SANDIA REPORT

SAND90—0566 • UC—405
Unlimited Release
Printed April 1990

GREPOS: A GENESIS Database Repositioning Program

Gregory D. Sjaardema

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
Office of Scientific and Technical Information
PO Box 62
Oak Ridge, TN 37831

Prices available from (615) 576-8401, FTS 626-8401

Available to the public from
National Technical Information Service
US Department of Commerce
5285 Port Royal Rd
Springfield, VA 22161

NTIS price codes
Printed copy: A03
Microfiche copy: A01

SAND90-0566
Unlimited Release
Printed April 1990

Distribution
Category UC-405

SAND--90-0566

DE90 010254

GREPOS:
A GENESIS Database Repositioning Program

Gregory D. Sjaardema
Applied Mechanics Division I
Sandia National Laboratories, Albuquerque, NM

Abstract

GREPOS is a mesh utility program that repositions or modifies the configuration of a two-dimensional or three-dimensional mesh. *GREPOS* can be used to change the orientation and size of a two-dimensional or three-dimensional mesh; change the material block, nodeset, and sideset IDs; or "explode" the mesh to facilitate viewing of the various parts of the model. *GREPOS* also updates the EXODUS Quality Assurance (QA) and information records to help track the codes and files used to generate the mesh. *GREPOS* reads and writes two-dimensional and three-dimensional mesh databases in the GENESIS database format; therefore, it is compatible with the preprocessing, postprocessing, and analysis codes used by the Engineering Analysis Department at Sandia National Laboratories (SNL).

Contents

Figures

1. Introduction

GREPOS is a mesh utility program that repositions or modifies the configuration of a two-dimensional or three-dimensional mesh. *GREPOS* is intended to be used with the mesh generation codes *GEN3D* [1], *GJOIN* [2], and *FASTQ* [3].

Several mesh generation codes are used in the Engineering Analysis Department at Sandia National Laboratories (SNL). Two-dimensional meshes are typically generated by *FASTQ* and *PATRAN* [4]. Until recently, three-dimensional meshes were generated entirely by *PATRAN*. Then, the codes *GEN3D*, which transforms a two-dimensional mesh into a three-dimensional mesh, and *GJOIN*, which merges several two-dimensional or three-dimensional mesh parts into a single mesh, were developed to provide additional three-dimensional mesh generation capability.

GREPOS was written to add additional flexibility to the *GEN3D* and *GJOIN* mesh generation system. It can be used to modify meshes generated by any of the two-dimensional or three-dimensional mesh generation codes used in the Engineering Analysis Department. *GREPOS* reads and writes files in the GENESIS [5] database file format which is the geometry definition portion of the EXODUS [6] database format used in the Engineering Analysis Department. All of the mesh generation programs in the Engineering Analysis Department write files in the EXODUS format, and all of the pre- and postprocessing programs read EXODUS format files. Therefore, the two-dimensional or three-dimensional mesh input to *GREPOS* can be generated by *FASTQ*, *PATRAN*, or any other two-dimensional or three-dimensional mesh generator that writes the GENESIS file format. The output from *GREPOS* can be graphically displayed by *BLOT* [7] or *PATRAN*, and examined by *GROPE* [8] or *NUMBERS* [9]. Figure 1.1 shows a schematic of the mesh generation process.

1.1 Overview of Capabilities in *GREPOS*

With *GREPOS*, the user can change the orientation and size of a two-dimensional or three-dimensional mesh; change or delete material block, nodeset, and sideset IDs; or "explode" the mesh to facilitate viewing of the various parts of the model. *GREPOS* also updates the EXODUS Quality Assurance (QA) and information records to help track the codes and files used to generate the mesh.

Orientation: The orientation of the mesh can be changed by revolving, offsetting, reflecting, zeroing, and scaling the nodal coordinates of the original mesh.

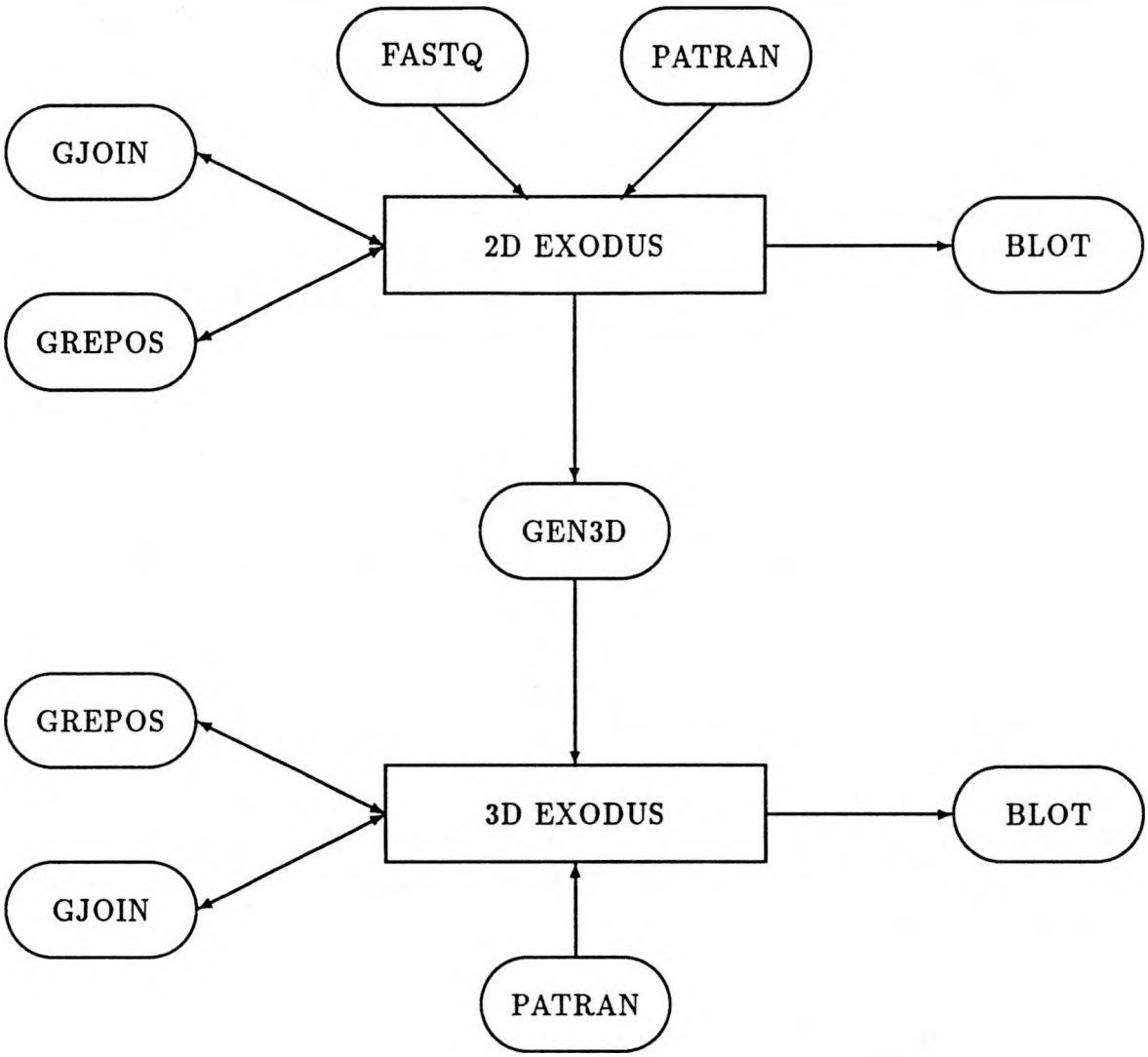

Figure 1.1. Mesh Generation Codes Used in the Engineering Analysis Department at SNL

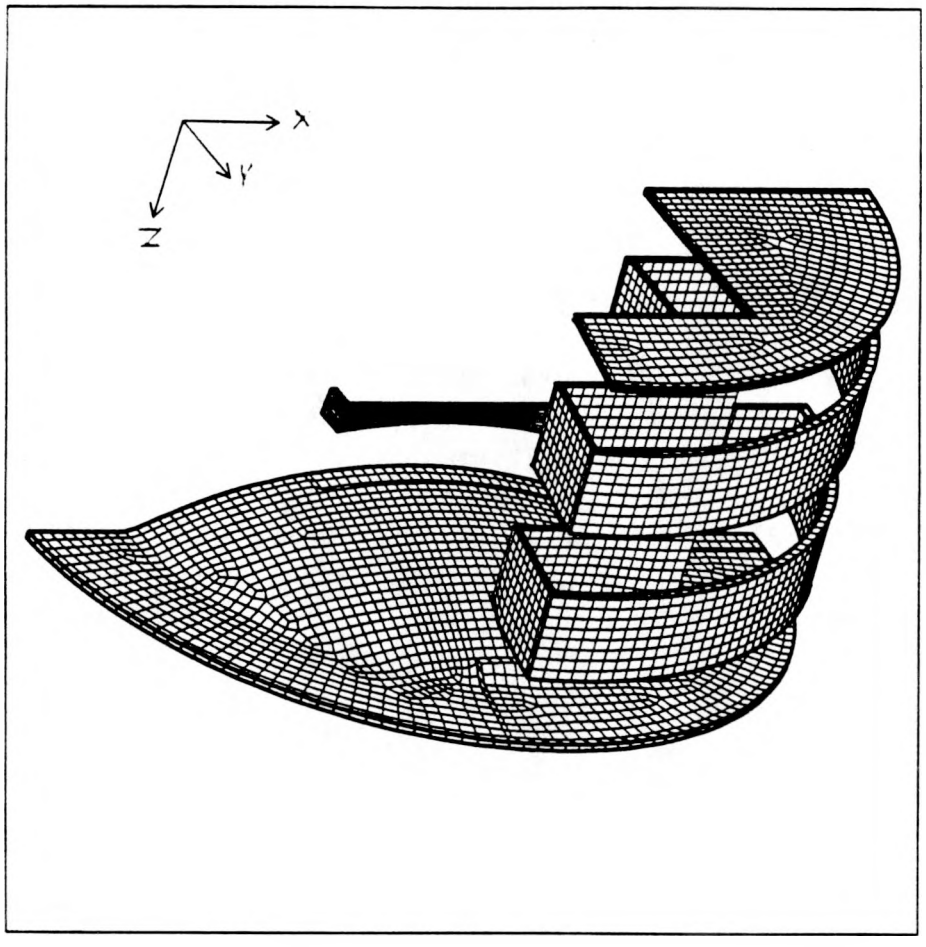

Figure 1.2. Example of the EXPLODE command

Material Blocks, Sidesets, and Nodesets: In the Engineering Analysis Department system, material properties are input according to material block IDs. Similarly, boundary conditions are associated with sideset and nodeset IDs. The input material block IDs, sideset IDs, and nodeset IDs can be changed to a new unique ID (that is they can be changed, but not combined). Material blocks, sidesets, and nodesets can also be deleted.

Mesh "Explosion" for Viewing Multiple Material Meshes: An exploded view generation capability has been implemented in *GREPOS*. An exploded view is generated by offsetting each material block by a specified distance in each of the two or three coordinate directions. This allows viewing pieces of the mesh that are hidden by other pieces. Figure 1.2 shows an example of an exploded mesh. In this figure, each of the material blocks was offset in the Z direction.

Quality Assurance (QA) and Information Records: A QA record for the *GREPOS* program is added to the input QA record(s), and the filename of the input file is added to the information records that are written to the output mesh. The information record is prefaced by *GREPOS* to help in identification. These records, which are explained in Reference [6], are useful in tracing the evolution of a mesh during the mesh generation process.

1.2 Organization of Report

The remainder of this report is organized as follows. Chapter 2 describes the command input and valid commands, Chapter 3 describes the informational and error messages written by *GREPOS*, and Chapter 4 describes the execution environment of *GREPOS*. Three appendices are contained in this report. Appendix A is a code segment defining the GENESIS binary database format, Appendix B is a summary of the commands in *GREPOS*, and Appendix C is a site supplement describing the operating system commands necessary to execute *GREPOS* on all currently supported operating systems at SNL, Albuquerque.

2. Command Input

The user directs the execution of *GREPOS* by entering commands to set processing parameters. The commands are in free-format and must adhere to the following syntax rules.

- Valid delimiters are a comma or one or more blanks.
- Either lowercase or uppercase letters are acceptable, but lowercase letters are converted to uppercase.
- A "$" character in any command line starts a comment. The "$" and any characters following it on the same line are ignored.
- A command may be continued over several lines with an "*" character. The "*" and any characters following it on the current line are ignored and the next line is appended to the current line.

Each command has an action keyword or "verb" followed by a variable number of parameters.

A command verb or keyword is a character string matching one of the valid commands. It may be abbreviated as long as enough characters are used to distinguish it from other commands.

The meaning and type of the parameters depend on the command verb. Most parameters are optional. If an optional parameter field is blank, a command-dependent default value is supplied. Valid entries for parameters are:

- A numeric parameter may be a real number or an integer. A real number may be in any legal FORTRAN numeric format (e.g., 1, 0.2, -1E-2). An integer parameter may be in any legal integer format.
- A string parameter is a literal character string. Most string parameters may be abbreviated.

The notation conventions used in the command descriptions are:

- The command verb is in **bold** type.
- A literal string is in all uppercase SANSERIF type and should be entered as shown (or abbreviated).
- The value of a parameter is represented by the parameter name in *italics*.

- A literal string in square brackets ("[ ]") represents a parameter option which is omitted entirely (including any following comma) if not appropriate. These parameters are distinct from most parameters in that they do not require a comma as a place holder to request the default value.
- The default value of a parameter is in angle brackets ("< >"). The initial value of a parameter set by a command is usually the default parameter value. If not, the initial setting is given in the command description.

2.1 Mesh Orientation

EXPLODE <no parameters>

EXPLODE initiates an input mode where *GREPOS* prompts for the coordinate offsets to be applied to each material block. Before performing the material block offsets, *GREPOS* checks for connections between the material blocks. All connected material blocks will be assigned the same offset to avoid distorting the mesh. An informational message will be output to the standard output device for each connected material block.

MIRROR $axis_1$, $axis_2$, ... <no default>
MIRROR RESET <no reflections>

MIRROR causes the mesh to be reflected about a coordinate plane. Each *axis* parameter specifies which coordinates (X or Y or Z) will be modified. Reflections are performed after the mesh has been repositioned by the REVOLVE and OFFSET commands.

The MIRROR RESET command resets to no reflection.

Reflections are not cumulative, that is, if MIRROR X Y Y X is entered, only one X and Y reflection will be performed. The element connectivity and the sideset face numbering will be correctly reordered.

OFFSET [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, ...
OFFSET [ADD] ALL *offset* <0.0>
OFFSET RESET <initial condition>
OFFSET *xoff* <0.0>, *yoff* <0.0>, *zoff* <0.0>

OFFSET specifies offsets to be added to the coordinates. If a REVOLVE command has been issued, the mesh is rotated before it is offset. The last form of the offset command is included to maintain compatibility with the offset command in *GEN3D*.

OFFSET ALL offsets all of the coordinates by the specified offset, and OFFSET RESET resets the offsets to zero.

If the optional keyword ADD is specified, offsets are cumulative, that is, if OFFSET ADD X 0.5 X 1.0 is entered, the X coordinates will be offset by 1.5. If ADD is omitted from the above line, the X coordinates will be offset by 1.0.

REVCEN *xcen* <2D minimum X coordinate>, *ycen* <2D minimum Y coordinate>, *zcen* <0.0> (three-dimensional mesh)

REVCEN sets the center of revolution for the REVOLVE command to the point (*xcen*,*ycen*,*zcen*).

REVOLVE $axis_1$, $ndeg_1$, $axis_2$, $ndeg_2$, ... <last selection>
REVOLVE RESET <initial condition>

REVOLVE causes the mesh to be rotated. Each ($axis_i$, $ndeg_i$) parameter pair specifies an axis (X or Y or Z) and the number of degrees to rotate. The axis refers to the "viewing" axis, not to the object axis[1]. The rotations are according to right-hand rule. The center of the rotation is specified by the REVCEN command.

Revolutions are cumulative and order-dependent; however, only one center of revolution (see REVCEN) may be specified. The REVOLVE RESET command resets to no rotation.

For a two-dimensional mesh, $axis_i$ must be Z.

SCALE $axis_1$, $scale_1$, $axis_2$, $scale_2$, ...
SCALE ALL *scale_factor* <1.0>
SCALE RESET <initial condition>

SCALE causes the specified coordinates of axis $axis_i$ to be multiplied by the scaling multiplier $scale_i$. The scaling multiplier must be greater than zero; the MIRROR command must be used with the SCALE command if a negative scaling multiplier is required. For example,

```
SCALE X 2.54
MIRROR X
```

will scale the X coordinates by -2.54.

SCALE ALL multiplies all of the coordinates by the specified multiplier, and SCALE RESET resets to no scaling.

Scalings are cumulative, that is, if SCALE X 0.5 X 0.6 is entered, the X coordinates will be scaled by 0.3.

---

[1]The viewing axis is defined relative to the viewing screen such that the X axis is horizontal, the Y axis is vertical, and the Z axis is out of the screen. The object axis is defined relative to the object.

SHIFT [**ADD**] $axis_1$, $shift_1$, $axis_2$, $shift_2$, ...
SHIFT [**ADD**] **ALL** *shift* <0.0>
SHIFT RESET <initial condition>
SHIFT *xoff* <0.0>, *yoff* <0.0>, *zoff* <0.0>

SHIFT is simply a synonym for the OFFSET command. See the description of the OFFSET command for more information.

ZERO $axis_1$, min_1, $axis_2$, min_2, ...
ZERO RESET <no automatic zeroing>

ZERO sets all $axis_i$ coordinates with an absolute value less than min_i equal to zero. The ZERO RESET command resets to no automatic zeroing. This command is used to zero nodal coordinates that should be equal to zero, but due to roundoff errors they have slightly nonzero values.

2.2 Modification of Material, Nodeset, or Sideset IDs

CHANGE MATERIAL|NODESET|SIDESET *old_id* <no default>*new_id* <no default>

CHANGE changes the identification number *old_id* of a material block, nodeset, or sideset to *new_id*. The *new_id* must be unique, that is, CHANGE can only be used to change the identification number; it cannot be used to combine or delete identification numbers (use *GJOIN* to combine material blocks, sidesets, or nodesets).

DELETE MATERIAL|NODESET|SIDESET id_i ...

DELETE deletes the identification material block, nodeset, or sideset with ID id_i. Multiple IDs can be deleted on the same line. Note: this command can not be undone, once an ID is deleted, it is gone[2].

2.3 Information and Processing

SHOW *command* <no parameter>

SHOW displays the settings of parameters relevant to the *command*. For example, the command SHOW REVOLVE displays information about the currently defined revolution.

LIST *option*

[2]Actually, DELETE sets the ID to zero to flag that the entity is deleted. Therefore, if you CHANGE ID 0 to the original ID number, you can recover deleted materials, nodesets, or sidesets.

LIST displays information about the requested *option*. Valid options are: VARS, MATERIAL, BLOCKS, NODESET, SIDESET, and COMMANDS.

LIST {VARS|VARIABLES}
: displays a summary of the database. The summary includes the database title; the number of nodes, elements, and element blocks; the number of node sets and side sets; and the number of each type of variable.

LIST {BLOCKS|MATERIALS}
: displays a summary of the element blocks. The summary includes the block identifier, the number of elements in the block, the element number of the first and last element in the block, and the block status, either selected or not selected.

LIST {NSETS|NODESETS}
: displays a summary of the node sets. The summary includes the set identifier and the number of nodes in the set.

LIST {SSETS|SIDESETS}
: displays a summary of the side sets. The summary includes the set identifier, the number of elements in the set, and the number of nodes in the set.

HELP *command* <general help>

HELP displays information about the program command given as the parameter. If no parameter is given, all the command verbs are displayed. This command is system-dependent and may not be available on some systems.

END
EXIT

END or EXIT ends the command input and starts the mesh transformation.

QUIT

QUIT ends the command input and exits the program immediately without writing an output database.

2.4 Order of Transformation

Although *GREPOS* commands may be entered in any order, processing of the commands is performed in the order shown below:

DELETE
REVOLVE
OFFSET
MIRROR
ZERO
SCALE
EXPLODE

If order-dependent processing is required, *GREPOS* will have to be run multiple times. For example, to mirror about a line not parallel to a coordinate axis, you would first revolve the mesh such that the reflection line was parallel to a coordinate axis, mirror about that axis, and write a temporary file. Then, you would rerun *GREPOS* on the temporary file and revolve the mesh back to the original orientation.

The CHANGE operations are performed in the order they are entered.

3. Informational and Error Messages

GREPOS first reads the input database, which must be a valid GENESIS database. If a database format error is discovered, the program prints an error of the following format:

DATABASE ERROR - Reading *database item*

and aborts.

After the database is read, command input is requested from the user. An error or warning message may appear in response to a command. If an error message appears, the command is usually ignored. If only a warning is printed, the command is usually performed. If the message is not sufficiently informative, the appropriate command description may be helpful. The display after the command shows the effect of the command.

When the command input is complete, *GREPOS* transforms the mesh and writes the output database.

The program allocates memory dynamically as it is needed. If the system runs out of memory, the following message is printed:

FATAL ERROR - Too much dynamic memory requested

and the program aborts. The user should first try to obtain more memory on the system. Another solution is to run the program in a less memory-intensive fashion. For example, reducing the number of transformations requires less memory.

GREPOS has certain programmer-defined limitations. The limits are not specified in this manual since they may change. In most cases the limits are chosen to be more than adequate. If the user exceeds a limit, a message is printed. If the user feels the limit is too restrictive, the code sponsor should be notified so the limit may be raised in future releases of *GREPOS*.

If the user tries to change an element block, sideset, or nodeset ID to an existing ID, an error message of the form

Cannot change to an existing ID

will be printed and the command will not be performed. The EXPLODE command checks that all nodes of the mesh are connected to an element. If a non-connected node is found an error message of the form

Node not connected to any elements

will be output. The EXPLODE command will be executed. In addition, if two or more element blocks are connected a message of the form

EXPXYZ – Material *id_1* is connected to material *id_2*

is output and the offset for material block *id_2* is set equal to that for material block *id_1*.

4. Executing *GREPOS*

The details of executing *GREPOS* are system dependent. The system manager of any system that runs *GREPOS* should provide a supplement to this manual that explains how to run the program on that particular system. Site supplements for all currently supported systems at SNL, Albuquerque, are in Appendix C.

4.1 Execution Files

The table below summarizes *GREPOS*'s file usage.

Description	Unit Number	Type	File Format
User input	standard input	input	Section 2
User output	standard output	output	ASCII
GENESIS database	9	input	Appendix A
GENESIS database	10	output	Appendix A

All files must be connected to the appropriate unit before *GREPOS* is run. Each file (except standard input and output) is opened with the name retrieved by the EXNAME routine of the *SUPES* [10] library.

4.2 Special Software

GREPOS is written in ANSI FORTRAN-77 [11] with the exception of the following system-dependent features:

- the VAX VMS help facility and
- the OPEN options for the files.

GREPOS uses the following software package:

- the *SUPES* package which includes dynamic memory allocation, a free-field reader, and FORTRAN extensions.

References

[1] A. P. Gilkey and G. D. Sjaardema, "GEN3D: A GENESIS Database 2D to 3D Transformation Program," SAND89-0485, Sandia National Laboratories, Albuquerque, NM, March 1989.

[2] J. H. Biffle, 1523, "Three-dimensional mesh generation with GJOIN," (Memo to 1520 Staff), Sandia National Laboratories, Albuquerque, NM, October 7, 1988.

[3] T. D. Blacker, "FASTQ Users Manual, Version 2.1," SAND88–1326, Sandia National Laboratories, Albuquerque, NM, July 1988.

[4] PDA Engineering Software Products Division, Santa Ana, CA, *PDA/PATRAN–G Version 2.1 User's Guide*, 1986.

[5] L. M. Taylor, D. P. Flanagan, and W. C. Mills-Curran, "The GENESIS Finite Element Mesh File Format," SAND86–0910, Sandia National Laboratories, Albuquerque, NM, May 1986.

[6] W. C. Mills-Curran, A. P. Gilkey, and D. P. Flanagan, "EXODUS: A Finite Element File Format for Pre- and Post-processing," SAND87–2977, Sandia National Laboratories, Albuquerque, NM, September 1988.

[7] A. P. Gilkey, "BLOT—A Mesh and Curve Plot Program for the Output of a Finite Element Analysis," SAND88–1432, Sandia National Laboratories, Albuquerque, NM, June 1989.

[8] A. P. Gilkey, "GROPE - A GENESIS/EXODUS Database Examination Program," (RS1523/88/02), Sandia National Laboratories, Albuquerque, NM. In preparation.

[9] G. D. Sjaardema, "NUMBERS: A Collection of Utilities for Pre- and Postprocessing Two- and Three-Dimensional EXODUS Finite Element Models," SAND88–0737, Sandia National Laboratories, Albuquerque, NM, March 1989.

[10] D. P. Flanagan, W. C. Mills-Curran, and L. M. Taylor, "SUPES A Software Utilities Package for the Engineering Sciences," SAND86-0911, Sandia National Laboratories, Albuquerque, NM, September 1986.

[11] "American national standard programming language fortran," Tech. Rep. ANSI X3.9–1978, American National Standards Institute, New York, 1978.

Appendix A
The GENESIS Database Format

The following code segment reads a GENESIS database.

```
C   --Open the GENESIS database file

      NDB = 9
      OPEN (UNIT=NDB, ..., STATUS='OLD', FORM='UNFORMATTED')

C   --Read the title

      READ (NDB) TITLE
C      --TITLE - the title of the database (CHARACTER*80)

C   --Read the database sizing parameters

      READ (NDB) NUMNP, NDIM, NUMEL, NELBLK,
     &   NUMNPS, LNPSNL, NUMESS, LESSEL, LESSNL
C      --NUMNP - the number of nodes
C      --NDIM - the number of coordinates per node
C      --NUMEL - the number of elements
C      --NELBLK - the number of element blocks
C      --NUMNPS - the number of node sets
C      --LNPSNL - the length of the node sets node list
C      --NUMESS - the number of side sets
C      --LESSEL - the length of the side sets element list
C      --LESSNL - the length of the side sets node list

C   --Read the nodal coordinates

      READ (NDB) ((CORD(INP,I), INP=1,NUMNP), I=1,NDIM)

C   --Read the element order map (each element must be listed once)

      READ (NDB) (MAPEL(IEL), IEL=1,NUMEL)
```

```
C   --Read the element blocks

      DO 100 IEB = 1, NELBLK

C      --Read the sizing parameters for this element block

         READ (NDB) IDELB, NUMELB, NUMLNK, NATRIB
C         --IDELB - the element block identification (must be unique)
C         --NUMELB - the number of elements in this block
C         --   (the sum of NUMELB for all blocks must equal NUMEL)
C         --NUMLNK - the number of nodes defining the connectivity
C         --   for an element in this block
C         --NATRIB - the number of element attributes for an element
C         --   in this block

C      --Read the connectivity for all elements in this block

         READ (NDB) ((LINK(J,I), J=1,NUMLNK, I=1,NUMELB)

C      --Read the attributes for all elements in this block

         READ (NDB) ((ATRIB(J,I), J=1,NATRIB, I=1,NUMELB)

  100 CONTINUE
```

```
C   --Read the node sets

      READ (NDB) (IDNPS(I), I=1,NUMNPS)
C      --IDNPS - the ID of each node set
      READ (NDB) (NNNPS(I), I=1,NUMNPS)
C      --NNNPS - the number of nodes in each node set
      READ (NDB) (IXNNPS(I), I=1,NUMNPS)
C      --IXNNPS - the index of the first node in each node set
C      --   (in LTNNPS and FACNPS)

      READ (NDB) (LTNNPS(I), I=1,LNPSNL)
C      --LTNNPS - the nodes in all the node sets
      READ (NDB) (FACNPS(I), I=1,LNPSNL)
C      --FACNPS - the factor for each node in LTNNPS

C   --Read the side sets

      READ (NDB) (IDESS(I), I=1,NUMESS)
C      --IDESS - the ID of each side set
      READ (NDB) (NEESS(I), I=1,NUMESS)
C      --NEESS - the number of elements in each side set
      READ (NDB) (NNESS(I), I=1,NUMESS)
C      --NNESS - the number of nodes in each side set
      READ (NDB) (IXEESS(I), I=1,NUMESS)
C      --IXEESS - the index of the first element in each side set
C      --   (in LTEESS)
      READ (NDB) (IXNESS(I), I=1,NUMESS)
C      --IXNESS - the index of the first node in each side set
C      --   (in LTNESS and FACESS)

      READ (NDB) (LTEESS(I), I=1,LESSEL)
C      --LTEESS - the elements in all the side sets
      READ (NDB) (LTNESS(I), I=1,LESSNL)
C      --LTNESS - the nodes in all the side sets
      READ (NDB) (FACESS(I), I=1,LESSNL)
C      --FACESS - the factor for each node in LTNESS
```

A valid GENESIS database may end at this point or after any point described below.

```
C   --Read the QA header information

      READ (NDB, END=...) NQAREC
C      --NQAREC - the number of QA records (must be at least 1)

      DO 110 IQA = 1, MAX(1,NQAREC)
         READ (NDB) (QATITL(I,IQA), I=1,4)
C         --QATITL - the QA title records; each record contains:
C         --   1) analysis code name (CHARACTER*8)
C         --   2) analysis code qa descriptor (CHARACTER*8)
C         --   3) analysis date (CHARACTER*8)
C         --   4) analysis time (CHARACTER*8)
  110 CONTINUE

C   --Read the optional header text

      READ (NDB, END=...) NINFO
C      --NINFO - the number of information records

      DO 120 I = 1, NINFO
         READ (NDB) INFO(I)
C         --INFO - extra information records (optional) that contain
C         --   any supportive documentation that the analysis code
C         --   developer wishes (CHARACTER*80)
  120 CONTINUE

C   --Read the coordinate names

      READ (NDB, END=...) (NAMECO(I), I=1,NDIM)
C      --NAMECO - the coordinate names (CHARACTER*8)

C   --Read the element type names

      READ (NDB, END=...) (NAMELB(I), I=1,NELBLK)
C      --NAMELB - the element type names (CHARACTER*8)
```

Appendix B
Command Summary

Mesh Orientation (page 12)

EXPLODE
: causes each material block in the mesh to be offset the specified distances.

MIRROR $axis_1$, $axis_2$, ...
MIRROR RESET
: causes the mesh to be reflected about the specified axes, or resets the mesh to no reflections.

OFFSET [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, ...
OFFSET [ADD] ALL *offset*
OFFSET RESET
OFFSET *xoff*, *yoff*, *zoff*
: specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for SHIFT command.

REVCEN *xcen*, *ycen*, *zcen*
: sets the center of rotation for the REVOLVE command.

REVOLVE $axis_1$, $ndeg_1$, $axis_2$, $ndeg_2$, ...
REVOLVE RESET
: causes the mesh to be rotated, or resets the mesh to no rotations.

SCALE $axis_1$, $scale_1$, $axis_2$, $scale_2$, ...
SCALE ALL *scale_factor*
SCALE RESET
: causes the mesh to be scaled, or resets the mesh to no scaling.

SHIFT [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, ...
SHIFT [ADD] ALL *offset*
SHIFT RESET
SHIFT *xoff*, *yoff*, *zoff*
: specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for OFFSET command.

ZERO $axis_1$, min_1, $axis_2$, min_2, ...
ZERO RESET

sets all *axis*$_i$ coordinates with an absolute value less than *min*$_i$ equal to zero, or resets the mesh to no automatic zeroing.

Modification of Material, Nodeset, or Sideset IDs (page 14)

CHANGE MATERIAL|NODESET|SIDESET *old_id new_id*
: modifies the identification number of the material block, nodeset, or sideset.

DELETE MATERIAL|NODESET|SIDESET *id* ...
: deletes the material block, nodeset, or sideset with ID *id*.

Information and Processing (page 14)

END
EXIT
: ends command input and starts Aprocessing.

HELP *command*
: displays information about a *GREPOS* command.

LIST *option*
: displays the database information specified by *option*.

```
LIST {VARS|VARIABLES}
LIST {BLOCKS|MATERIALS}
LIST {NSETS|NODESETS}
LIST {SSETS|SIDESETS}
```

QUIT
: quits command input and aborts processing.

SHOW *command*
: displays the processing parameters set by a command.

Order of operations
: REVOLVE OFFSET MIRROR ZERO SCALE EXPLODE

Appendix C
Site Supplements

C.1 VAX VMS

The command to execute *GREPOS* on VMS is:

GREPOS input_database output_database user_input

input_database is the filename of the input GENESIS database. A prompt appears if *input_database* is omitted. A default file type of **GEN** is assumed.

output_database is the filename of the output GENESIS database. A prompt appears if *output_database* is omitted. The default is the same filename as *input_database.*

If *user_input* is given, the user input is read from this file. The default file type is **GREPOS**. Otherwise it is read from SYS$INPUT (the terminal keyboard). User output is directed to SYS$OUTPUT (the terminal).

GREPOS operates in either interactive or batch modes.

Index

Distribution:

1510 J. W. Nunziato
1511 D. K. Gartling
1511 L. A. Schoof
1520 L. W. Davison
1521 H. S. Morgan and Staff (12)
1521 G. D. Sjaardema (10)
1522 R. C. Reuter, Jr. and Staff (15)
1523 J. H. Biffle and Staff (12)
1524 D. R. Martinez and Staff (12)
1530 J. R. Asay
1533 S. T. Montgomery
1550 C. W. Peterson, Jr.
3141 S. A. Landenberger (5)
3151 W. I. Klein (3)
3154-1 C. L. Ward for DOE/OSTI (8)
6258 D. S. Preece
6322 R. E. Glass
6334 R. P. Rechard
8240 C. W. Robinson
8241 G. A. Benedetti
8241 M. L. Chiesa
8241 P. E. Nielan
8241 K. J. Perano
8242 M. R. Birnbaum
8242 V. D. Revelli
8242 L. I. Weingarten
8243 M. L. Callabresi
8243 D. J. Bammann
8244 C. M. Hartwig
8245 R. J. Kee
8524 J. A. Wackerly