	9000	0.125000	55000	0.875000		
CsFluoride	9000	0.125000	55000	0.875000		
CsI	53000	0.489000	55000	0.511000		
CesiumIodide	53000	0.489000	55000	0.511000		
CsIodide	53000	0.489000	55000	0.511000		
Chloroform	1000	0.008000	6000	0.101000	17000	0.891000
ChCl3	1000	0.008000	6000	0.101000	17000	0.891000
Cncrtl	1000	0.005500	8016	0.485300	11000	0.025700
	12000	0.015000	13000	0.042900	14000	0.331400

COG COGLEX Dictionary Listing

10/30/2024

	20000	0.060100	26000	0.034100		
LLNL.con	1000	0.005500	8016	0.485300	11000	0.025700
	12000	0.015000	13000	0.042900	14000	0.331400
	20000	0.060100	26000	0.034100		
(Cncrt1)	301001	0.005500	-1001	0.005500	8016	0.485300
	11000	0.025700	12000	0.015000	13000	0.042900
	14000	0.331400	20000	0.060100	26000	0.034100
(LLNL.con)	301001	0.005500	-1001	0.005500	8016	0.485300
	11000	0.025700	12000	0.015000	13000	0.042900
	14000	0.331400	20000	0.060100	26000	0.034100
Cncrt2	1000	0.005000	8016	0.435400	11000	0.018300
	12000	0.013400	13000	0.006200	14000	0.301300
	20000	0.189600	26000	0.030800		
ORNL.con	1000	0.005000	8016	0.435400	11000	0.018300
	12000	0.013400	13000	0.006200	14000	0.301300
	20000	0.189600	26000	0.030800		
(Cncrt2)	301001	0.005000	-1001	0.005000	8016	0.435400
	11000	0.018300	12000	0.013400	13000	0.006200
	14000	0.301300	20000	0.189600	26000	0.030800
(ORNL.con)	301001	0.005000	-1001	0.005000	8016	0.435400
	11000	0.018300	12000	0.013400	13000	0.006200
	14000	0.301300	20000	0.189600	26000	0.030800
Cncrt3	1000	0.003900	6000	0.013000	8016	0.509100
	11000	0.014500	12000	0.012000	13000	0.050000
	14000	0.305000	19000	0.001500	20000	0.062000
	26000	0.029000				
MFE.conc	1000	0.003900	6000	0.013000	8016	0.509100
	11000	0.014500	12000	0.012000	13000	0.050000
	14000	0.305000	19000	0.001500	20000	0.062000
	26000	0.029000				
(Cncrt3)	301001	0.003900	-1001	0.003900	6000	0.013000
	8016	0.509100	11000	0.014500	12000	0.012000
	13000	0.050000	14000	0.305000	19000	0.001500
	20000	0.062000	26000	0.029000		
(MFE.conc)	301001	0.003900	-1001	0.003900	6000	0.013000
	8016	0.509100	11000	0.014500	12000	0.012000
	13000	0.050000	14000	0.305000	19000	0.001500
	20000	0.062000	26000	0.029000		
Cncrt4	1001	0.003585	8016	0.311622	12000	0.001195
	13027	0.004183	14000	0.010457	16032	0.107858
	20000	0.050194	26000	0.047505	56138	0.463400
Barite.c	1001	0.003585	8016	0.311622	12000	0.001195
	13027	0.004183	14000	0.010457	16032	0.107858
	20000	0.050194	26000	0.047505	56138	0.463400
(Cncrt4)	301001	0.003585	-1001	0.003585	8016	0.311622
	12000	0.001195	13027	0.004183	14000	0.010457
	16032	0.107858	20000	0.050194	26000	0.047505
	56138	0.463400				
(Barite.c)	301001	0.003585	-1001	0.003585	8016	0.311622
	12000	0.001195	13027	0.004183	14000	0.010457
	16032	0.107858	20000	0.050194	26000	0.047505
	56138	0.463400				
Cncrt5	1001	0.004530	8016	0.512600	11023	0.015270
	13027	0.003555	14000	0.360360	20000	0.057910
	26000	0.013780				
MCNP.con	1001	0.004530	8016	0.512600	11023	0.015270
	13027	0.003555	14000	0.360360	20000	0.057910
	26000	0.013780				
(Cncrt5)	301001	0.004530	-1001	0.004530	8016	0.512600
	11023	0.015270	13027	0.003555	14000	0.360360
	20000	0.057910	26000	0.013780		
(MCNP.con)	301001	0.004530	-1001	0.004530	8016	0.512600
	11023	0.015270	13027	0.003555	14000	0.360360
	20000	0.057910	26000	0.013780		
Cncrt6	1001	0.003113	8016	0.330504	12000	0.009338
	13027	0.023486	14000	0.025750	16000	0.001415
	20000	0.071024	22000	0.054329	23000	0.003113
	24000	0.001698	25055	0.001981	26000	0.474250

COG COGLEX Dictionary Listing

10/30/2024

Mag.conc	1001	0.003113	8016	0.330504	12000	0.009338
	13027	0.023486	14000	0.025750	16000	0.001415
	20000	0.071024	22000	0.054329	23000	0.003113
	24000	0.001698	25055	0.001981	26000	0.474250
(Cncrt6)	301001	0.003113	-1001	0.003113	8016	0.330504
	12000	0.009338	13027	0.023486	14000	0.025750
	16000	0.001415	20000	0.071024	22000	0.054329
	23000	0.003113	24000	0.001698	25055	0.001981
	26000	0.474250				
(Mag.conc)	301001	0.003113	-1001	0.003113	8016	0.330504
	12000	0.009338	13027	0.023486	14000	0.025750
	16000	0.001415	20000	0.071024	22000	0.054329
	23000	0.003113	24000	0.001698	25055	0.001981
	26000	0.474250				
Cncrt7	1001	0.008485	6000	0.050064	8016	0.473483
	12000	0.024183	13027	0.036063	14000	0.145100
	16000	0.002970	19000	0.001697	20000	0.246924
	26000	0.011031				
NBS03.conc	1001	0.008485	6000	0.050064	8016	0.473483
	12000	0.024183	13027	0.036063	14000	0.145100
	16000	0.002970	19000	0.001697	20000	0.246924
	26000	0.011031				
(Cncrt7)	301001	0.008485	-1001	0.008485	6000	0.050064
	8016	0.473483	12000	0.024183	13027	0.036063
	14000	0.145100	16000	0.002970	19000	0.001697
	20000	0.246924	26000	0.011031		
(NBS03.conc)	301001	0.008485	-1001	0.008485	6000	0.050064
	8016	0.473483	12000	0.024183	13027	0.036063
	14000	0.145100	16000	0.002970	19000	0.001697
	20000	0.246924	26000	0.011031		
CrOxide	8016	0.316000	24000	0.684000		
Cr2O3	8016	0.316000	24000	0.684000		
CuIodide	29000	0.333000	53000	0.667000		
CopperIodide	29000	0.333000	53000	0.667000		
CuI	29000	0.333000	53000	0.667000		
CuOxide	8016	0.201000	29000	0.799000		
CopperOxide	8016	0.201000	29000	0.799000		
CuO	8016	0.201000	29000	0.799000		
Cyclohexane	1000	0.143800	6000	0.856200		
C6H12	1000	0.143800	6000	0.856200		
(Cyclohexane)	201001	0.143800	-1001	0.143800	6000	0.856200
(C6H12)	201001	0.143800	-1001	0.143800	6000	0.856200
Ethane	1000	0.201100	6000	0.798900		
C2H6	1000	0.201100	6000	0.798900		
(Ethane)	201001	0.201100	-1001	0.201100	6000	0.798900
(C2H6)	201001	0.201100	-1001	0.201100	6000	0.798900
EthylAlcohol	1000	0.131300	6000	0.521400	8016	0.347300
C2H5OH	1000	0.131300	6000	0.521400	8016	0.347300
(EthylAlcohol)	201001	0.131300	-1001	0.131300	6000	0.521400
	8016	0.347300				
(C2H5OH)	201001	0.131300	-1001	0.131300	6000	0.521400
	8016	0.347300				
Ethylene	1000	0.143700	6000	0.856300		
C2H4	1000	0.143700	6000	0.856300		
(Ethylene)	201001	0.143700	-1001	0.143700	6000	0.856300
(C2H4)	201001	0.143700	-1001	0.143700	6000	0.856300
EyeLens	1000	0.099300	6000	0.193700	7000	0.053300
	8016	0.653700				
(EyeLens)	301001	0.099300	-1001	0.099300	6000	0.193700
	7000	0.053300	8016	0.653700		
FerrousOxide	8016	0.222700	26000	0.777300		
FeO	8016	0.222700	26000	0.777300		
FerricOxide	8016	0.300600	26000	0.699400		
Fe2O3	8016	0.300600	26000	0.699400		
Magnetite	8016	0.276400	26000	0.723600		
Fe3O4	8016	0.276400	26000	0.723600		
FiberGlass	1000	0.001400	6000	0.159000	7000	0.013600
	8016	0.343000	12000	0.052000	13000	0.127000

COG COGLEX Dictionary Listing

10/30/2024

	14000	0.304000				
FG	1000	0.001400	6000	0.159000	7000	0.013600
	8016	0.343000	12000	0.052000	13000	0.127000
	14000	0.304000				
(FiberGlass)	201001	0.001400	-1001	0.001400	6000	0.159000
	7000	0.013600	8016	0.343000	12000	0.052000
	13000	0.127000	14000	0.304000		
(FG)	201001	0.001400	-1001	0.001400	6000	0.159000
	7000	0.013600	8016	0.343000	12000	0.052000
	13000	0.127000	14000	0.304000		
Formvar	1000	0.074000	6000	0.558000	8016	0.368000
C10H16O5	1000	0.074000	6000	0.558000	8016	0.368000
(Formvar)	201001	0.074000	-1001	0.074000	6000	0.558000
	8016	0.368000				
(C10H16O5)	201001	0.074000	-1001	0.074000	6000	0.558000
	8016	0.368000				
Freon12	6000	0.099300	9000	0.314200	17000	0.586500
CF2Cl2	6000	0.099300	9000	0.314200	17000	0.586500
Freon12B	6000	0.057200	9000	0.181100	35000	0.761700
CF2Br2	6000	0.057200	9000	0.181100	35000	0.761700
Freon13	6000	0.115000	9000	0.545600	17000	0.339400
CF3Cl	6000	0.115000	9000	0.545600	17000	0.339400
Freon13B	6000	0.080700	9000	0.382700	35000	0.536600
CF3Br	6000	0.080700	9000	0.382700	35000	0.536600
Freon13I	6000	0.061300	9000	0.290900	53000	0.647800
CF3I	6000	0.061300	9000	0.290900	53000	0.647800
GalliumArsenide	31000	0.482000	33000	0.518000		
GaAs	31000	0.482000	33000	0.518000		
GaArsenide	31000	0.482000	33000	0.518000		
Glass	8016	0.459800	11000	0.096400	14000	0.336600
	20000	0.107200				
Heptane	1000	0.160900	6000	0.839100		
C7H16	1000	0.160900	6000	0.839100		
(Heptane)	201001	0.160900	-1001	0.160900	6000	0.839100
(C7H16)	201001	0.160900	-1001	0.160900	6000	0.839100
Hexane	1000	0.163700	6000	0.836300		
C6H14	1000	0.163700	6000	0.836300		
(Hexane)	201001	0.163700	-1001	0.163700	6000	0.836300
(C6H14)	201001	0.163700	-1001	0.163700	6000	0.836300
HgIodide	53000	0.558600	80000	0.441400		
HgI2	53000	0.558600	80000	0.441400		
LX040	1000	0.026000	6000	0.186000	7000	0.322000
	8016	0.367000	9000	0.099000		
LX041	1000	0.026000	6000	0.186000	7000	0.322000
	8016	0.367000	9000	0.099000		
(LX040)	201001	0.026000	-1001	0.026000	6000	0.186000
	7000	0.322000	8016	0.367000	9000	0.099000
(LX041)	201001	0.026000	-1001	0.026000	6000	0.186000
	7000	0.322000	8016	0.367000	9000	0.099000
LX07	1000	0.026000	6000	0.178000	7000	0.340000
	8016	0.389000	9000	0.067000		
(LX07)	201001	0.026000	-1001	0.026000	6000	0.178000
	7000	0.340000	8016	0.389000	9000	0.067000
LX09	1000	0.028000	6000	0.172000	7000	0.363000
	8016	0.414000	9000	0.023000		
(LX09)	201001	0.028000	-1001	0.028000	6000	0.172000
	7000	0.363000	8016	0.414000	9000	0.023000
LX10	1000	0.027000	6000	0.169000	7000	0.359000
	8016	0.411000	9000	0.034000		
(LX10)	201001	0.027000	-1001	0.027000	6000	0.169000
	7000	0.359000	8016	0.411000	9000	0.034000
PBX9404	1000	0.028000	6000	0.168000	7000	0.360000
	8016	0.430000	15000	0.003000	17000	0.011000
PBX9409	1000	0.028000	6000	0.168000	7000	0.360000
	8016	0.430000	15000	0.003000	17000	0.011000
(PBX9404)	201001	0.028000	-1001	0.028000	6000	0.168000
	7000	0.360000	8016	0.430000	15000	0.003000
	17000	0.011000				

COG COGLEX Dictionary Listing

10/30/2024

MockHE	1000	0.034000	6000	0.313000	7000	0.308000
	8016	0.345000				
(MockHE)	201001	0.034000	-1001	0.034000	6000	0.313000
	7000	0.308000	8016	0.345000		
Lexan	1000	0.055000	6000	0.756000	8016	0.189000
C16H14O3	1000	0.055000	6000	0.756000	8016	0.189000
(Lexan)	201001	0.055000	-1001	0.055000	6000	0.756000
	8016	0.189000				
(C16H14)	201001	0.055000	-1001	0.055000	6000	0.756000
	8016	0.189000				
LiCO3	3000	0.187800	6000	0.162600	8016	0.649600
LiH	1000	0.127000	3000	0.873000		
LithiumHydride	1000	0.127000	3000	0.873000		
LiD	1002	0.249000	3000	0.751000		
LithiumDeuteride	1002	0.249000	3000	0.751000		
Li6H	1000	0.142000	3006	0.858000		
Li7H	1000	0.126000	3007	0.874000		
LiF	3000	0.268000	9000	0.732000		
LithiumFluoride	3000	0.268000	9000	0.732000		
LiI	3000	0.051900	53000	0.948100		
LithiumIodide	3000	0.051900	53000	0.948100		
Li2O	3000	0.464600	8016	0.535400		
LithiumDioxide	3000	0.464600	8016	0.535400		
Lucite	1000	0.080000	6000	0.600000	8016	0.320000
Perspex	1000	0.080000	6000	0.600000	8016	0.320000
C5H8O2	1000	0.080000	6000	0.600000	8016	0.320000
(Lucite)	1001001	0.080000	-1001	0.080000	6000	0.600000
	8016	0.320000				
(Perspex)	1001001	0.080000	-1001	0.080000	6000	0.600000
	8016	0.320000				
(C5H8O2)	1001001	0.080000	-1001	0.080000	6000	0.600000
	8016	0.320000				
MgCO3	6000	0.142500	8016	0.569200	12000	0.288300
MgF2	9000	0.609900	12000	0.390100		
MgOxide	8016	0.397000	12000	0.603000		
MgO	8016	0.397000	12000	0.603000		
Methane	1000	0.251300	6000	0.748700		
CH4	1000	0.251300	6000	0.748700		
(Methane)	301001	0.251300	-1001	0.251300	6000	0.748700
(CH4)	301001	0.251300	-1001	0.251300	6000	0.748700
Methanol	1000	0.125800	6000	0.374800	8016	0.499400
CH3OH	1000	0.125800	6000	0.374800	8016	0.499400
(Methanol)	301001	0.125800	-1001	0.125800	6000	0.374800
	8016	0.499400				
(CH3OH)	301001	0.125800	-1001	0.125800	6000	0.374800
	8016	0.499400				
Mylar	1000	0.042000	6000	0.625000	8016	0.333000
C10H8O4	1000	0.042000	6000	0.625000	8016	0.333000
(Mylar)	201001	0.042000	-1001	0.042000	6000	0.625000
	8016	0.333000				
(C10H8O4)	201001	0.042000	-1001	0.042000	6000	0.625000
	8016	0.333000				
Napthalene	1000	0.062900	6000	0.937100		
C10H8	1000	0.062900	6000	0.937100		
(Napthalene)	201001	0.062900	-1001	0.062900	6000	0.937100
(C10H8)	201001	0.062900	-1001	0.062900	6000	0.937100
NiOxide	8016	0.214000	28000	0.786000		
NickelOxide	8016	0.214000	28000	0.786000		
NiO	8016	0.214000	28000	0.786000		
Nitrobenzene	1000	0.040900	6000	0.585400	7000	0.113800
	8016	0.259900				
C6H5NO2	1000	0.040900	6000	0.585400	7000	0.113800
	8016	0.259900				
(Nitrobenzene)	201001	0.040900	-1001	0.040900	6000	0.585400
	7000	0.113800	8016	0.259900		
(C6H5NO2)	201001	0.040900	-1001	0.040900	6000	0.585400
	7000	0.113800	8016	0.259900		
N2O	7000	0.636500	8016	0.363500		

COG COGLEX Dictionary Listing

10/30/2024

Octane	1000	0.158800	6000	0.841200		
C8H18	1000	0.158800	6000	0.841200		
(Octane)	301001	0.158800	-1001	0.158800	6000	0.841200
(C8H18)	301001	0.158800	-1001	0.158800	6000	0.841200
Oy	92235	0.932000	92238	0.068000		
Oralloy	92235	0.932000	92238	0.068000		
Paraffin	1000	0.148600	6000	0.851400		
C25H52	1000	0.148600	6000	0.851400		
(Paraffin)	201001	0.148600	-1001	0.148600	6000	0.851400
(C25H52)	201001	0.148600	-1001	0.148600	6000	0.851400
Paryl-Chloride	1000	0.035000	6000	0.555000	17000	0.410000
C4H3Cl	1000	0.035000	6000	0.555000	17000	0.410000
(Paryl-Chloride)	201001	0.035000	-1001	0.035000	6000	0.555000
	17000	0.410000				
(C4H3Cl)	201001	0.035000	-1001	0.035000	6000	0.555000
	17000	0.410000				
Paryl-N	1000	0.129000	6000	0.871000		
C8H8	1000	0.129000	6000	0.871000		
(Paryl-N	201001	0.129000	-1001	0.129000	6000	0.871000
(C8H8)	201001	0.129000	-1001	0.129000	6000	0.871000
Pentane	1000	0.167600	6000	0.832400		
C5H12	1000	0.167600	6000	0.832400		
(Pentane	201001	0.167600	-1001	0.167600	6000	0.832400
(C5H12)	201001	0.167600	-1001	0.167600	6000	0.832400
Polyethy	1000	0.144000	6000	0.856000		
CH2	1000	0.144000	6000	0.856000		
Polyform	1000	0.067000	6000	0.400000	8016	0.533000
CH2O	1000	0.067000	6000	0.400000	8016	0.533000
(Polyfor	201001	0.067000	-1001	0.067000	6000	0.400000
	8016	0.533000				
(CH2O)	201001	0.067000	-1001	0.067000	6000	0.400000
	8016	0.533000				
Polypropolene	1000	0.123000	6000	0.877000		
C3H6	1000	0.123000	6000	0.877000		
(Polypropolene)	201001	0.123000	-1001	0.123000	6000	0.877000
(C3H6)	201001	0.123000	-1001	0.123000	6000	0.877000
Polystyrene	1000	0.077000	6000	0.923000		
CH	1000	0.077000	6000	0.923000		
(Polystyrene)	201001	0.077000	-1001	0.077000	6000	0.923000
(CH)	201001	0.077000	-1001	0.077000	6000	0.923000
PVA	1000	0.070000	6000	0.558000	8016	0.372000
C2H3O	1000	0.070000	6000	0.558000	8016	0.372000
(PVA)	201001	0.070000	-1001	0.070000	6000	0.558000
	8016	0.372000				
(C2H3O)	201001	0.070000	-1001	0.070000	6000	0.558000
	8016	0.372000				
PVB	1000	0.091600	6000	0.681500	8016	0.226900
C8H13O2	1000	0.091600	6000	0.681500	8016	0.226900
(PVB)	201001	0.091600	-1001	0.091600	6000	0.681500
	8016	0.226900				
(C8H13O2)	201001	0.091600	-1001	0.091600	6000	0.681500
	8016	0.226900				
PVC	1000	0.048000	6000	0.384000	17000	0.568000
C2H3Cl	1000	0.048000	6000	0.384000	17000	0.568000
(PVC)	201001	0.048000	-1001	0.048000	6000	0.384000
	17000	0.568000				
(C2H3Cl)	201001	0.048000	-1001	0.048000	6000	0.384000
	17000	0.568000				
PVF	1000	0.031500	6000	0.375100	9000	0.593400
C2H2F2	1000	0.031500	6000	0.375100	9000	0.593400
(PVF)	201001	0.031500	-1001	0.031500	6000	0.375100
	9000	0.593400				
(C2H2F2)	201001	0.031500	-1001	0.031500	6000	0.375100
	9000	0.593400				
Pyrex	5000	0.040000	8016	0.539700	11023	0.028200
	13027	0.011600	14000	0.377200	19000	0.003300
Corning7	5000	0.040000	8016	0.539700	11023	0.028200
	13027	0.011600	14000	0.377200	19000	0.003300

COG COGLEX Dictionary Listing

10/30/2024

Scintillator	1000	0.085000	6000	0.915000		
NE110	1000	0.085000	6000	0.915000		
NE111	1000	0.085000	6000	0.915000		
NE113	1000	0.085000	6000	0.915000		
NE114	1000	0.085000	6000	0.915000		
PilotB	1000	0.085000	6000	0.915000		
PilotF	1000	0.085000	6000	0.915000		
PilotU	1000	0.085000	6000	0.915000		
PilotY	1000	0.085000	6000	0.915000		
Teflon	6000	0.240000	9000	0.760000		
CF2	6000	0.240000	9000	0.760000		
Saran	1000	0.021000	6000	0.248000	17000	0.731000
C2H2Cl2	1000	0.021000	6000	0.248000	17000	0.731000
(Saran)	201001	0.021000	-1001	0.021000	6000	0.248000
	17000	0.731000				
(C2H2Cl2)	201001	0.021000	-1001	0.021000	6000	0.248000
	17000	0.731000				
KI	19000	0.235500	53000	0.764500		
K2O	8016	0.169900	19000	0.830100		
Propane	1000	0.182900	6000	0.817100		
C3H8	1000	0.182900	6000	0.817100		
(Propane)	201001	0.182900	-1001	0.182900	6000	0.817100
(C3H8)	201001	0.182900	-1001	0.182900	6000	0.817100
Pyridine	1000	0.063700	6000	0.759200	7000	0.177100
C5H5N	1000	0.063700	6000	0.759200	7000	0.177100
(Pyridin	201001	0.063700	-1001	0.063700	6000	0.759200
	7000	0.177100				
(C5H5N)	201001	0.063700	-1001	0.063700	6000	0.759200
	7000	0.177100				
SiO2	8016	0.533000	14000	0.467000		
Quartz	8016	0.533000	14000	0.467000		
Rubber1	1000	0.143700	6000	0.856300		
C4H8	1000	0.143700	6000	0.856300		
(Rubber1)	201001	0.143700	-1001	0.143700	6000	0.856300
(C4H8)	201001	0.143700	-1001	0.143700	6000	0.856300
Rubber2	1000	0.118400	6000	0.881600		
C5H8	1000	0.118400	6000	0.881600		
(Rubber2)	201001	0.118400	-1001	0.118400	6000	0.881600
(C5H8)	201001	0.118400	-1001	0.118400	6000	0.881600
Rubber3	1000	0.056900	6000	0.542700	17000	0.400400
C4H5Cl	1000	0.056900	6000	0.542700	17000	0.400400
(Rubber3)	201001	0.056900	-1001	0.056900	6000	0.542700
	17000	0.400400				
(C4H5Cl)	201001	0.056900	-1001	0.056900	6000	0.542700
	17000	0.400400				
SilverBromide	35000	0.425500	47000	0.574500		
AgBromide	35000	0.425500	47000	0.574500		
AgBr	35000	0.425500	47000	0.574500		
SilverChloride	17000	0.247400	47000	0.752600		
AgChloride	17000	0.247400	47000	0.752600		
AgCl	17000	0.247400	47000	0.752600		
SilverIodide	47000	0.459500	53000	0.540500		
AgIodide	47000	0.459500	53000	0.540500		
AgI	47000	0.459500	53000	0.540500		
Na2CO3	6000	0.113300	8016	0.452900	11000	0.433800
SodiumIodide	11000	0.153000	53000	0.847000		
NaI	11000	0.153000	53000	0.847000		
Na2O	8016	0.258100	11000	0.741900		
NaNO3	7000	0.164800	8016	0.564700	11000	0.270500
Steel	6000	0.000400	14000	0.004600	24000	0.190000
	25000	0.009000	26000	0.699000	28000	0.097000
SS304	6000	0.000400	14000	0.004600	24000	0.190000
	25000	0.009000	26000	0.699000	28000	0.097000
Stilbene	1000	0.067100	6000	0.932900		
C14H12	1000	0.067100	6000	0.932900		
(Stilbene)	201001	0.067100	-1001	0.067100	6000	0.932900
(C14H12)	201001	0.067100	-1001	0.067100	6000	0.932900
Terphenyl	1000	0.044500	6000	0.955500		

COG COGLEX Dictionary Listing

10/30/2024

C18H10	1000	0.044500	6000	0.955500		
(Terphenyl)	201001	0.044500	-1001	0.044500	6000	0.955500
(C18H10)	201001	0.044500	-1001	0.044500	6000	0.955500
TiO2	8016	0.400600	22000	0.599400		
Toluene	1000	0.087500	6000	0.912500		
C7H8	1000	0.087500	6000	0.912500		
(Toluene)	201001	0.087500	-1001	0.087500	6000	0.912500
(C7H8)	201001	0.087500	-1001	0.087500	6000	0.912500
WF6	9000	0.382700	74000	0.617300		
Water	1000	0.112000	8016	0.888000		
H2O	1000	0.112000	8016	0.888000		
(Water)	301001	0.112000	-1001	0.112000	8016	0.888000
(H2O)	301001	0.112000	-1001	0.112000	8016	0.888000
D2O	1002	0.201000	8016	0.799000		
HeavyWater	1002	0.201000	8016	0.799000		
Xylene	1000	0.094900	6000	0.905100		
C8H10	1000	0.094900	6000	0.905100		
(Xylene)	201001	0.094900	-1001	0.094900	6000	0.905100
(C8H10)	201001	0.094900	-1001	0.094900	6000	0.905100
ZrH	1000	0.011000	40000	0.989000		
UO2	8016	0.118000	92000	0.882000		
(UO2)	308016	1.000000	-8016	0.118000	-92238	0.882000
U233O2	8016	0.121000	92233	0.879000		
U234O2	8016	0.120000	92234	0.880000		
U235O2	8016	0.120000	92235	0.880000		
U236O2	8016	0.119000	92236	0.881000		
U237O2	8016	0.119000	92237	0.881000		
U238O2	8016	0.119000	92238	0.881000		
U239O2	8016	0.118000	92239	0.882000		
U240O2	8016	0.118000	92240	0.882000		
D38	92235	0.007000	92238	0.993000		
Tuballoy	92235	0.007000	92238	0.993000		
binary	41000	0.064000	92000	0.936000		
ABF	99997	1.000000				
ABc	99998	1.000000				
ABS	99999	1.000000				

23 COG Reaction Number Listing

This is a list of the reactions that a particle can undergo during COG transport.. Knowledge of the numbers assigned to reactions is generally needed only if the user wishes to use a Reaction Rate (**R-RATE**) or Dosimetry (**IRDF-R-R**) Detector Response Function, or uses the **WALK-REACTION** random walk modification.

Reaction.	Number	Reaction	Number
(n,n)	10	(n,XHe3)	53
(n,n γ)	11	(n,X α)	54
(n,2n γ)	12	(n,X γ)	55
(n,3n γ)	13	(n,Xn)	56
(n,4n γ)	14	(n,Xe)	57
(n,fission)	15	(n,2p γ)	67
(n,X)	16	(n,pd γ)	68
(n,3np γ)	17	Rayleigh	71
(n,n2p γ)	18	Compton	72
(n,n3 α /3 α)	19	Photoelectric	73
(n,np γ)	20	Pair prod	74
(n,pn γ)	21	Triplet prod	75
(n,nd γ)	22	(γ ,n)	76
(n,nd $\alpha\gamma$)	23	(g,X)	77
(n,nt γ)	24	(γ ,2n)	78
(n,nHe3 γ)	25	(γ ,3n)	79
(n,na γ)	26	(γ ,na)	80
(n,n2a γ)	27	(γ ,np)	81
(n,nta γ)	28	tot (n,p γ)	82
(n,2np γ)	29	tot (n,d γ)	83
(n,n $\alpha\gamma$)	30	tot (n,t γ)	84
(n,2np $\alpha\gamma$)	31	tot (n,He3 γ)	85
(n,2nd γ)	32	tot (n, $\alpha\gamma$)	86
(n,2n $\alpha\gamma$)	33	tot (n,n γ)	87
(n,np $\alpha\gamma$)	34	(γ ,n2 α)	88
(n,nd α /nd2 α)	35	(γ ,fission)	89
(n,2n2 α /nt2 α)	36	(γ ,Xn)	90
(n,2 $\alpha\gamma$ /3 $\alpha\gamma$)	37	(γ ,Xp)	91

COG Reaction Number Listing**10/30/2024**

(n,He3 $\alpha\gamma$)	38	(γ ,p)	92
(n,tp γ)	39	(d,n γ)	93
(n,p γ)	40	(d,n $\alpha\gamma$)	94
(n,d γ)	41	(d,p γ)	95
(n,t γ)	42	tot (α ,n γ)	100
(n,t $\alpha\gamma$)	43	(α ,X)	101
(n,He3 γ)	44	(α ,2n γ)	102
(n, $\alpha\gamma$)	45	(α ,n $\alpha\gamma$)	103
(n, γ)	46	(α ,np γ)	104
(n,d $\alpha\gamma$ /d2 α)	47	(α ,n γ)	105
(n,p $\alpha\gamma$)	48	(α ,Xn)	106
(n,2p $\alpha\gamma$)	49	(d,d)	107
(n,Xp)	50	(d, $\alpha\gamma$)	108
(n,Xd)	51	(d,t γ)	109
(n,Xt)	52	(d,X)	110

24 Index

A

Accuracy of Results, 333
 ACTIVATION Data Block, 302
 ACTIVATION, 27, 33, 195, 302, 303, 304
 ACTIVATION Data Block Transport, 302
 ALPHA Transport Data Block
 adestep, 354
 aecut, 354
 afrac, 355
 alphatrans, 354
 astepmax, 354
 astepmin, 354
 astragflag, 354
 matat, 354, 356
 ANALYSIS Data Block
 Calculated Detector IMPORTANCE, 217, 227
 Plots of Collision Sites, 215
 ASSIGN Data Block
 ASSIGN-D, 69
 Assigning Physical Properties to Sectors, 25, 45, 67,
 73, 74, 366, 367
 ASSIGN-M, 67
 ASSIGN-MC, 30, 34, 72
 ASSIGN-MD, 68
 ASSIGN-ML, 69
 ASSIGN-R, 70, 71
 ASSIGN-RL, 71

B

BASIC Data Block, 18, 24, 41, 42, 43, 44, 45, 46, 47, 48,
 49, 50, 51, 52, 57, 59, 71, 82, 83, 93, 108, 114, 125,
 127, 128, 129, 130, 131, 132, 155, 160, 169, 171, 173,
 176, 178, 179, 181, 205, 208, 214, 216, 232, 234, 236,
 238, 240, 242, 243, 244, 245, 250, 258, 259,
 291, 308, 336, 346, 366, 378, 393, 401, 404
 Example, 24, 52
 Overview, 24
 Parameters
 ALPHA, 41, 52, 108, 297, 298
 ANNIH-GAMMAS, 53
 BBOUNDCHECK, 51, 345, 347, 351
 CRITDETVR, 46, 53
 DEDX PROTON, 51, 53, 347
 Default Values, 41, 42
 DEUTERON, 41, 52, 108
 DNEAR, 51, 53, 337, 340, 347, 352

ELECTRON, 41
 EPFILE, 51
 ESTEP PROTON, 51, 53, 347
 ESTRAGGLE PROTON, 51, 53, 347, 352
 FRACNX, 53
 FREYA, 50
 HADRONIC PROTON MODEL, 51, 53, 346
 MAXFISSION, 49
 Miscellaneous Parameters, 80
 MSCAT PROTON, 51, 53, 346, 352
 NDUMP, 18, 19, 44, 53
 NEUTRON, 41
 NEUTRONPRODUCTION, 48
 NOFISSION, 49
 NRF, 50, 51, 53
 Number of Secondary Particles Exiting a Collision,
 48
 Particle Type, 41
 PHOTON, 41
 PHOTON PRODUCTION, 48
 PHOTONUCLEAR, 12, 41, 46, 53
 POI, 50
 Proton Physics Options, 51
 RN-Starting Random Number Seeds, 44, 53, 83,
 308, 340, 352
 TDUMP, 18, 19, 43, 44, 53
 URRPT, 50, 53
 Writing Production Data Files, 51
 Parameters
 DELAYEDNEUTRONS, 41, 47, 53, 291
 TMAX, 43, 52, 53, 212, 214
 PHOTONUCLEAR, 12, 41, 46, 53
 BFIELD Data Block
 Magnetic Field Sectors, 345, 350
 PBAXIS, 350, 351
 PMAXIS, 350, 351
 PMORIGIN, 350

C

COG Reaction Number Listing, 438
 COGLEX11.2 Dictionary Listing, 407
 CRITICALITY Data Block, 291, 292, 293, 296, 311, 316,
 317, 394, 402
 ALPHA DYNAMIC, 297
 ALPHA EIGENVALUE, 297
 ALPHA NEW, 297, 298
 DUMP, 27, 33, 299, 300, 301
 NBATCH, 292, 300
 NFIRST, 291, 292, 293, 300, 301, 316, 317

NORM, 292, 293, 300, 316, 317

NPART, 80, 81, 83, 85, 89, 151, 153, 154, 158, 159,
160, 161, 189, 262, 268, 273, 284, 286, 292, 293,
295, 300, 303, 316, 317, 353

NSOURCE, 293, 300, 316, 317

Parameters, 292

RESTART, 20, 299

SDT, 292, 293, 300, 316, 317

D

Data Blocks, 24

ACTIVATION, 27, 33, 195, 302, 303, 304

ALPHATRANSPORT, 27, 33

ANALYSIS, 9, 20, 27, 33, 70, 93, 215, 216, 263, 264,
267, 268, 273, 274, 280, 283, 285, 288, 321

ASSIGN, 25, 26, 33, 36, 37, 57, 58, 10, 20, 30, 31, 34,
36, 45, 63, 67, 68, 69, 70, 71, 72, 73, 74, 75, 171,
261, 262, 271, 273, 283, 285, 303, 340, 353, 366,
367

BASIC, 15, 18, 21, 24, 33, 37, 41, 42, 43, 44, 45, 46,
47, 48, 49, 50, 51, 52, 57, 59, 71, 39, 48, 60, 82, 83,
86, 93, 108, 110, 112, 114, 115, 116, 117, 118, 119,
121, 125, 127, 128, 129, 130, 131, 132, 136, 139,
155, 157, 160, 169, 171, 173, 176, 178, 179, 181,
191, 196, 204, 205, 208, 214, 216, 220, 232, 234,
236, 238, 240, 242, 243, 244, 245, 249, 250, 256,
258, 259, 260, 262, 270, 273, 283, 285, 291, 294,
302, 308, 336, 337, 340, 346, 352, 366, 378, 381,
393, 400, 401, 404

BFIELD, 28, 33, 345, 347, 350, 351

CRITICALITY, 17, 27, 33, 76, 291, 292, 293, 294,
295, 296, 297, 299, 300, 311, 316, 317, 393, 394,
400, 402

DETECTOR, 9, 10, 14, 15, 20, 21, 26, 33, 35, 37, 38,
55, 13, 60, 70, 81, 90, 143, 166, 168, 175, 177, 180,
184, 187, 188, 189, 209, 210, 211, 213, 214, 215,
220, 221, 222, 227, 261, 262, 271, 273, 274, 284,
285, 291, 302, 303, 321, 322, 326, 341, 353, 367,
381, 388, 391

DEUTTRANSPORT, 27, 33

DUMP, 27, 33, 299, 300, 301

EGS, 12, 27, 33, 46, 336, 338, 339, 340

END, 22, 23, 26, 33, 38, 29, 262, 273, 284, 303, 367,
393, 394, 400

GEOMETRY, 14, 15, 21, 24, 25, 33, 34, 35, 55, 57,
58, 102, 7, 8, 10, 12, 15, 16, 18, 20, 29, 32, 67, 185,
262, 273, 283, 285, 302, 320, 340, 352, 366, 381,
393, 400, 402

I/O, 11, 27, 33, 56, 64, 66, 321, 326, 353

MIX, 15, 21, 25, 33, 36, 37, 58, 9, 10, 20, 22, 31, 45,
47, 49, 50, 56, 57, 58, 59, 61, 63, 64, 67, 68, 69, 71,

197, 198, 261, 262, 271, 273, 283, 285, 299, 302,
321, 340, 353, 366, 378, 393, 394, 400, 401, 407

PARALLEL, 27, 33, 308, 312, 313

RESTART, 17, 18, 19, 20, 27, 33, 43, 44, 299, 300,
301, 317

SOURCE, 10, 14, 18, 20, 21, 22, 26, 33, 35, 36, 37, 38,
76, 77, 80, 81, 82, 83, 84, 85, 86, 87, 89, 90, 91, 93,
106, 108, 128, 133, 134, 136, 138, 141, 143, 144,
146, 147, 148, 149, 150, 151, 153, 154, 155, 156,
157, 158, 159, 160, 161, 173, 189, 203, 207, 215,
217, 218, 219, 220, 227, 229, 241, 262, 263, 268,
273, 284, 285, 286, 287, 289, 291, 300, 303, 321,
322, 341, 353, 367, 382, 383, 384, 385, 388, 389,
390

Summary, 33

SURFACES, 18, 21, 24, 30, 33, 34, 35, 37, 55, 57, 59,
60, 69, 91, 102, 103, 12, 14, 17, 18, 19, 20, 185,
261, 262, 271, 273, 283, 285, 302, 340, 352, 393,
400

TITLE, 21, 22, 24, 33, 37, 38, 31, 35, 41, 168, 184,
185, 188, 213, 214, 215, 216

WALK, 15, 26, 27, 33, 46, 70, 91, 218, 220, 224, 229,
230, 231, 232, 233, 234, 235, 237, 238, 240, 241,
242, 243, 244, 245, 246, 248, 249, 250, 251, 256,
257, 258, 259, 263, 265, 267, 268, 269, 273, 275,
276, 281, 285, 287, 326, 367, 383, 385, 405, 438

Data Library

ACTL92, 50

ENDF/B-VIII.0, 11, 39, 51, 53

ENDFB8R0, 11, 48, 49, 51, 56, 57

ENDFB8R0.ACE, 11, 51

EPDL89, 11, 46, 50

EPDL97, 12, 39, 46, 50, 59

PT.ENDFB8R0.ACE, 11, 52

T.ENDFB8R0, 11, 53

T.ENDFB8R0.ACE, 11, 53

DETECTOR Data Block

COG Detectors, 70, 166, 168, 187, 215, 291, 302, 367,
381

Detector ANALYSIS, 215

Detector Differential

ANGLE, 169, 206, 207

ENERGY, 167, 169, 182, 204, 206, 209

TIME, 169, 178, 205

Detector IMPORTANCE, 217, 227

Detector Mask

MASK-A, 192

MASK-C, 194

MASK-E, 178, 191

MASK-HL, 195, 302

MASK-ISO, 193

MASK-REA, 194

MASK-REG, 191, 194

MASK-T, 178, 191
 Detector Multiple Bins
 2D Bins, 206
 3D and 4D Bins, 208
 Detector Plot Scales/Lables, 211
 Detector Response
 DRF-A, 201, 202, 203
 DRF-E, 196, 320
 DRF-T, 200
 Detector Specification, 168
 Detector Type
 Imaging, 184
 Point, 194
 Pulse, 180
 Reaction, 170
 Volume Point, 179
 DEUTERON Transport Data Block
 ddestep, 360
 decut, 360
 deuttrans, 360
 dfrac, 360
 dstepmax, 360
 dstepmin, 360
 dstragflag, 360
 matdt, 318, 360

E

EGS Data Block, 336, 338, 339
 CALLPEGS, 338
 ECUT, 338, 339
 EGS, 12, 27, 33, 46, 336, 338, 339, 340
 ESECTORS, 338, 339
 PEGSLIB, 338, 339
 Errors, 34
 Code Physics, 333
 Geometry, 39, 320
 Job Setup, 334
 Of Results, 335
 PICTURES, 39
 Under Sampling, 320
 VOLUME, 34, 41
 Errors checking, 34
 PICTURES, 34
 SWEEP, 34

G

GEOMETRY Data Block, 24, 25, 34, 55, 58, 102, 7, 16,
 29, 67, 320, 366, 381, 393, 402
 Absorber Sector, VOID, 9
 Boundary Conditions, 11
 Boundary Reflecting, 12

Boundary Vacuum, 11
 DEFINE UNIT, 10, 15
 Exclusion Operator, NOT, 107
 Fill sectors, FILL, 9
 Lattice Feature, {}, 22
 Periodic Boundary, 14
 PICTURE cross-section, CS, 30
 PICTURE Patterns, 30, 34
 PICTURE Perspective, 8, 34, 35, 37, 72, 320
 PICTURES, 29, 72, 320
 Rectangular Lattice, 22
 SECTOR, 102
 Sweeps a line, SWEEP, 43
 Union Operator, OR, 105
 UNIT FILL, 22
 USE UNIT, 15, 16, 17, 18, 19, 20, 21, 22
 Volume Calculation, VOLUME, 34, 41

I

I/O Data Block
 Output Additional Problem Information, 11, 56, 64,
 321, 326
 Plot isotope/element cross section, PLOTXS, 64, 66
 Write all isotope/element cross section, PRINTXS, 64,
 66

M

MIX Data Block
 Activation Library, ACTL92, 50
 Apha Library, ALIB, 54, 59
 ATOM-FRACTION, 45, 47, 48
 BUNCHES, 47, 48
 COGNUP, 12, 51
 Cross Section Libraries, 56
 Delayed Fission Gamma Library, DFG.ENDFB7R1,
 51, 60
 Delayed Fission Gamma Library, DFGLIB, 51, 60
 Density of components, (gm/cc), 47
 Deuteron Library, DLIB, 59
 Dosimetry Library, DOSLIB, 54, 60
 Dosimetry Library, IRDFF1.02, 54, 60
 Neutron Library specification, NLIB, 51, 56
 Neutron Library, ENDFB7R1, 51, 56
 NLIB, 47, 48, 56, 57, 63, 302
 Nuclear Resonance Fluorescence Library, COGNRF,
 51
 Photon Library, EDPL97, 50, 59
 Photon Library, PLIB, 50, 59
 Photonuclear Library, PN.ENDFB7R1, 51, 60
 Photonuclear, PNLIB, 51, 60

Probability Table Library, PT.ENDFB7R1.BNL, 52, 60
 Probability Table Library, PTLIB, 52, 60
 S(α , β) Libraries, SABLIB and SABLIB2, 58
 Secondary Neutron Libraries specification, NLIB2 and NLIB3, 57
 Specify Material, MIX, 25, 58, 9, 10, 22, 31, 45, 47, 49, 50, 56, 58, 59, 61, 63, 64, 67, 68, 197, 198, 299, 321, 366, 378, 394, 401
 TEMP (Free-Gas Thermal), 48
 Thermal Library, T.ENDFB7R1, 53
 Thermal Treatment, (), 50
 WEIGHT-PERCENT, 45, 47, 48

N

Notation Conventions, 30, 240
 Notation Shortcuts, 29, 78

O

Output Files, 20
 Overview
 Activation calculations, 14
 Alpha Transport, 13, 354
 COG Particle Transport, 10
 Deuteron Transport, 13
 Electron Transport, 8, 12, 27, 39, 336, 338, 339, 342, 376, 400
 Neutron Transport and Cross Sections, 11
 Photon Transport, 11, 46
 Photonuclear Reactions, 12, 46
 Proton Transport, 13, 344

P

Parallel Processing, 306, 307, 308, 315
 Criticality Job Restart, 316
 HPC (LLNL LC Only), 313
 Maximum CPUs, 312
 sbatch, 313, 315
 Source Job Restart, 318
 Very Big COG job, 316

R

RESTART, 17, 18, 19, 20, 27, 33, 43, 44, 299, 300, 301, 317
 A Criticality Job, 27, 33, 299, 300, 301
 NDUMP, 18, 19, 44, 53
 Running COG Longer, 17, 18, 19, 20, 27, 33, 43, 44, 299, 300, 301, 317

TDUMP, 18, 19, 43, 44, 53
 Running COG, 16, 17, 44, 312, 313
 Background Job, 16, 20
 Batch Runs, 17
 Job Execution, 16
 Longer Job, 17, 44
 Parallel Runs, 17

S

SOURCE Data Block, 18, 22, 26, 76, 77, 80, 81, 82, 83, 85, 87, 148, 153, 173, 189, 217, 220, 227, 286, 289, 291, 321, 322, 367, 382
 ANGLE dependence, 76, 133
 ANGLE Graphics, 136
 Angle source, 142
 ANGLE-dependent source, 87, 90, 91, 133, 136, 140, 158, 160, 161
 BIN distribution, 141
 BIN Energy Source, 121, 180, 181
 BIN Time Source, 131
 CASCADE Line, 112
 CASCADE Line Energy Source, 112
 CENSUS, 153, 173
 CIRCLE Line source, 96
 CORRELATED parameter, 20, 80, 81, 84, 85, 189, 190, 291
 DELTA Function Time source, 127
 discrete energy lines, 110
 Double-Angle source, 143, 203, 207
 ENERGY dependence, 87, 108, 158, 159, 160, 161, 336, 346
 ENERGY-dependent source distribution, 108
 FissSrc (full fission source), 123, 124
 FIXED Direction Angle Source, 133, 134, 138, 140, 192, 193, 201, 202, 203, 205, 206, 207, 220
 GAUSSIAN Energy Source, 115
 LINE Energy Source, 110
 LINE Energy-dependent, 111
 Maximum Particles, 80
 Maxwellian distribution, 117
 Maxwellian Energy Source, 117
 Miscellaneous Parameters, 80
 NPART parameter, 80, 81, 83, 85, 89, 151, 153, 154, 158, 159, 160, 161, 189, 262, 268, 273, 284, 286, 292, 293, 295, 300, 303, 316, 317, 353
 ONE/E Energy Source, 118
 POINT source, 93, 95
 Point-Wise distribution, 140
 Point-Wise Energy Distribution, 119
 Point-Wise Time Distribution, 130
 POSITION dependence, 87, 93, 158, 159, 160, 161
 POSITION Graphics, 93

PULSE Time Source, 128
 RADSRC, 113
 RADSRC Energy Source, 113
 READSOURCE parameter, 80, 81, 85
 RETRACE parameter, 22, 80, 81, 82, 83, 166, 176, 209, 291, 325, 388, 389
 SFS (spontaneous fission source), 123, 124, 161
 Source ANGLE Dependence, 76, 133
 Source ENERGY Dependence, 76, 108
 Source INCREMENTS, 86, 123, 124
 Source POSITION, 93, 353
 Source POSITION Dependence, 76, 93
 Source TIME Dependence, 76, 125
 Special Free-Form Inputs, 78
 Spontaneous Fission Energy Source, 123, 161
 STEADY State Time Source, 132, 153
 Straight LINE source, 96
 TIME dependence, 76, 125
 TIME GAUSSIAN Time source, 129
 TIME Graphics, 126
 TIME-dependent source, 125, 126, 127, 128, 129, 130, 131, 132, 178
 USRSOR, 151
 WATT Fission Neutron source, 116
 Watt fission spectrum, 116
 WRITESOURCE parameter, 80, 81, 189
 Special Surfaces, 99
 ANALYTIC Surface, 99, 100
 ANALYTIC Surface, ANA, 99
 Topographical Surface, 101
 Topographical Surface, TOP, 101, 106, 216
 SURFACE Data Block, 25, 55, 58, 67, 72, 17, 367, 379, 393, 402
 bounded by planes, 62
 Box, 3, 75
 Cone, CON, 90
 Cylinder, CYL, 83, 285
 Ecyliner, ECYL, 85
 Ellipsoid, ELL, 82
 Elliptic Paraboloid, EPAR, 94
 Elliptical Cone, ECON, 92
 First Order Surfaces, 70
 Hexahedron, 76
 Hyperbolic cylinder, HCYL, 87
 Hyperbolic Parabola, HPAR, 95
 Hyperboloid of Two Sheets, HYP2, 93
 Hyperboloid, HYP1, 86, 93
 negative side of surface, 60, 61, 62, 71, 103
 Parabolic Cylinder, PCYL, 89
 Pentahedron, 74
 Plane, 70, 30
 positive side of a surface, 60, 61, 62, 103
 Prism, 78

Pyramid, 79
 Revolution, REV, 97, 98
 RPP, 75
 Special Figures of Rotation, 96
 Sphere, SPH, 81
 SURFACE-NAME, 59
 Tetrahedron, 73
 Torus, TOR, 96
 TR, 63
 Surface Sources, 97
 SS-PARALLELOGRAM, 97
 SS-SPHERE, 99

V

Volume Sources, 103, 107
 Arbitrary volume source, 107, 149
 BOX, 103
 CONE, 105
 CYLINDER, 104
 PARALLELEPIPED, 103
 SPHERE, 107

W

WALK Data Block, 46, 224, 229
 WALK-AGE (Age Cutoff), 234
 WALK-BC (Splitting at a Boundary), 235
 WALK-COLLISION (Splitting at a Collision), 235
 WALK-ENERGY (Low-energy Cutoff), 240
 WALK-FC (Forced Collisions), 241, 244
 WALK-FPP (Fast Photon Physics), 12, 243
 WALK-ISOTOPE (Forced Collision with an Isotope), 244
 WALK-PRODUCTION (Secondary Particle Control), 245
 WALK-PS (Path Stretching), 246, 251
 WALK-REACTION (Forced Reaction), 250
 WALK-SDB (Scattered Direction Bias), 251
 WALK-SURVIVAL (Survival Weighting), 257
 WALK-WT (Weight Window), 259
 WALK-XX, 230
 WALK-AGE, 232, 234
 Age of a particle, 232, 234
 Age Termination, 234