

# **PORPST: A Statistical Postprocessor for the PORMC Computer Code User's Manual, Version 1.0**

Prepared for the U.S. Department of Energy  
Office of Environmental Restoration and  
Waste Management



**Westinghouse**  
**Hanford Company** Richland, Washington

Hanford Operations and Engineering Contractor for the  
U.S. Department of Energy under Contract DE-AC06-87RL10930

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Approved for Public Release

**LEGAL DISCLAIMER**

---

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use 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 product, 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 any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

---

This report has been reproduced from the best available copy. Available in paper copy and microfiche.

Available to the U.S. Department of Energy  
and its contractors from  
Office of Scientific and Technical Information  
P.O. Box 62  
Oak Ridge, TN 37831  
(615) 576-8401

Available to the public from the U.S. Department of Commerce  
National Technical Information Service  
5285 Port Royal Road  
Springfield, VA 22161  
(703) 487-4650

Printed in the United States of America

DISCLM-1.CHP (1-91)

# INFORMATION RELEASE REQUEST

References:  
WHC-CM-3-4

of Release	Purpose		New ID Number <b>WHC - EP - 0423</b>	
	of Presentation	(Check only one suffix)	<input type="checkbox"/> Speech or Presentation	<input type="checkbox"/> Reference
			<input type="checkbox"/> Full Paper	<input type="checkbox"/> Technical Report
			<input type="checkbox"/> Summary	<input type="checkbox"/> Thesis or Dissertation
<input type="checkbox"/> Abstract	<input type="checkbox"/> Manual	Existing ID Number (include revision, volume, etc.)		
<input type="checkbox"/> Visual Aid	<input type="checkbox"/> Brochure/Flier	If previously cleared, list ID number		
<input type="checkbox"/> Speakers Bureau	<input checked="" type="checkbox"/> Software/Database	Date Release Required <b>June 28, 1991</b>		
<input type="checkbox"/> Poster Session	<input type="checkbox"/> Controlled Document	Title <b>PORPST: A statistical Postprocessor for the PORMC computer Code (Users Guide Version 1.0)</b>		
<input type="checkbox"/> Videotape	<input type="checkbox"/> Other	Unclassified Category <b>UC-</b>		
Title of Journal <b>N/A</b>		Group or Society Sponsoring		
Date(s) of Conference or Meeting		City/State		
Title of Conference or Meeting		Will proceedings be published? <input type="checkbox"/> Yes <input type="checkbox"/> No		
		Will material be handed out? <input type="checkbox"/> Yes <input type="checkbox"/> No		
		Impact Level <b>4</b>		

### CHECKLIST FOR SIGNATORIES

Review Required per WHC-CM-3-4	Yes	No	Reviewer Name (printed)	Signature	Date
Classification/Unclassified Controlled Nuclear Information	<input type="checkbox"/>	<input checked="" type="checkbox"/>			
Patent - General Counsel	<input checked="" type="checkbox"/>	<input type="checkbox"/>	CAROL Disibio	<i>Carol K. Disibio</i>	6-19-91
Legal - General Counsel	<input checked="" type="checkbox"/>	<input type="checkbox"/>	CAROL Disibio	<i>Carol K. Disibio</i>	6-19-91
Applied Technology/Export Controlled Information or International Program	<input type="checkbox"/>	<input checked="" type="checkbox"/>			
WHC Program	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<i>See below for signature J.C. Sonnichsen</i>		
Communications	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<i>RECEIVED DOE-RL</i>		
DOE-RL Program	<input checked="" type="checkbox"/>	<input type="checkbox"/>	M.J. Furman	<i>M. J. Furman</i>	JUN 18 1991 6/26/91
Publications Services	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Leslie A. Brown	<i>Leslie A. Brown</i>	6/27/91
Other Program	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Reviewed As PNL-7698	contact: Dr. M.J. Fayer	
References Available to Intended Audience	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Leslie A. Brown	<i>Leslie A. Brown</i>	6/27/91
Transmit to DOE-HQ/Office of Scientific and Technical Information	<input type="checkbox"/>	<input checked="" type="checkbox"/>			

Information conforms to all applicable requirements. The above information is certified to be correct. *See comments on draft release report and all*

Author/Requestor (Printed/Signature) <b>J.C. Sonnichsen, Jr.</b> <i>John C. Sonnichsen, Jr.</i>	Date <b>6/11/91</b>	INFORMATION RELEASE ADMINISTRATION APPROVAL STAMP Stamp is required before release. Release is contingent upon resolution of mandatory comments. 
Responsible Manager (Printed/Signature) <b>J.C. Sonnichsen, Jr.</b> <b>J.W. Cammann</b> <i>John C. Sonnichsen, Jr.</i>	Date <b>6/11/91</b>	
Intended Audience <input type="checkbox"/> Internal <input type="checkbox"/> Sponsor <input checked="" type="checkbox"/> External		DISTRIBUTION STATEMENT IS UNLIMITED Date Received <b>6/12/91</b>

WHC-EP--0423

DE92 002105

# **PORPST: A Statistical Postprocessor for the PORMC Computer Code**

**User's Manual, Version 1.0**

Prepared by:

**P. W. Eslinger  
B. T. Didier**  
Pacific Northwest Laboratory

Date Published  
June 1991

Prepared for the U.S. Department of Energy  
Office of Environmental Restoration and  
Waste Management



**Westinghouse  
Hanford Company**

P.O. Box 1970  
Richland, Washington 99352

Hanford Operations and Engineering Contractor for the  
U.S. Department of Energy under Contract DE-AC06-87RL10930

**MASTER**

*JMB*  
DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Approved for Public Release

**PORPST: A STATISTICAL POSTPROCESSOR FOR THE  
PORMC COMPUTER CODE**

**USER'S MANUAL, VERSION 1.0**

**P. W. Eslinger  
B. T. Didier  
Pacific Northwest Laboratory**

**ABSTRACT**

*This report describes the theory underlying the PORPST code and gives details for using the code. The PORPST code is designed to do statistical postprocessing on files written by the PORMC computer code. The data written by PORMC are summarized in terms of means, variances, standard deviations, or statistical distributions. In addition, the PORPST code provides for plotting of the results, either internal to the code or through use of the CONTOUR3 postprocessor.*

*Section 2.0 discusses the mathematical basis of the code, and Section 3.0 discusses the code structure. Section 4.0 describes the free-format input command language. Section 5.0 describes in detail the commands to run the program. Section 6.0 provides an example program run, and Section 7.0 provides the references.*

This page intentionally left blank.

CONTENTS

1.0	INTRODUCTION . . . . .	1
1.1	MOTIVATION . . . . .	1
1.2	MAJOR FEATURES OF PORPST . . . . .	1
	1.2.1 Data Selection . . . . .	2
	1.2.2 Statistical Processing . . . . .	2
	1.2.3 Plotting Options . . . . .	2
	1.2.4 Tabular Output Options . . . . .	2
1.3	PURPOSE . . . . .	3
2.0	MATHEMATICAL BASIS . . . . .	3
2.1	DATA TRANSFORMATIONS . . . . .	3
2.2	SUMMARY STATISTICS . . . . .	3
2.3	ESTIMATION OF STATISTICAL DISTRIBUTIONS . . . . .	4
	2.3.1 Empirical Distribution Functions . . . . .	5
	2.3.2 Histogram Density Estimates . . . . .	5
	2.3.3 Kernel Density Estimates . . . . .	5
	2.3.4 Cumulative Distribution Functions from a Kernel Density Estimate . . . . .	6
	2.3.5 Practical Use Considerations . . . . .	6
3.0	CODE STRUCTURE AND PROGRAM EXECUTION . . . . .	7
3.1	INPUT AND OUTPUT FILES . . . . .	7
	3.1.1 Description of the Input Files . . . . .	7
	3.1.1.1 Keyword Control File . . . . .	8
	3.1.1.2 PORMC Data Files . . . . .	8
	3.1.2 Description of the Output Files . . . . .	8
	3.1.2.1 Report File . . . . .	8
	3.1.2.2 Contour File . . . . .	8
	3.1.2.3 DISSPLA Metafile . . . . .	8
3.2	RUNNING THE PROGRAM . . . . .	9
3.3	PORPST MODULES AND THEIR FUNCTIONS . . . . .	9
3.4	VARIABLE DEFINITIONS AND DIMENSION PARAMETERS . . . . .	9
3.5	ERROR MESSAGES . . . . .	14
4.0	DESCRIPTION OF FREE-FORMAT COMMAND LANGUAGE . . . . .	15
4.1	RECORD TYPES . . . . .	15
	4.1.1 The Keyword Record . . . . .	15
	4.1.2 Continuation Records . . . . .	16
	4.1.3 Comment Records . . . . .	16
4.2	ELEMENTS OF THE INPUT RECORD . . . . .	17
	4.2.1 The Keyword . . . . .	17
	4.2.2 The Modifier . . . . .	17
	4.2.3 The Numeric Field . . . . .	18
	4.2.4 The Separator Field . . . . .	19
	4.2.5 The Terminator Field . . . . .	20
	4.2.6 The Comment Field . . . . .	20
5.0	DETAILED DESCRIPTION OF KEYWORD COMMANDS . . . . .	21
5.1	GENERAL INSTRUCTIONS FOR COMBINING KEYWORDS . . . . .	22
5.2	CLEAR COMMAND . . . . .	24
	5.2.1 Purpose . . . . .	24

## CONTENTS (cont)

	5.2.2	Syntax . . . . .	24
	5.2.3	Examples . . . . .	24
5.3		CORRELAT COMMAND . . . . .	24
	5.3.1	Purpose . . . . .	24
	5.3.2	Syntax . . . . .	26
	5.3.3	Example . . . . .	26
5.4		DISTRIBU COMMAND . . . . .	26
	5.4.1	Purpose . . . . .	26
	5.4.2	Syntax . . . . .	26
	5.4.3	Examples . . . . .	27
5.5		END COMMAND . . . . .	28
	5.5.1	Purpose . . . . .	28
	5.5.2	Syntax . . . . .	28
	5.5.3	Examples . . . . .	28
5.6		ENDINPUT COMMAND . . . . .	28
	5.6.1	Purpose . . . . .	28
	5.6.2	Syntax . . . . .	28
	5.6.3	Examples . . . . .	28
5.7		FILE COMMAND . . . . .	29
	5.7.1	Purpose . . . . .	29
	5.7.2	Syntax . . . . .	29
	5.7.3	Examples . . . . .	29
5.8		MEAN COMMAND . . . . .	29
	5.8.1	Purpose . . . . .	29
	5.8.2	Syntax . . . . .	31
	5.8.3	Examples . . . . .	31
5.9		PLOT COMMAND . . . . .	31
	5.9.1	Purpose . . . . .	31
	5.9.2	Syntax . . . . .	32
	5.9.3	Examples . . . . .	32
5.10		REALIZAT COMMAND . . . . .	33
	5.10.1	Purpose . . . . .	33
	5.10.2	Syntax . . . . .	34
	5.10.3	Examples . . . . .	34
5.11		SCREEN COMMAND . . . . .	34
	5.11.1	Purpose . . . . .	34
	5.11.2	Syntax . . . . .	34
	5.11.3	Examples . . . . .	35
5.12		STOP COMMAND . . . . .	35
	5.12.1	Purpose . . . . .	35
	5.12.2	Syntax . . . . .	35
	5.12.3	Examples . . . . .	35
5.13		SUMMARY COMMAND . . . . .	36
	5.13.1	Purpose . . . . .	36
	5.13.2	Syntax . . . . .	36
	5.13.3	Examples . . . . .	36
5.14		TABLES COMMAND . . . . .	36
	5.14.1	Purpose . . . . .	36
	5.14.2	Syntax . . . . .	37
	5.14.3	Examples . . . . .	38
5.15		TIMES COMMAND . . . . .	38

CONTENTS (cont)

5.15.1	Purpose	38
5.15.2	Syntax	39
5.15.3	Examples	39
5.16	TITLE COMMAND	39
5.16.1	Purpose	39
5.16.2	Syntax	40
5.16.3	Examples	40
5.17	TRANSFOR COMMAND	40
5.17.1	Purpose	40
5.17.2	Syntax	40
5.17.3	Examples	41
5.18	USER COMMAND	41
5.18.1	Purpose	41
5.18.2	Syntax	41
5.18.3	Examples	41
5.19	VARIANCE COMMAND	41
5.19.1	Purpose	41
5.19.2	Syntax	41
5.19.3	Examples	42
5.20	WINDOW COMMAND	42
5.20.1	Purpose	42
5.20.2	Syntax	43
5.20.3	Examples	43
5.21	XLABEL COMMAND	44
5.21.1	Purpose	44
5.21.2	Syntax	44
5.21.3	Examples	44
5.22	YLABEL COMMAND	44
5.22.1	Purpose	44
5.22.2	Syntax	44
5.22.3	Examples	44
6.0	EXAMPLE RUN	45
7.0	REFERENCES	47

FIGURE

1	Module Calling Hierarchy for the PORPST Code	13
---	--	----

TABLES

1	Modules and Their Functions	10
2	Parameter Definitions in the PORPST Code	14

TABLES (cont)

3	Keywords and Their Functions for the PORPST Code . . . . .	23
4	Modifiers and Their Description for the CLEAR Keyword. . . . .	25
5	Modifiers and Their Descriptions for the DISTRIBU Keyword . . . . .	27
6	Modifiers, Their Descriptions, and Numerical Values for the FILE Keyword . . . . .	30
7	Modifiers and Their Descriptions for the MEAN Keyword . . . . .	31
8	Modifiers, Their Descriptions, and Numeric Fields for the PLOT Keyword . . . . .	33
9	Modifiers, Their Descriptions, Numerical Fields, and Default Values for the REALIZAT Keyword . . . . .	34
10	Modifiers and Their Descriptions for the SCREEN Keyword . . . . .	35
11	Numeric Modifiers and Their Descriptions for the SUMMARY Keyword . .	36
12	Modifiers, Their Descriptions, and Numerical and Default Values for the TABLES Keyword . . . . .	38
13	Modifiers and Their Descriptions for the TIMES Keyword . . . . .	39
14	Modifiers and Their Descriptions for the TRANSFOR Keyword . . . . .	40
15	Modifiers and Their Descriptions for the VARIANCE Keyword . . . . .	42
16	Numeric Modifiers, Their Descriptions, and Default Values for the WINDOW Keyword . . . . .	43
17	Input Data File: EXAMPLE1.DAT . . . . .	46

**PORPST: A STATISTICAL POSTPROCESSOR FOR THE  
PORMC COMPUTER CODE**

**USER'S MANUAL, VERSION 1.0**

**1.0 INTRODUCTION**

**1.1 MOTIVATION**

The discharge of radioactive and chemical wastes to the subsurface has been a widespread and common waste disposal practice at U.S. Department of Energy (DOE) installations. Some of the waste discharges have been planned; others have been accidental. Needs for future discharges have been identified. For example, future disposal of commercially derived, high-level nuclear wastes is planned at the Yucca Mountain Site in southern Nevada. The potential for disposal of low-level radioactive and mixed (radioactive and chemical) wastes is being examined at the Hanford Site in southeastern Washington State and at other DOE sites.

To evaluate the safety of the proposed actions and the effectiveness of methods proposed for cleanup of sites contaminated by past disposal practices, predictions of the future performance of waste isolation systems must be made. Mathematical models are analytical tools used to make these predictions and evaluations. The PORFLO-3 (Sagar and Runchal 1990) code incorporates one such mathematical model.

The PORMC computer code (WHC 1991) has embedded the PORFLO-3 code in the framework of a Monte Carlo driver to allow easier analysis of the effects of uncertainties in input data on a performance measure (i.e., transport of a contaminant). The driver provides for generation of statistically independent or dependent values for key hydrological parameters. When using a Monte Carlo code such as PORMC, one quickly obtains a large volume of output data that requires further processing for visualization and comprehension. The PORPST code is a postprocessor for PORMC that aids in the visualization process.

**1.2 MAJOR FEATURES OF PORPST**

The PORMC computer code can operate in an iterative (Monte Carlo) mode. In any single iteration, the code solves a complex problem involving a combination of pressure, temperature, and concentration equations. Data may be written to output data files at several time steps within each iteration, and output files can be produced for a large number of variables. Each output data file contains information over the entire computational grid for every iteration.

The PORPST code postprocesses the data files written by the PORMC code. The major features of the PORPST code are described in the following subsections.

### 1.2.1 Data Selection

A major feature of the PORPST code is the ability to select subsets of the data in the PORMC data files. Data can be extracted by realization number or by time slices within realizations. In addition, a window capability allows processing on a subset of the physical model domain.

### 1.2.2 Statistical Processing

Another major function of the PORPST code is to compute summary statistics based on the PORMC data file. Statistics implemented include the mean and variance of individual variables and the correlation between two variables. In addition to the summary statistics, a statistical distribution can be generated in the form of either a probability density function (PDF) estimate or a cumulative distribution function (CDF) estimate. For example, the CDF describing the variability of contaminant concentration at a single location can be generated. The data selection features can be invoked to select the data of primary interest for each statistical processing function.

### 1.2.3 Plotting Options

The PORPST code allows the generation of plots for most of the data-processing functions. First, those operations that have results that can be summarized in the form of a line plot can be plotted on the screen, or a graphics metafile can be generated. Examples of these operations are generating a PDF or CDF or examining the value of a variable at a location over the set of realizations.

In addition to line plots, the PORPST code will process data for contour plots. The contour plots are not actually generated by PORPST; instead, a file is written that can be used in the CONTOUR3 plotting program. The CONTOUR3 program was developed to postprocess files written by the PORFLO-3 code.

### 1.2.4 Tabular Output Options

Another major option of the code is to output values in the form of tables. A typical use of this feature would be to examine explicitly the value of some variable (e.g., hydraulic head) at some location over the set of realizations.

In addition to the tabular data output, the program can scan a data file and print a one-page summary of the data. This feature is useful in regenerating information about the specific PORMC run.

### 1.3 PURPOSE

This document describes the theory underlying the PORPST code and gives details for using the code. The PORPST code provides a tool for performing statistical postprocessing on the data files generated by the PORMC code.

## 2.0 MATHEMATICAL BASIS

The PORPST code implements commonly accepted algorithms to compute a number of statistics. Options for data transformations are provided, and the statistics are computed after the transformations are performed. Three of the statistics are summary in nature, providing for the calculation of the mean value or variance of a single variable, or the (Pearson product moment) correlation between two variables. The other statistics deal with generation of a statistical density or CDF. Algorithms for each of the statistics are given in this section.

### 2.1 DATA TRANSFORMATIONS

Version 1.0 of the PORPST code implements two options for data transformations. The two options are logarithms using base e or base 10. The transformations are the first data-processing step performed. Either of these transformation options will cause the current problem definition to be skipped if any of the data values in the data window (Section 5.0) is zero or negative.

### 2.2 SUMMARY STATISTICS

The mean and variance calculations are performed on values from a single variable (e.g., hydraulic conductivity or pressure), while the correlation calculation requires two variables. Let  $\{x_i:i=1,n\}$  denote the  $n$  values from the first variable, and let  $\{y_i:i=1,n\}$  denote the  $n$  values from the second variable.

For notational convenience, let

$$\begin{aligned} SX &= \sum_{i=1}^n x_i, & SXX &= \sum_{i=1}^n x_i^2 x_i, \\ SY &= \sum_{i=1}^n y_i, & SY Y &= \sum_{i=1}^n y_i^2 y_i, \text{ and} \\ SXY &= \sum_{i=1}^n x_i^2 y_i. \end{aligned}$$

The mean value  $E(x)$ , also called the statistical expectation, and variance  $V(x)$  of the variable  $x$  (the same formulas apply to the variable  $y$ ) are computed using the expressions

$$E(x) = sx / n$$

and

$$V(x) = (sxx - sx*sx/n) / (n-1).$$

The variance is undefined if only a single sample value ( $n=1$ ) is available. This is reasonable intuitively because the variance is interpreted as the spread in the data about the central value  $E(x)$ , and a single point gives no measure of spread. The code also allows computation of the standard deviation of the data. The standard deviation is defined to be the positive square root of the variance.

The correlation  $C(x,y)$  between the two variables  $x$  and  $y$  is computed using the expression

$$C(x,y) = [(sxy - sx*sy/n)/(n-1)] / \sqrt{[V(x)*V(y)]}.$$

The correlation between two variables is undefined if one or both variables have a zero variance. A zero variance is obtained only if all  $n$  values of the variable are identical. Computation of this statistic requires the sample size  $n$  to be 2 or larger.

### 2.3 ESTIMATION OF STATISTICAL DISTRIBUTIONS

The PORPST code provides for the computation of statistical distribution descriptions in the form of PDF estimates or CDF estimates. The computational options are discussed in the following two sections.

The most important assumption made when computing statistical distributions is that the data values form a statistically random sample. If this assumption is violated, the computed CDF may not be representative of the statistical distribution actually defining the physical phenomenon that was sampled. The computed CDF will always have the usual properties (continuous, nonnegative, and integrate to unity), even if the random-sample assumption is violated.

Most elementary statistics textbooks (e.g., Strait 1989, p. 330) discuss the concept of a random sample. The essence of the random-sample concept is that all values are collected under identical conditions and are identified with the same governing probability distribution.

### 2.3.1 Empirical Distribution Functions

Consider the case in which  $n$  data have been collected (a random sample). The data are assumed to come from some unknown distribution with CDF  $F(x)$ . The empirical CDF  $EF(x)$  (Mood et al. 1974, p. 264) estimates the population CDF using the sample data. The empirical CDF is defined as follows

$$EF(x) = N(x) / n$$

where  $N(x)$  is the number of data values that are less than or equal to  $x$ . The empirical CDF is defined for the entire real line. If each of the data values is unique,  $EF(x)$  has jumps of height  $1/N$  at each of the data locations. If  $k$  of the data values have the same value  $x$  (i.e.,  $k$  ties are in the data set at location  $x$ ), the jumps in  $EF(x)$  have height  $k/N$  at location  $x$ .

### 2.3.2 Histogram Density Estimates

The simplest method used for computing sample-based PDFs is to compute a histogram  $H(x)$  based on the sample data. Given a partition on a contiguous subset of the real line that contains  $m-1$  line segments defined by the set of endpoints  $\{b_k, k=1, m\}$ , the histogram is defined as

$$H(z; b_k, b_{k+1}) = N(b_k, b_{k+1}) / n$$

for all location  $z$  in the interval  $(b_k, b_{k+1})$ , where  $N(b_k, b_{k+1})$  denotes the number of sample points that fall in the interval  $(b_k, b_{k+1})$ . The  $b_k$ 's are typically called bin boundaries.

The implementation in the PORPST code uses bins with equal widths. Thus, the user needs only to specify a lower bound ( $b_1$ ) and the distance between the bin boundaries.

### 2.3.3 Kernel Density Estimates

A class of PDF estimators called kernel estimators was considered by Parzen (1962). The kernel estimator has the form

$$g(z) = \frac{1}{nC_N s_N} \sum_{i=1}^N w[(z-x_i)/(c_N s_N)]$$

where

$\{c_N\}$  = a sequence of constants converging to zero at an appropriate rate

$s_N$  = a scale (standard deviation) estimator

$w$  = a smooth density on the real line

$x_i$  = a sample data value.

For application purposes,  $w$  is defined as

$$w(z) = \begin{cases} 0.75(1-z^2) & \text{if } |z| < 1 \\ 0 & \text{otherwise.} \end{cases}$$

This kernel has been shown to have optimal mean-squared-error properties for density estimation (Epanechnikov 1969). The scale estimator used is the median absolute deviation divided by 0.6745, as suggested by Beran (1977). The sequence  $\{c_N\}$  chosen is  $c_N = 2.283 \cdot N^{-.287}$ , which is based on work by Eslinger and Woodward (1990).

#### 2.3.4 Cumulative Distribution Functions from a Kernel Density Estimate

The CDF of the variable  $X$  at the location  $x$  is defined to be the integral of the PDF over the range  $-\infty$  to  $x$ . An approach to finding a sample-based CDF is to generate a PDF estimate based on the sample values and then integrate it to obtain the CDF.

The PORPST code uses the sample data and the function  $g(z)$  defined in Section 2.3.3 to generate the sample PDF at a large number of  $z$  values. This set of PDF values is then integrated using a trapezoidal rule to obtain the sample-based CDF.

#### 2.3.5 Practical Use Considerations

Both of the CDF estimation procedures are "consistent," in that the sample-based CDF converges to the theoretical CDF when a large sample is drawn from some specified probability distribution (Serfling 1980, p. 57; Devroye 1987, p. 37). In addition, convergence rates based on asymptotic arguments are available from the same references. The code requires at least four data values to successfully generate CDF plots. More than four values are desirable because the procedure estimates the shape of an entire function. As a general rule, at least 20 values are required to obtain a representative sample-based CDF. It is preferable to have 100 or more values.

The two CDF estimation procedures have subtle differences. In general, the empirical CDF approach applies a probability mass of size  $1/N$  at the discrete locations corresponding to the data locations, while the kernel estimation approach spreads the same probability mass over a small region centered at the data location. No definite theoretical explanation exists to substantiate the preference of one approach to the other. The author generally prefers the kernel-based approach to the empirical-based approach.

The histogram (PDF) or empirical-based (CDF) approaches may be preferable when some data are obtained "close" to a region eliminated by physical model considerations (e.g., porosity is constrained between 0 and 1). The kernel-based approach spreads the probability mass associated with a data point over a small region; thus, it may assign a non-zero probability to an "impossible" region. The empirical-based CDF is confined between the smallest and largest data values, thus assigns zero probability to impossible regions. Given a properly constructed set of bin boundaries, the histogram approach does not encounter this problem.

The histogram PDF is useful for comparisons by authors who present results in order-of-magnitude jumps. The kernel PDF can be plotted using order-of-magnitude axes labels, but the interpretation of the plot will not be the same as for the histogram PDF. In the asymptotic sense, both the kernel-based and histogram-based PDF estimators are consistent. However, a bin width selection that supports consistency for the histogram-based PDF estimator is not readily apparent to the user. This may not be a serious limitation because the computation time required by the PORMC code precludes generating extremely large sample sizes.

### 3.0 CODE STRUCTURE AND PROGRAM EXECUTION

#### 3.1 INPUT AND OUTPUT FILES

The PORPST program uses several input and output files. The PORPST code may use six different files at one time. Only one of these files is restricted in its naming by the software. This is the DISSPLA metafile, which is created by default when performing line plots. The name of the metafile is dependent on the host system. The PORPST program contains only one user prompt. This prompt is for the name of the keyword control file. The PORMC output files (STOCH.\* files, where the \* denotes any valid suffix) are used as the input data files to PORPST. The PORPST program always writes a report file and can optionally create a file acceptable to the CONTOUR3 contouring program. This last file is referred to as a contour file in this document. The report file, the input data files, and the contour file are identified by using the FILE keyword command. A more detailed description of the files follows.

##### 3.1.1 Description of the Input Files

Three input files can be used by the PORPST program. The first input file is a keyword file containing program control statements, and the second and third input files contain data for further processing. Multiple problem definitions can be made in a single keyword control file, and each problem definition can use up to two input data files. The user is required to close the input data files (see the FILE or CLEAR keywords) used in a problem definition before new input files can be selected for a new problem definition. Because this programming convention has been implemented, the discussion of the input data files will consider only two files.

**3.1.1.1 Keyword Control File.** The first input file is the keyword control file that contains sequences of free-format keyword commands to control the operation of PORPST. This file is a formatted [American Standard Code for Information Interchange (ASCII)] file that the user must generate using an editing program. The user is prompted for the name of this file by PORPST. The free-format command language is described in Section 4.0, and a detailed description of the keyword-controlled options is given in Section 5.0.

**3.1.1.2 PORMC Data Files.** The remaining two input files are assumed to be files created by the PORMC program (the STOCH.\* files). These files contain the data to be operated on by the defined problem. Either one or two input data files are required for any specific problems defined. Generally, two data files can be used only in a single problem definition when computing correlations or when file summaries are being performed. These files can be either formatted (ASCII characters) or unformatted (binary). The user provides file names by using the **FILE** keyword.

The two input data files must be written using a prescribed format. Subroutine **HEADER** is used to read the header from the data files, and subroutine **RDVAL** is used to read data slices. These two subroutines contain comments that describe the format requirements of the data file. The user typically will use the PORMC code to create the data files and thus will not need to know the specific format requirements for these files.

### 3.1.2 Description of the Output Files

Three output files can be written by the PORPST code. The three files are described in the following subsections.

**3.1.2.1 Report File.** The first output file, called the report file, is written for every problem defined in the keyword control file. The report file is an ASCII character file and is controlled by the user with the **FILE** keyword. A new report file can be used for each problem defined in the keyword control file, or the same report file can be used for multiple problems defined in the keyword control file. The report file typically will contain the program identification, information about the problem definition, any file summaries, error and warning messages, and any created tables.

**3.1.2.2 Contour File.** The second output file, called the contour file, contains the grid of computed values in a format acceptable to the **CONTOUR3** contouring program. This file is written only if the **OUTPUT** option of the **MEAN**, **CORRELATION**, or **VARIANCE** keyword is used. This file is an ASCII character file to allow easy transfer between computer systems. A new contour file must be used for each problem that writes output to the contour file because of the possibility of different grid sizes, coordinate systems, etc., between the two defined problems.

**3.1.2.3 DISSPLA Metafile.** The third output file is the **DISSPLA** metafile. This file is created if line plots are created by the defined problem and sent to this file (the default) instead of being plotted on the screen. This file is a binary file. The name of the metafile is dependent on the host system.

If using the CRAY,<sup>1</sup> the user types POPFIL; if using the MicroVAX,<sup>2</sup> the user types DISSPLA.POP. A DISSPLA postprocessor is used to condition the metafile for output on various graphics devices. The name of the DISSPLA postprocessor is dependent on the host computer system.

### 3.2 RUNNING THE PROGRAM

The PORPST code can run in an interactive or batch mode. Once the code has been compiled and linked into an executable form on a particular computer, the user can execute the program. To run the program interactively, the user types 'PORPST' on the CRAY or 'RUN PORPST' on a VAX. The code then displays a few lines of information and prompts the user for the keyword control file name. Assume that the user has created the keyword control file EXAMPLE1.DAT, which is listed in Section 6.0. The user would then enter the text string EXAMPLE1.DAT and press the ENTER key. The PORPST code would then run the problems defined in EXAMPLE1.DAT. For batch-mode processing, the user should confer with a system consultant on creating a command file to operate PORPST.

### 3.3 PORPST MODULES AND THEIR FUNCTIONS

The PORPST code has 51 modules, in which a module is a FORTRAN subroutine, function, or the main program. To the extent possible, each module has been assigned a single, distinct function. Subroutine IDENC is the only subroutine that is dependent on the machine or operating system used. Versions of this subroutine are provided for the CRAY and MicroVAX II systems. The modules are given in Table 1 with a brief description of their major code function. The main program, PORPST, is listed first; the other modules follow in alphabetical order.

Figure 1 shows the calling hierarchy for the modules. Some utility subroutines (e.g., RUSE, SEPCHR, and XFKEY) are called from more than one routine.

### 3.4 VARIABLE DEFINITIONS AND DIMENSION PARAMETERS

The PORPST code was designed so that all global variables are contained in common blocks. A description of the variables will not be given here because they are defined in internal comments in the source code.

The PORPST code uses PARAMETER statements to define the dimensions of various arrays. Depending on the problem that is run, the user may need to change the dimension of some arrays. The parameters that may be subject to change and their definitions are given in Table 2.

---

<sup>1</sup>CRAY is a trademark of Cray Research Inc., Minneapolis, Minnesota.

<sup>2</sup>VAX is a trademark of the Digital Equipment Corporation.

Table 1. Modules and Their Functions. (sheet 1 of 3)

Module	Description
PORPST	Main (top-level) module for the program. Provides the basic structure that calls modules to read problem definitions, read input files, perform computations, and write results.
ASCORR	Subroutine that assembles the results from the correlation computation into the appropriate variables to be used for output (table, line plot, or contour file).
ASDIST	Subroutine that assembles the results from the distribution computation into the appropriate variables to be used for output (table or line plot).
ASMEAN	Subroutine that assembles the results from the mean value computation into the appropriate variables to be used for output (table, line plot, or contour file).
ASVARI	Subroutine that assembles the results from the variance computation into the appropriate variables to be used for output (table, line plot, or contour file).
ASNCLB	Subroutine that assigns numeric values to the column labels used in table output.
ASNPLB	Subroutine that assigns numeric values to the plane labels used in table output.
ASNRLB	Subroutine that assigns numeric values to the row labels used in table output.
CEXIST	Part of the set of routines called by RDBLK to decode keyword commands. This function determines if an eight-letter keyword is one of the modifiers for the current data record.
COMCHR	Part of the set of routines called by RDBLK to decode keyword commands. This function determines if an input character is a comment identifier.
CONTUR	Subroutine that writes header information and computed grid values to the contour file in a format acceptable to the CONTOUR3 contouring program.
CORREL	Subroutine that computes correlations between two variables.
EPAN	Subroutine that uses the kernel density estimation technique to compute a distribution in both density and cumulative form.
ERRCHK	Subroutine that performs extensive checking to ensure that a valid problem has been defined and that the problem can be performed on the data selected.
FINSEP	Part of the set of routines called by RDBLK to decode keyword commands. This subroutine locates all the separators in the current input record.
FNAME	Subroutine that extracts a file name (character string) from an input card image.

Table 1. Modules and Their Functions. (sheet 2 of 3)

Module	Description
GKERN	Function that evaluates the kernel density estimate at a single location.
GPLOTX	Subroutine that generates a basic line plot for either the Tektronix 4014 screen or the DISSPLA metafile.
HEADER	Subroutine that reads the header information from the input data files (the STOCH.* files from PORMC).
IDENC	Subroutine that generates identification variables for each run of the code. This subroutine is operating-system dependent.
LOCNP	Subroutine that converts an array index specified by three dimensions into an array index specified by a single dimension. It also performs the inverse transformation.
MEAN	Subroutine that computes mean values for a single variable.
MKTABL	Subroutine that transposes a three-dimensional grid of computed values into a specified (or default) orientation for use as table output.
NEWPGE	Subroutine that writes a FORTRAN new-page control character.
OUTCLB	Subroutine that writes a given set of column labels.
OUTHED	Subroutine that writes a given line of character data.
OUTLNE	Subroutine that writes a given row label and a given single line of values.
PRSLTS	Subroutine that provides control for the output of results from PORPST.
PRTTAB	Subroutine that provides control for the output of tables from PORPST.
RDBLK	Subroutine that is the controlling module of the set of modules that decode keyword data.
RELEXS	Function that determines if a requested data slice is present in the data file.
RESET	Subroutine that initializes control information for the specified keyword(s).
RUSE	Subroutine that determines if a given data slice is to be included in the computations.
SAVTTL	Part of the set of routines called by RDBLK to decode keyword commands. This subroutine moves the contents of the input record, following the first separator, into an output array.
SCALE	Subroutine that computes the scaled median absolute deviation for the data set.

Table 1. Modules and Their Function. (sheet 3 of 3)

Module	Description
SCNFIL	Subroutine that reads through the input data file and accumulates summary information.
SEPCHR	Part of the set of routines called by RDBLK to decode keyword commands. This function determines if a single input character is a separator character.
SETUP	Subroutine that performs program initialization.
SORT	Subroutine that performs an algebraic sort on a data vector.
SUMPAG	Subroutine that prints summary information of a given data file.
TMATCH	Function that determines if two real values are equivalent within a given tolerance.
TRANS	Subroutine that performs transformations on the input data.
UPCASE	Part of the set of routines called by RDBLK to decode keyword commands. This subroutine converts alphabetic characters to upper case.
VARIAN	Subroutine that computes variances for a single variable.
XFKEY	Part of the set of routines called by RDBLK to decode keyword commands. This subroutine transfers keywords from an input record to a storage location.
ZEROD	Subroutine that initializes a double-precision array to zero.

Figure 1. Module Calling Hierarchy for the PORPST Code.

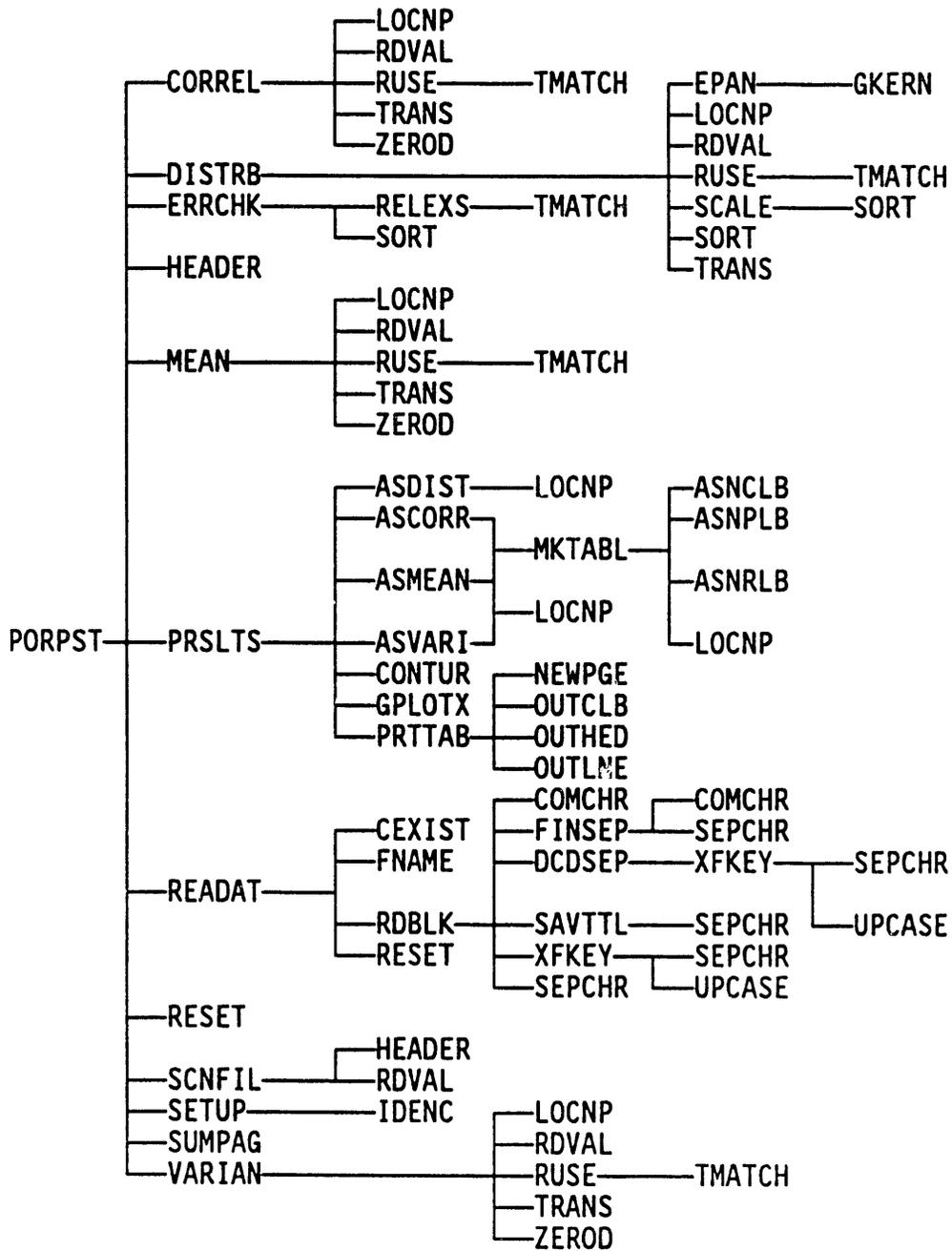


Table 2. Parameter Definitions in the PORPST Code.

Parameter	Value	Definition
MAXINT	1000	Maximum number of integration steps that can be used with the kernel density method for generating a statistical distribution.
MAXTMS	1000	Maximum number of data slices that can be read from a data file and provided in a data file summary. This parameter is the sum of the number of data slices that exist for each realization.
MXPLOT	2000	Maximum number of values that can be used in a line plot.
NKEYS	50	Maximum number of modifiers that can be contained on a single keyword record, including any associated continuation records.
NVALS	20	Maximum number of numerical values that can be contained on a single keyword record, including any associated continuation records.
PNI	10	Maximum number of nodes in the I (X) dimension. Because of concerns of limiting the amount of unused allocated storage, this parameter has been made to fit the current problem and will generally change as the user's problem changes.
PNJ	10	Maximum number of nodes in the J (Y) dimension. Because of concerns of limiting the amount of unused allocated storage, this parameter has been made to fit the current problem and will generally change as the user's problem changes.
PNVMAX	50	Maximum number of variables used by the PORMC code. Limited information is contained in the data file headers on all PORMC variables.
PNZ	200	Maximum number of nodes in the K (Z) dimension. Because of concerns of limiting the amount of unused allocated storage, this parameter has been made to fit the current problem and will generally change as the user's problem changes.

### 3.5 ERROR MESSAGES

The PORPST code performs extensive checking on keyword inputs, valid problem definitions, and appropriate input data for a defined problem. The PORPST code generates two levels of error messages. The first level is an informational or warning message, used to inform the user that an unexpected event occurred and the program has performed a corrective action. The program is able to continue processing the defined problem. If the default action is inappropriate, then the user must alter the problem definition and rerun the problem. The second level is a fatal-error level. Depending on the severity of the error, either just the current problem definition is terminated or the

entire PORPST program is terminated. Fatal-error messages identify the subroutine that detected the error and indicate the user action necessary to correct the error. Both the informational messages and the fatal-error messages are written to the report file. The error messages are not discussed further because the run-time error messages give explicit instructions for correcting problems that are detected.

#### 4.0 DESCRIPTION OF FREE-FORMAT COMMAND LANGUAGE

The user interface with PORPST is through a free-format command language. This command language reduces user input to a set of conversational, English-like commands. In most cases, the commands are largely free of any requirements of format and hierarchy.

The free-format data input routine (ADATA) used in PORFLO and PORMC was developed by A. K. Runchal (Runchal and Sagar 1989). A similar functionality in input structure was desired for PORPST because it is a postprocessor for PORMC. The RDBLK set of routines were developed separately by Westinghouse Hanford to provide this functionality. Although the routines are independent, the input conventions for PORPST are almost identical to those used in PORFLO. As such, the description that follows was taken from the PORFLO User's Guide (Runchal and Sagar 1989) and is included herein with only minor modification.

The free-form language reads the input data in distinct blocks. Each block begins with a keyword (Section 4.1.1) that identifies the nature of the data within the block. The character and numeric data in a block may include several lines of an input data file. The numeric data within a block are taken to apply to the keyword at the start of the block. Character data may be modifiers (i.e., user options), comments, or even inconsequential text that enhances the user readability of the input. A block ends when the next keyword, or an end-of-file, is encountered.

Within the input file, three record types are recognized: keyword, comment, and continuation. These record types are described in Section 4.1. Within each record of the input file are six distinct data element types, which are described in Section 4.2.

#### 4.1 RECORD TYPES

Three types of records are described in detail in the following sections. Each 80-column card image read from the data file is considered to correspond to one of these record types. Only the first 80 columns of a record are processed; any further characters are truncated.

##### 4.1.1 The Keyword Record

**Function:** The keyword record indicates the purpose of the numeric and character data on the keyword record and any subsequent continuation records.

**Structure:** The keyword has the following attributes.

- A keyword record must begin with a keyword in column 1.
- Only one keyword per record is allowed.
- The keyword may be followed by modifiers and numerical fields.
- The keyword, the modifiers, and the numerical fields must be separated from one another by separator fields.
- Any character or numeric data in a keyword record after the first occurrence of a terminator is ignored.

#### 4.1.2 Continuation Records

**Function:** A continuation record continues the input of numeric and character data started by the preceding keyword record.

**Structure:** The following rules are to be followed.

- The continuation record must begin with a separator as the first character of the record.
- A continuation record must always occur after a keyword record for that group.
- A continuation record must consist only of a combination of modifiers and numerical fields isolated from each other by separators.
- Any character or numerical data in a continuation record after the first occurrence of a terminator is ignored.
- Any number of continuation records may follow a keyword record.

#### 4.1.3 Comment Records

**Function:** The comment record enhances the clarity and readability of the input.

**Structure:** The comment record has the following attributes.

- A comment record must begin with a slash (/), asterisk (\*), dollar sign (\$), or exclamation point (!) in the first column of the record. Any combination of characters can follow the first character.
- A comment record is not processed. No numerical or character data are extracted.

- A comment record cannot be extended by a continuation record.
- A comment record can be inserted anywhere in the input.

## 4.2 ELEMENTS OF THE INPUT RECORD

One or more of the following six basic elements form an input record: keyword, modifier, numeric field, separator field, terminator field, and comment field. These elements are described in the following subsections.

### 4.2.1 The Keyword

**Function:** The keyword identifies an input group.

**Structure:** The keyword has the following attributes.

- A keyword must begin in the first column of a record.
- The keyword must start with one of the characters 'a' through 'z' or 'A' through 'Z'.
- The keyword may consist of any character except separator or terminator characters. The concept of a keyword is similar to that of a variable name in FORTRAN.
- The keyword may be from one to eight characters in length.
- The keyword is terminated with the first occurrence of a valid separator or terminator character.
- The keyword may be entered in upper- or lowercase.

**Examples:**

ABCD, A123, and B&C are all valid examples of a keyword. The keyword specifications ABCDEFGH, ABCDefgh, and AbCdEfGhXXX (where X can be any character) are all equivalent because only the first eight characters are significant and the input is case-insensitive.

'AB, \$A, =junk, and .HELP, are examples of invalid keyword specifications. Each starts with a separator or comment character.

Note the specification of ABC', ABC\$, and ABC( as keywords, although valid, are all equivalent to ABC because the last character of each example is either a separator or a terminator.

### 4.2.2 The Modifier

Any character information in an input record following a keyword, except that embedded in a numeric or comment field, is treated as a modifier.

**Function:** The modifier contains character data that help to interpret the other data in the record. Many program options are implemented through the use of modifiers.

**Structure:** The modifier has the following attributes.

- A modifier in any input group, if present, must follow the keyword.
- The modifier is identical to the keyword in structure, except that it cannot start in column 1 of the record.
- The modifier must be separated from the keyword, other modifiers, or numeric data by a separator or terminator field.

**Examples:**

The structure of the modifier is identical to that of a keyword, except that the modifier cannot start in the first column of a record. Examples of keywords were given in Section 4.2.1.

#### 4.2.3 The Numeric Field

Any numeric characters in a keyword or continuation record, except those embedded in a keyword, modifier, or comment field, are treated as numeric data.

**Function:** A numeric field contains numeric data for input variables.

**Structure:** The numeric field has the following attributes.

- A numeric field is a continuous string of characters that must begin with the numeric character set. In this context, the numeric character set consists of the numerals (0-9), the decimal point (.), and the plus (+) and minus (-) operators.
- The numeric field cannot contain embedded blanks or separators.
- A numeric field must consist only of the numeric character set defined above, the asterisk (\*), and the exponent in lower- (e) or uppercase (E). It must not contain any other characters.
- The plus (+) or minus (-) sign, if present, must immediately precede the numerical value without any intervening blank or other characters; or, if used in an exponent, it must follow immediately after the exponent character e or E.
- The asterisk (\*) or the exponent (E or e), if present, must be embedded; the numeric field must not begin or end with one of these characters.
- A numeric field must be separated from the keyword, modifiers, and other numeric fields by a valid separator or terminator field.

- A numeric field can be located anywhere on a keyword or continuation record, except in column 1.
- The numeric values can be specified in any of the following formats:
  - Integer (e.g., 999)
  - Real (e.g., 999.0 or -.99)
  - Exponent (e.g., 9.99E+02 or 9.99E-1).
- Successive, repetitive, or identical numeric values may be specified using the asterisk (\*) option. Thus, (30., 30., 30.) may be represented as (3\*30. or 3\*3.E+1); embedded separators or nonnumeric characters must not appear in such specifications.

#### Examples:

The input character strings 1, 0.1234, .567, +123, -1.005, 1.2e+0, 1.35E0, and 3\*1.2 are all valid examples of a numeric field. The input specifications of 123, 123., 1.23e02, +0.123E+3, 1\*123, and 1\*1.23E+02 are all equivalent.

The strings 1AB, 11X11, and 1+2 are examples of invalid numerical fields. The first two fields have embedded nonnumeric values, and the last field has a legal numeric character (+) embedded within the field.

A specification of 1.2)2 or 1.2=2, although valid, will be equivalent to specifying the two separate numeric fields of 1.2 and 2 because of the embedded separator character. A specification of 1.2\$2 is equivalent to 1.2 because of the presence of the terminator character (\$).

#### 4.2.4 The Separator Field

**Function:** A separator field separates the keyword, the modifiers, and the numeric fields of an input record.

**Structure:** Any continuous string of characters in an input record that consists of only characters from the separator character set is treated as a separator field. The valid separator characters are the space ( ), the comma (,), the equal sign (=), the colon (:), the semicolon (;), the apostrophe ('), the left paren '(', and the right paren ')' characters.

Any character with a storage code (FORTRAN ICHAR function) less than 10 will also be treated as a separator character. This convention allows tab characters to be treated as separators.

#### Examples:

The sequence of characters, ';:: ))', '====', '=', and ( ;) are all valid separator fields. However, the characters (a) and (l) are not valid separator fields. In the first case, the character 'a' will be processed as a modifier; in the second case, the character 'l' will be processed as a numeric field.

#### 4.2.5 The Terminator Field

**Function:** A terminator ends input from an individual keyword or continuation record; however, subsequent continuation cards are processed normally. It also provides a vehicle for the user to insert comments in those records.

**Structure:** The terminator field has the following attributes.

- The dollar sign (\$) and exclamation point (!) are the only valid terminators.
- The terminator ends the input for the keyword or continuation record in which it occurs; input associated with that particular keyword may continue in a continuation record that follows.
- The terminator may appear anywhere in a record.
- Any characters following the terminator in that input record are not processed; rather, they are treated as comments.

#### Examples:

The character sequences XYZ !Comments now, \$ any comments here, and 123.456\$789 are all examples of sequences with embedded terminators. In the first sequence, XYZ will be treated as valid character data (either keyword or modifier, depending on its starting location in the input record), whereas the characters following the ! will be ignored. In the second example, the total sequence will be ignored. In the third example, 789 will be ignored, whereas 123.456 will be treated as numeric data.

#### 4.2.6 The Comment Field

**Function:** A comment field provides the vehicle for the user to insert comments into the input stream to enhance the clarity and readability of the input. The comment field may be used in the middle of a keyword and continuation record sequence.

**Structure:** The comment field has the following attributes.

- A comment field may be in the form of a trailing comment or a comment record.
- A trailing comment field may occur in a keyword or continuation record. It must begin with a terminator character; any combination of characters can follow the terminator character. The comment field is terminated at the end of the 80th character of that record.
- A comment field in a comment record may consist of any combination of characters. The comment record must begin with a slash (/), asterisk (\*), dollar sign (\$), or exclamation point (!) in the first column of the record.

**Examples:**

In the input record ARRAY = 1., 2., 3., 4. \$Example 1, the character string \$Example 1 is an example of a trailing comment in a keyword record. Input processing stops with the \$ character; all characters including and following the \$ are ignored. The same result would be achieved if \$ were replaced with !.

The following are examples of comment records:

```
/ARRAY = 1., 2., 3., 4. $Example 1
*ARRAY = 1., 2., 3., 4. !Example 1
!ARRAY = 1., 2., 3., 4. $Example 1
$ARRAY = 1., 2., 3., 4. !Example 1
```

All of these strings of characters will be treated as comment records.

**5.0 DETAILED DESCRIPTION OF KEYWORD COMMANDS**

The method of providing input control data to PORPST is based on reading free-format keywords. The details of keyword structure were described in Section 4.0. Each input command starts with a keyword that identifies the nature of the data to follow. The keyword is followed by alphanumeric data. The following is the notational convention for the input commands to PORPST.

**BOLD** The keywords for PORPST are shown in uppercase characters in boldface type. The string of keyword characters may be specified by the user in upper- or lowercase. Boldface is used here only for notational purposes; it must not be used as operator input.

**CAPS** Uppercase characters in standard typeface are modifiers of the PORPST keywords that are significant for machine interpretation of user input. The string of characters shown may be specified by the user in either upper- or lowercase.

**char** Lowercase characters denote information on keyword commands that is not significant for machine interpretation of user input, but improves the clarity or readability of the input. The string of characters shown may be specified by the user in either upper- or lowercase. In addition, they may be omitted or replaced with other character strings by the operator.

| The vertical bar indicates a choice. For items enclosed in braces or square brackets, only one of the items separated by the bar may be specified. For items enclosed in parentheses, at least one of the items is required and more than one of the items can be specified.

{ } Braces indicate that the enclosed item (or one of the enclosed items separated from the other enclosed items by vertical bars) is required and must be specified.

- [ ] Square brackets indicate that the enclosed item or selection of one of the enclosed items (if several items are listed and separated by vertical bars) is optional.
- ( ) Parentheses indicate that at least one of the enclosed items is required and more than one of the enclosed items can be specified.
- .... Ellipses indicate that other, similar items may follow as shown.
- Nn The n-th numeric value that is associated with an input command.

The PORPST code is controlled completely through the use of keywords. The keywords for PORPST and their functions are summarized in Table 3. Detailed descriptions of the keywords are given in the remainder of Section 5.0.

## 5.1 GENERAL INSTRUCTIONS FOR COMBINING KEYWORDS

This section provides general guidelines for defining a problem for PORPST.

Keyword control information is carried across problem definitions. This allows a single analysis to be set up in the first problem definition and used on a number of data files. If this effect is not desired, the first keyword card of the following problem definition should be **CLEAR ALL**. In a single problem statement, only the **PLOT**, **TITLE**, **XLABEL** **YLABEL**, and **FILE** keywords can be entered more than once. All other keywords will implement only the last entry.

Correlation computations use two (variables) data input files. Distribution, mean, and variance computations use a single (variable) data input file. Only one of the four computational operations (correlation, distribution, mean, or variance) can be done in a single problem definition. The mean value computation will allow a single value to be averaged, thus allowing the use and output of the original data (this is the only method available to view the original data from PORPST).

A problem definition that prints only file summaries is valid. Summary information can be printed from one or two data files in a single problem definition. By putting several problem definitions in the same keyword control file, any number of files can be summarized and analyzed. The **END** keyword card is used to signal the end of a problem definition.

The user can indicate data subsets in various ways. The **WINDOW** keyword can be used to identify a subset of locations to process. The **REALIZATIONS** keyword can be used to identify a subset of realizations to process. The **TIMES** keyword can be used to specify the time slice of the selected realization(s) to process (by default, the last time slice is used). Only one time slice of the selected realization(s) is processed per problem definition.

Table 3. Keywords and Their Functions for the PORPST Code.

Keyword	Input function
<b>CLEAR</b>	Resets keyword control information to default values.
<b>CORRELAT</b>	Performs correlation calculations between two variables.
<b>DISTRIBU</b>	Performs the calculation of a statistical distribution for a single variable.
<b>END</b>	Signals the end of a single problem definition.
<b>ENDINPUT</b>	Signals the end of the input control file.
<b>FILE</b>	Selects and controls input and output files.
<b>PLOT</b>	Generates basic line plots.
<b>REALIZAT</b>	Specifies which realizations of input data are to be used in computations.
<b>SCREEN</b>	Displays output on the terminal.
<b>STOP</b>	Terminates execution immediately.
<b>SUMMARY</b>	Prints summary information for an input data file.
<b>TABLES</b>	Generates a table of computed values.
<b>TIMES</b>	Specifies which time slice of each selected realization is to be used in computations.
<b>TITLE</b>	Specifies the title for a line plot.
<b>TRANSFOR</b>	Performs the specified transformation on the input data values.
<b>USER</b>	Specifies a name for the user.
<b>VARIANCE</b>	Performs variance calculations for a single variable.
<b>XLABEL</b>	Specifies the x-axis label for a line plot.
<b>YLABEL</b>	Specifies the y-axis label for a line plot.

For certain output, the user can further subset the data by using the **SLICE** modifier on the **PLOT** and **TABLES** keywords. The values given to the **SLICE** modifier must be a subset of the computational window or the output will not be performed.

Because of some restrictions in the set of software modules that implement the keyword input, special care is necessary when multiple modifiers that require numeric input are used (see the **PLOT** and **TABLES** keyword descriptions). Special ordering is also required of the **PLOT**, **TITLE**, **XLABEL**, and **YLABEL** keywords in the keyword control file (see the corresponding keyword descriptions).

Because of restrictions in the **DISSPLA** software, all of the plots in a single run of the **PORPST** code must be directed to the same output device. The device may be either the terminal screen or a plot file.

## 5.2 CLEAR COMMAND

### 5.2.1 Purpose

The **CLEAR** keyword is used to reset control keyword information to default values.

### 5.2.2 Syntax

```
CLEAR { ALL | EXFILE | (CORRELAT | DISTRIBU | FILE | MEAN | PLOT | REALIZAT |
        SCREEN | SUMMARY | TABLES | TIMES | TRANSFOR | VARIANCE | WINDOW) }
```

If the **ALL** modifier or the **EXFILE** modifier is used, no other modifiers that exist on the same input card will be recognized. If the **ALL** and **EXFILE** modifiers are not used, one or more of the remaining modifiers must be entered on the input card. An informational message is printed if no modifiers are found on the **CLEAR** keyword card.

### 5.2.3 Examples

The first example below completely erases a problem definition; this card can be used as the first input card in a new problem definition to ensure that all control information has been reset. The second example resets control information for all keywords except the **FILE** keyword. The third example resets the time step and the grid used in computations to default values and disables all output to the terminal screen.

```
CLEAR ALL
CLEAR EXFILE
CLEAR TIMES WINDOW SCREEN
```

## 5.3 CORRELAT COMMAND

### 5.3.1 Purpose

The **CORRELAT** keyword allows the user to compute correlations between two variables. Correlations can be computed only at grid locations using realizations as replicates. A single time slice of the grid is used from each realization. Two data files (one for each variable) are used concurrently for correlations. This requires that the data files match up in coordinate system

Table 4. Modifiers and Their Descriptions for the CLEAR Keyword.

Modifier	Remarks and explanations
ALL	Resets control information for all keywords, produces the same result as using CLEAR with all the other modifiers.
CORRELAT	Turns off the computation of correlations and resets the CORRELAT modifiers.
DISTRIBU	Turns off the computation of distributions and resets the DISTRIBU modifiers.
EXFILE	Resets the control information for all keywords except the FILE keyword.
FILE	Closes all files and resets the FILE modifiers.
MEAN	Turns off the computation of mean values and resets the MEAN modifiers.
PLOT	Turns off the generation of line plots and resets the PLOT modifiers.
REALIZAT	Resets the control information so all realizations available will be used in computations.
SCREEN	Turns off all output to the screen.
SUMMARY	Turns off the generation of summary data for both input data files.
TABLES	Turns off the generation of tables and resets the TABLES modifiers.
TIMES	Resets the control information so the last time step of each realization will be used for computations.
TRANSFOR	Turns off any transformations being performed on the input data.
VARIANCE	Turns off the computation of variances and reset all variance modifiers.
WINDOW	Resets the size of the grid used in computations to the entire grid.

type, grid size, and in the realizations and time slice selected for computation. If the files fail to match, an appropriate error message is printed and the problem definition is skipped. The default (no keyword card present) is to not compute correlations.

### 5.3.2 Syntax

#### CORRELAT [OUTPUT]

The OUTPUT modifier is optional; if used it writes the grid of computed correlations to the contour file. The contour file is an ASCII file in a format acceptable for use by the CONTOUR3 contouring program. A different contour file should be used for each problem definition that creates contour output (use the FILE keyword to open a new file). The default is to not output the computed correlations to the contour file.

### 5.3.3 Example

In this example, correlations will be computed and the resulting grid of correlations will be output to the contour file.

#### CORRELATION OUTPUT

## 5.4 DISTRIBU COMMAND

### 5.4.1 Purpose

The DISTRIBU keyword is used to compute a statistical distribution for a sample from a single variable (only a single data input file is used). Several modifiers exist to allow the user flexibility in specifying the type of statistical distribution to be created and how samples from the variable are to be made. The default value (no keyword card present) is to not compute a distribution, and there are no default values for the modifiers.

### 5.4.2 Syntax

```
DISTRIBU {LOCATION | SINGLE} {DENSITY {HISTOGRA N1 N2 | KERNEL} |
          CUMULATI {EMPIRICA | KERNEL} }
```

Only one of the two modifiers LOCATION and SINGLE can be used. These modifiers specify how the random sample is to be taken from the variable. Only one of the two modifiers DENSITY or CUMULATI can be used. These modifiers specify the form of the statistical distribution. If the DENSITY modifier is used, then either the HISTOGRA or KERNEL modifier must be chosen. If the CUMULATI modifier is chosen, either the EMPIRICA or KERNEL modifier must be chosen. If a specification for a distribution computation is incomplete, an informational message is written to the report file and a distribution will not be computed. If two mutually exclusive modifiers are both entered, the problem will not error terminate and will not produce a message. Distribution computations will be performed with one of the complete definitions. The type of distribution created can be found by reviewing the report file.

Table 5. Modifiers and Their Descriptions for the DISTRIBU Keyword.

Modifier	Numeric field(s)	Remarks and explanations
SINGLE	--	Computes the distribution by using as the random sample data from each realization at the same location. The <b>WINDOW</b> keyword card must be used to identify the single location where the distribution is to be calculated. If a single location is not specified, an error message will be printed and the problem definition skipped.
LOCATION	--	Computes the distribution for a single realization by using data at all locations in the data window as the random sample. The <b>REALIZAT</b> keyword card must be used to identify the single realization used for computation of the distribution. If a single realization is not specified, an error message will be printed and the problem definition skipped.
DENSITY	--	Represents the distribution in the form of a statistical density distribution.
CUMULATI	--	Represents the distribution in the form of a cumulative distribution.
HISTOGRA	N1, N2	Modifier for the DENSITY modifier. This modifier selects a histogram density estimate. The user must supply two numerical values. These values are N1, the lower boundary of the first bin for the histogram, and N2, the bin width. These values are specified in transformed units if a transformation is chosen (see <b>TRANSFOR</b> keyword description).
EMPIRICA	--	Modifier for the CUMULATI modifier. This modifier selects the empirical cumulative distribution function technique.
KERNEL	--	Modifier for the DENSITY and CUMULATI modifiers. This modifier selects a kernel density estimation technique.

### 5.4.3 Examples

The first example computes the sample-based density using the kernel density estimation technique. The random sample is collected by using data from each realization at the same single location. The second example computes the distribution using the empirical CDF technique. The random sample is collected by using data from a single realization at all locations in the data window.

```
DISTRIBUTION SINGLE DENSITY KERNEL
DISTRIBUTION LOCATION CUMULATIVE EMPIRICAL
```

## 5.5 END COMMAND

### 5.5.1 Purpose

The **END** keyword is used as the last keyword card in a problem definition. It informs the program that keyword input for the current problem definition has been completed. Additional problem definitions can follow the **END** keyword. All problem definitions require an **END** keyword card for normal completion (even the last problem definition).

### 5.5.2 Syntax

**END**

There are no modifiers or numerical values for this keyword.

### 5.5.3 Examples

This example signals the program that the user has completed a problem definition, and the program is to now perform the specified computations.

**END**

## 5.6 ENDINPUT COMMAND

### 5.6.1 Purpose

The **ENDINPUT** keyword is used as the last entry in the keyword input file. This keyword tells the program that all problem definitions have been completed and computed. The program will terminate normally if **ENDINPUT** is not used.

### 5.6.2 Syntax

**ENDINPUT**

There are no modifiers or numerical values for this keyword.

### 5.6.3 Examples

The following example signals the program that the user is finished defining problems and wants to exit the program.

**ENDINPUT**

## 5.7 FILE COMMAND

### 5.7.1 Purpose

The FILE keyword allows the user to identify and open or close the files that will be used for a problem definition.

### 5.7.2 Syntax

```
FILE {DATA {N1} [FORMATTE | UNFORMAT] | REPORT | CONTOUR}
    [OPEN | CLOSE] ["filename"]
```

Only one of the three modifiers DATA, REPORT, and CONTOUR can be used in a single FILE keyword card. There is no default value. The DATA modifier requires the use of the numeric field N1 to refer to the desired data file. The FORMATTED and UNFORMATted modifiers are optional modifiers for the DATA modifier with FORMATTED being the default. The OPEN and CLOSE modifiers are optional, with OPEN being the default operation. The file name is required for OPEN operations. There are no default file names. Multiple FILE keyword cards can be used in a single problem definition.

At least one data file is required in each problem definition. Two data files are required only for correlations. A report file must always be opened in the first problem definition and can be changed in subsequent problem definitions. A contour file must be opened in the first problem definition performing output for the CONTOUR3 contour program (a new file must be used for each problem definition performing contour output). A separate file card is required for each file action desired. The name of the DISSPLA plot file used for line plots cannot be controlled by the user; a default file name will be chosen by the DISSPLA program (see Section 3.0).

### 5.7.3 Examples

The first example closes the first data file. The second example opens the unformatted file PORMC.007 as the second data file. The third example opens the file TEST.RPT as the report file.

```
FILE DATA 1 CLOSE
FILE OPEN DATA 2 UNFORMATted "PORMC.007"
FILE OPEN REPORT "TEST.RPT"
```

## 5.8 MEAN COMMAND

### 5.8.1 Purpose

The MEAN keyword allows the user to compute mean values of a single variable (one input data file is required). The default (no keyword card present) is to not perform mean value calculations.

Table 6. Modifiers, Their Descriptions, and Numerical Values for the FILE Keyword.

Modifier	Numeric field(s)	Remarks and explanations
CLOSE	--	Specifies that the desired action is to close the file.
CONTOUR	--	Identifies the CONTOUR file for use in the specified action. The contour file contains the window of computed values and the proper header information for use in the CONTOUR3 contouring program. Contour files are formatted files.
DATA	N1	Identifies a data file for use in the specified action (open or close). The data file is expected to contain output from the program PORMC for a single variable. Because two input data files (one for each variable) are allowed, the DATA modifier requires a numeric field that contains either a 1 or a 2 to allow reference to the two possible data files. There is no ascending restriction on the use of the numeric values 1 and 2 (the value 1 does not need to be used if only one file is being used).
FORMATTE	--	Specifies that the input data file is formatted (ASCII). This modifier is available only with the DATA modifier and is the default-format type.
OPEN	--	Specifies that the desired action is to open the file. This is the default action. If a file of the same specification (DATA 1, DATA 2, REPORT, or CONTOUR) is open already, this file is closed and the new file is opened.
REPORT	--	Identifies the REPORT file for use in the specified action. The report file contains the program identification, problem definition, data file summaries, error and informational messages, and tables. Report files are formatted files.
UNFORMAT	--	Specifies that the input data file is unformatted (binary). This modifier is available only with the DATA modifier.
filename	--	The file name is required to open a file. The file name is optional when closing a file. If used, the file name is limited to 64 characters, which must appear on the keyword card (not a continuation card) and must be enclosed by <u>double</u> quotation marks.

### 5.8.2 Syntax

**MEAN** {LOCATION | SINGLE [OUTPUT]}

Either the LOCATION or the SINGLE modifier is required for specifying how mean values are to be computed (the data are not processed in a default manner). The OUTPUT modifier is optional, but is only recognized when used with the SINGLE modifier.

Table 7. Modifiers and Their Descriptions for the **MEAN** Keyword.

Modifier	Remarks and explanations
LOCATION	Computes mean values for realizations by using data at the specified locations as replicates. A single time slice of the grid is used from each realization. This will produce a single mean value for each realization.
OUTPUT	Outputs the grid of computed mean values to the contour file. This modifier is appropriate only if the SINGLE modifier is used. The contour file is an ASCII file in a format acceptable for use by the CONTOUR3 contouring program. A different contour file must be used for each problem definition that creates contour output (use the FILE keyword to open a new file). The default is to not output mean values to the contour file.
SINGLE	Computes mean values at the selected locations by using the realizations as replicates. A single time slice of the grid is used from each realization. This will produce a grid the size of the current window consisting of mean values.

### 5.8.3 Examples

The first example card requests mean values to be computed at grid locations using realizations as replicates. The resulting mean value grid is to be output to the contour file. The second example requests mean values to be computed for the realizations using the grid locations as replicates.

```
MEAN SINGLE OUTPUT
MEAN LOCATION
```

## 5.9 PLOT COMMAND

### 5.9.1 Purpose

The PLOT keyword is used to control the plotting of results from the type of computation performed (**CORRELATION**, **DISTRIBUTION**, **MEAN**, or **VARIANCE**) in the current problem definition. Only line plots are created, and the type of

computation performed determines the form of the line plot generated by this command. The default (no keyword card present) is to not create line plots.

Line plots can take one of three forms: (1) a statistical distribution plot, (2) a line plot between the values at a node and the node distances along the axis of a one-dimensional data set, or (3) a plot between the value and the realization number. The first and second forms are valid for **CORRELATION** computations and **MEAN** and **VARIANCE** computations with the **SINGLE** modifier; the third form is valid for **MEAN** and **VARIANCE** computations with the modifier **LOCATIONS**.

More than one plot can be made in a single problem statement. The **TITLE**, **XLABEL**, and **YLABEL** keywords are necessary to provide informative titles and labels on line plots. If used, the corresponding title and labels should immediately follow the **PLOT** keyword.

The code structure governing inputs keeps options implemented until they are reset when multiple problem definitions are used in the same keyword control file. If different plots are desired for different problems, each problem statement should start with a **CLEAR PLOT** keyword.

### 5.9.2 Syntax

```
PLOT [ABSCISSA N1 N2] [ORDINATE N3 N4] [SLICE N5 N6 N7 N8 N9 N10]
```

The **ABSCISSA** and **ORDINATE** modifiers are optional for each type of line plot. The **SLICE** modifier is required for the generation of a line plot between node values and node distances if the computational window is multidimensional. There are no default values for the numeric fields. If more than one of the modifiers **ABSCISSA**, **ORDINATE**, and **SLICE** are used, the associated numerical entries are entered assuming the modifiers were entered in alphabetical order. In other words, if **ABSCISSA** and **SLICE** are used, the first two numeric values will be used for the definition of **ABSCISSA** and the next six for the definition of **SLICE**, regardless of the relative order of the **ABSCISSA** and **SLICE** modifiers.

### 5.9.3 Examples

The first example produces a plot that selects nodes 5 through 20 in the Z dimension for the plane (X=1,Y=2) and then limits the x-axis of the plotting window to values ranging between -10.0 and -9.5. The second example produces a default plot for the currently computed values. The third example produces a line plot that limits the plotting window on both the x and y axis.

```
PLOT ABSCISSA -10.0 -9.5 SLICE (1,2,5) TO (1,2,20)
PLOT
PLOT ABSCISSA 2.0 6.0 ORDINATE 0.0 1.0
```

Table 8. Modifiers, Their Descriptions, and Numeric Fields for the PLOT Keyword.

Modifier	Numeric fields	Remarks and explanations
ABSCISSA	N1, N2	Provides minimum and maximum limits on the abscissa (horizontal) of the plot. These values must be expressed in the units of the (transformed) data.
ORDINATE	N3, N4	Provides minimum and maximum limits on the ordinate (vertical) of the plot. These values must be expressed in the units of the (transformed) data.
SLICE	N5...N10	Selects the one-dimensional slice of data to use for the line plot. SLICE defines the data slice using a range of indices on the three dimensions. The first three values are the minimum node numbers for the x, y, and z axes, respectively. The second three values are the maximum node values for the x, y, and z axes, respectively. Two of the three dimensions must be singular, i.e., SLICE (1,1,1) TO (1,1,10) to define a one-dimensional data set in the Z dimension covering the nodes 1 to 10. This modifier is required (unless the computational window is one-dimensional) when line plots of CORRELATIONS, MEANS (if the SINGLE modifier was used), or VARIANCES (if the SINGLE modifier was used) are being performed. The SLICE modifier is ignored when plots of DISTRIBUTIONS or plots of MEANS and VARIANCES (which were computed with the LOCATIONS modifier) are being performed. If the indices used in the SLICE modifier exceed the computational window, an informational message is written and the PLOT keyword is ignored.

## 5.10 REALIZAT COMMAND

### 5.10.1 Purpose

The **REALIZAT** keyword is used to specify the subset of realizations to be processed. The default (no keyword card present) is to process all realizations.

### 5.10.2 Syntax

**REALIZAT** [ALL | RANGE N1 N2 N3]

No modifiers are required. Using **REALIZAT** with no modifier has the same effect as using the ALL modifier. The RANGE modifier can be used to specify a subset of the realizations to be processed.

Table 9. Modifiers, Their Descriptions, Numerical Fields, and Default Values for the **REALIZAT** Keyword.

Modifier	Numeric field	Default value	Remarks and explanations
ALL	--	--	Specifies that all realizations will be processed.
RANGE	N1, N2, N3	--	Specifies a range of realization numbers to be processed. N1 and N2 specify the first and last realization numbers in the range, while N3 specifies the step between processed realizations.

### 5.10.3 Examples

The first example allows all realizations for a variable to be processed. The second example selects realizations 1, 3, and 5 for processing.

```
REALIZATION ALL
REALIZATION RANGE 1,5,2
```

## 5.11 SCREEN COMMAND

### 5.11.1 Purpose

The **SCREEN** keyword allows results to be displayed on the terminal screen. The default (no keyword card present) is for no results to be displayed on the terminal screen.

### 5.11.2 Syntax

**SCREEN** (PLOT | TABLES | SUMMARY )

At least one of the modifiers must be specified. Actions resulting from use of the **SCREEN** keyword are set explicitly each time the keyword is entered and do not carry forward from a previous problem definition.

Table 10. Modifiers and Their Descriptions for the **SCREEN** Keyword.

Modifier	Remarks and explanations
PLOT	Displays line plots on the terminal screen. The terminal must be capable of emulating a Tektronix 4014 graphics terminal. The default is to write the line plots to the DISSPLA output file. Line plots can be written to only the DISSPLA file or the screen.
SUMMARY	Displays the summary information of the input data file(s) on the terminal screen. The default is to write the summary information to the report file but not to the screen. If summary information is written to the screen, it will also continue to be written to the report file.
TABLES	Displays tables of computed values for the data variable(s) on the terminal screen. The default is to write tables to the report file but not to the screen. If tables are written to the screen, they will also continue to be written to the report file.

### 5.11.3 Examples

The following example allows tables and line plots to be displayed on the terminal screen and ensures that the screen output of file summaries is inactivated.

```
SCREEN TABLES PLOT
```

## 5.12 STOP COMMAND

### 5.12.1 Purpose

This keyword terminates execution immediately during the reading of the keyword control file.

### 5.12.2 Syntax

```
STOP
```

There are no modifiers or numerical values for this keyword.

### 5.12.3 Examples

The following example card signals the program to terminate immediately.

```
STOP
```

### 5.13 SUMMARY COMMAND

#### 5.13.1 Purpose

The **SUMMARY** keyword allows the user to print summary information from input data files 1 and/or 2. The default is for no summary information to be printed. The **FILE** keyword is used to define the data files for use.

#### 5.13.2 Syntax

**SUMMARY** {N1} [N2]

One numeric value is required; the second numeric value is optional. There are no default numeric values.

Table 11. Numeric Modifiers and Their Descriptions for the **SUMMARY** Keyword.

Numeric field	Numeric value	Remarks and explanations
N1	1 or 2	Identifies the input data file. The appropriate index (1 or 2) was assigned with the <b>FILE</b> keyword.
N2	1 or 2	Identifies the second input data file. The appropriate index (1 or 2) was assigned with the <b>FILE</b> keyword.

#### 5.13.3 Examples

The first example activates the printing of summary information to the report file for data file 1. The second example activates the printing of summary information to the report file for data files 1 and 2.

```
SUMMARY 1
SUMMARY 1 2
```

### 5.14 TABLES COMMAND

#### 5.14.1 Purpose

The **TABLES** keyword is used to control the tabling of results from the type of computation performed (**CORRELATION**, **DISTRIBUTION**, **MEAN**, or **VARIANCE**) in the current problem definition. The default (no keyword card present) is to not create tables. Only one **TABLES** keyword card is allowed per problem definition.

5.14.2 Syntax

TABLES [PLANE PP] [SLICE N1 N2 N3 N4 N5 N6] [STEPX N7] [STEPY N8] [STEPZ N9]

No modifiers are required. If a TABLES keyword card is entered with no modifiers, the computed values are tabled in a default fashion that depends on the type of computation. Tables for DISTRIBUTIONS and for MEANS and VARIANCES computed with the LOCATIONS modifier ignore definitions of all modifiers. The modifiers are used only to provide additional control for creating tables of CORRELATIONS, and MEANS and VARIANCES computed with the SINGLE modifier.

Table 12. Modifiers, Their Descriptions, and Numerical and Default Values for the TABLES Keyword. (sheet 1 of 2)

Modifier	Numeric value	Default value	Remarks and explanations
PLANE	--	--	Requires a character field to specify the orientation of the plane displayed in the table, i.e., PLANE XZ to produce tables of the X-Z plane for each node selected on the Y axis. The X axis will lie horizontally and the Z axis will lie vertically using this specification. The character field is restricted to one of the following six values: XY, YX, XZ, ZX, YZ, and ZY. The default plane orientation is assigned based on two rules: (1) an axis with a single node is used as the plane index and (2) the Z axis is not allowed to be the horizontal axis. This allows only the XY, YX, XZ, and YZ orientations as <u>default</u> possibilities.
SLICE	N1...N6	--	Requires six numeric values to select the "slice" (subset) of data to print in the table, i.e., SLICE (1,1,1) TO (1,1,10) for a table covering nodes 1 to 10 in the Z direction. The first three values are the minimum node numbers for the x, y, and z axes respectively. The second three values are the maximum node values for the x, y, and z axes respectively. The SLICE modifier is not limited to specifying a one-dimensional slice as in the PLOT keyword command. The default is to use all locations that have computed values (specified by the WINDOW keyword). This modifier is valid only if CORRELATIONS were computed or if VARIANCES or MEANS were computed with the SINGLE modifier. If the indices used in the SLICE modifier exceed the computational window, an informational message is written and the TABLES keyword is ignored.

Table 12. Modifiers, Their Descriptions, and Numerical and Default Values for the TABLES Keyword. (sheet 2 of 2)

Modifier	Numeric value	Default value	Remarks and explanations
STEPY	N8	1	Specifies the increment on indices in the Y dimension (of the data) when creating tables. Must be positive.
STEPZ	N9	1	Specifies the increment on indices in the Z dimension (of the data) when creating tables. Must be positive.

If the modifiers SLICE, STEPX, STEPY, and STEPZ are used, the associated numerical entries are entered assuming the modifiers were entered in alphabetical order. In other words, if STEPZ and SLICE are used, the first six numeric values will be used for the definition of SLICE and the next numeric value will be used for the definition of STEPZ, regardless of the relative order of the SLICE and STEPZ modifiers.

### 5.14.3 Examples

The first example below creates a table in a default format that depends on the type of computation performed. The second example creates a table in the YZ plane (the X dimension is singular) for the grid defined by the Y axis nodes 1 to 3 and the Z axis nodes 1, 4, 7, and 10. The Y axis will lie horizontally and the Z axis will lie vertically. The third example creates a table for the computational window with Y and X axes flipped.

```
TABLES
TABLES SLICE (1,1,1) TO (1,3,10) STEPZ 3
TABLES PLANE YX
```

## 5.15 TIMES COMMAND

### 5.15.1 Purpose

The TIMES keyword is used to identify which time slice of each realization is to be used in the computations. This keyword has no effect if only a single time value is present in the data file. The default operation (no keyword card present) is to use the last time slice of each realization.

### 5.15.2 Syntax

**TIMES** [FIRST | LAST | SINGLE N1]

No modifiers are required. Using **TIMES** with no modifier has the same effect as using the **LAST** modifier. There are no default numeric values for this keyword.

Table 13. Modifiers and Their Descriptions for the **TIMES** Keyword.

Modifier	Numeric value	Remarks and explanations
FIRST	--	Requires that all time slices for all realizations in the input data file are examined and the earliest time slice that exists is used.
LAST	--	Requires that time slices for all realizations in the input data file be examined and the latest time slice that exists is used. This is the default action.
SINGLE	N1	Requires that the specified time slice is used. N1 is a real value and represents the PORMC output time. NOTE: The <b>SUMMARY</b> keyword card can be used in an earlier problem definition to get a listing of time slices per realization.

### 5.15.3 Examples

The first example below instructs the program to use the earliest time slice found in the data file for each realization processed. The second example specifies that the time slice occurring at .0125 is to be used for each realization processed.

```
TIMES FIRST
TIMES SINGLE .0125
```

## 5.16 TITLE COMMAND

### 5.16.1 Purpose

The **TITLE** keyword allows the user to provide a title for a line plot.

### 5.16.2 Syntax

TITLE {"Title"}

The title is limited to 60 characters, must appear on the keyword card (not a continuation card), and must be enclosed by double quotation marks. If used, the TITLE keyword card must follow the associated PLOT keyword card before another PLOT keyword card is specified. A default dummy title is used for each plot created with no explicitly specified title.

### 5.16.3 Examples

In the following example a title is given for the previously defined line plot.

TITLE "Mean Value By Realization"

## 5.17 TRANSFOR COMMAND

### 5.17.1 Purpose

The TRANSFOR keyword is used to specify the data transformation to perform to the input data. The default (no keyword card present) is to perform no data transformations.

### 5.17.2 Syntax

TRANSFOR [LOG10 | LOGE]

One of the two modifiers LOG10 or LOGE is required to specify the type of transformation to perform. There is no default transformation. No data transformations are done unless a TRANSFORM keyword is entered, and the transformation option stays in effect for all subsequent problem definitions until it is reset using the CLEAR keyword.

Table 14. Modifiers and Their Descriptions for the TRANSFOR Keyword.

Modifier	Remarks and explanations
LOG10	Performs a base 10 logarithm transformation. This transformation requires all input data values to be greater than zero.
LOGE	Performs a base e logarithm transformation. This transformation requires all input data values to be greater than zero.

### 5.17.3 Examples

This example specifies a base 10 logarithm transformation on the input data.

```
TRANSFORM LOG10
```

## 5.18 USER COMMAND

### 5.18.1 Purpose

The **USER** keyword can be used to define a new user name. A default user name of "XXXXXX X. XXXXX" is defined and will be used unless the user enters this card. The entire user name is written to the report file. The user name is also written to the contour file, however, only the first eight characters are used.

### 5.18.2 Syntax

```
USER {"New Name"}
```

The user name can contain up to 16 characters, must appear on the keyword card (not a continuation card), and must be contained between double quote marks.

### 5.18.3 Examples

The following example changes the user name to John Doe.

```
USER "John Doe"
```

## 5.19 VARIANCE COMMAND

### 5.19.1 Purpose

The **VARIANCE** keyword allows the user to compute variances of a single variable (one input data file is required). The default (no keyword card present) is to not perform variance calculations.

### 5.19.2 Syntax

```
VARIANCE {LOCATION | SINGLE [OUTPUT]} [STANDARD]
```

Either the **LOCATION** or the **SINGLE** modifier is required for specifying how variances are to be computed (the data are not processed in a default manner). The **OUTPUT** modifier is optional but is only recognized when used with the **SINGLE** modifier. The **STANDARD** modifier is optional.

Table 15. Modifiers and Their Descriptions for the **VARIANCE** Keyword.

Modifier	Remarks and explanations
LOCATION	Computes variances for realizations by using data at the specified locations as replicates. A single time slice of the grid is used from each realization. This will produce a single variance (standard deviation if the STANDARD modifier is used) for each realization.
OUTPUT	Outputs the grid of computed variances (or standard deviations) to the contour file. This modifier is appropriate only if the SINGLE modifier is used. The contour file is an ASCII file in a format acceptable for use by the CONTOUR3 contouring program. A different contour file should be used for each problem definition that creates contour output (use the FILE keyword to open a new file). The default is to not output variances to the contour file.
SINGLE	Computes variances at the selected locations by using the realizations as replicates. A single time slice of the grid is used from each realization. This will produce a grid the size of the current window consisting of variances (standard deviation if the STANDARD modifier is used).
STANDARD	Specifies that standard deviations should be computed. Variances are computed unless this modifier is used.

### 5.19.3 Examples

The first example card shown below requests variances to be computed at grid locations using realizations as replicates. The resulting variance grid is to be output to the contour file. The second example requests standard deviations to be computed for the realizations using the grid locations as replicates.

```
VARIANCE SINGLE OUTPUT
VARIANCE LOCATION STANDARD
```

## 5.20 WINDOW COMMAND

### 5.20.1 Purpose

The **WINDOW** keyword allows the user to use a portion of the grid, rather than the whole grid, for computations. Output of computed values at grid nodes is limited to the size of the computational window. The default value is to use the entire grid.

5.20.2 Syntax

WINDOW {N1 N2 N3 N4 N5 N6}

All six numeric modifiers are required to specify the minimum and maximum index limits in each dimension (X,Y,Z), yielding a subset of the grid.

Table 16. Numeric Modifiers, Their Descriptions, and Default Values for the WINDOW Keyword.

Numeric field	Numeric value	Default value	Remarks and explanations
N1	Positive	1	Specifies the minimum node to be included in the computational window in the X dimension (R dimension for cylindrical coordinates).
N2	Positive	1	Specifies the minimum node to be included in the computational window in the Y dimension (Theta dimension for cylindrical coordinates).
N3	Positive	1	Specifies the minimum node to be included in the computational window in the Z dimension.
N4	Positive	--	Specifies the maximum node to be included in the computational window in the X dimension (R dimension for cylindrical coordinates).
N5	Positive	--	Specifies the maximum node to be included in the computational window in the Y dimension (Theta dimension for cylindrical coordinates).
N6	Positive	--	Specifies the maximum node to be included in the computational window in the Z dimension.

5.20.3 Examples

The first example below sets the computational window to use grid nodes 1 through 3 in the X dimension, 1 through 4 in the Y dimension, and 1 through 7 in the Z dimension. The second example sets the computational window to the single grid node (1,1,1).

```
WINDOW (1,1,1) TO (3,4,7)
WINDOW (1,1,1) TO (1,1,1)
```

## 5.21 XLABEL COMMAND

### 5.21.1 Purpose

The **XLABEL** keyword allows the user to provide a label for the horizontal (x) axis of a line plot.

### 5.21.2 Syntax

```
XLABEL {"Label"}
```

The label is limited to 64 characters, must appear on the keyword card (not a continuation card), and must be enclosed by double quote marks. If used, the **XLABEL** keyword card must follow the associated **PLOT** keyword card before another **PLOT** keyword card is specified. A default dummy x-axis label is used for each plot created with no explicitly specified x-axis label.

### 5.21.3 Examples

In the following example, an x-axis label is given for the previously defined line plot.

```
XLABEL "Realization Number"
```

## 5.22 YLABEL COMMAND

### 5.22.1 Purpose

The **YLABEL** keyword allows the user to provide a label for the vertical (y) axis of a line plot.

### 5.22.2 Syntax

```
YLABEL {"Label"}
```

The label is limited to 64 characters, must appear on the keyword card (not a continuation card), and must be enclosed by double quotation marks. If used, the **YLABEL** keyword card must follow the associated **PLOT** keyword card before another **PLOT** keyword card is specified. A default dummy y-axis label is used for each plot created with no explicitly specified y-axis label.

### 5.22.3 Examples

In the following example, a y-axis label is given for the previously defined line plot.

```
YLABEL "Mean Value"
```

## 6.0 EXAMPLE RUN

This section contains one example run of the code. The example (Table 17) contains two problem definitions that exercise different features of the code and illustrate the free-form command language.

The first defined problem computes correlations between the variables in the two data files for only the grid nodes specified. The last time slice of all realizations will be used by default. The file EXAMPLE.RPT is opened as the report file. Summary information on the two data files is written to the report file and to the screen. A table is created from the results and written to the report file. Plots with accompanying titles and labels are produced and written to the DISSPLA metafile.

The second defined problem turns off correlation computations, the creation of summary information, and output to the screen. It also closes the first data file and clears the previously defined plots. In problem 2, the report file, the window size, and the creation of tables will be the same as in problem 1. Mean computations will be performed using the first time slice of the odd-numbered realizations from 1 to 15. A single mean value will be computed for each of these realizations. A plot is also defined with title and labels and will be written to the DISSPLA metafile.

Table 17. Input Data File: EXAMPLE1.DAT.

---

```
/ Control File EXAMPLE1.DAT for the PORPST User's Guide
/
USER is "BT DIDIER"
/
FILE OPEN REPORT "EXAMPLE.RPT"
FILE OPEN DATA file 1 FORMATTED "STOCH.3"
FILE OPEN DATA file 2 FORMATTED "STOCH.7"
SUMMARY for files 1 and 2
SCREEN output for SUMMARY
WINDOW for computations is (2,2,10) TO (2,2,75)
CORRELATIONS will be computed
TABLES will be created
PLOT the computational window
TITLE is "Correlations between variables 3 and 7 for Z axis nodes 10 - 75"
XLABEL is "Node distance"
YLABEL is "Correlation"
PLOT limit ABSCISSA from -5 meters to -1.5 meters
TITLE is "Correlations between variables 3 and 7 for Z axis nodes 15 - 50"
XLABEL is "Node distance"
YLABEL is "Correlation"
END of first problem

CLEAR CORRELATIONS SUMMARY SCREEN PLOT
FILE CLOSE DATA 1
TIMES FIRST
MEAN values computed over LOCATIONS
REALIZATIONS RANGE 1 to 15 by 2
PLOT of the mean value by realization
TITLE is "Mean value vs realization"
XLABEL is "Realization number"
YLABEL is "Mean value"
END
ENDINPUT
```

---

## 7.0 REFERENCES

- Beran, R., 1977, "Minimum Hellinger Distance Estimates for Parametric Models," *The Annals of Statistics*, Vol. 5, No. 3, pp. 445-463.
- Devroye, L., 1987, *A Course in Density Estimation*, Birkhauser, Boston, Massachusetts.
- Epanechnikov, V. A., 1969, "Non-Parametric Estimation of a Multivariate Probability Density," *Theory of Probability and its Applications*, Vol. XIV, No. 1, pp. 153-158.
- Eslinger, P. W., and W. A. Woodward, 1990, "Minimum Hellinger Distance Estimation for Normal Models," in the *Australian Journal of Statistics*.
- Mood, A. M., F. A. Graybill, and D. C. Boes, 1974, *Introduction to the Theory of Statistics*, Third Edition, McGraw-Hill Book Co., New York, New York.
- Parzen, E., 1962, "On the Estimation of a Probability Density Function and the Mode," *Annals of Mathematical Statistics*, Vol. 33, pp. 1065-1076.
- Runchal, A. K., and B. Sagar, 1989, *PORFLO-3: A Mathematical Model for Fluid Flow, Heat, and Mass Transport in Variably Saturated Geologic Media, Users Manual, Version 1.0*, WHC-EP-0041, Westinghouse Hanford Company, Richland, Washington.
- Sagar, B., and A. K. Runchal, 1990, *PORFLO-3: A Mathematical Model for Fluid Flow, Heat, and Mass Transport in Variably Saturated Geologic Media*, WHC-EP-0042, Westinghouse Hanford Company, Richland, Washington.
- Serfling, R. J., 1980, *Approximation Theorems of Mathematical Statistics*, John Wiley and Sons, New York, New York.
- Strait, P. T., 1989, *A First Course in Probability and Statistics with Applications*, Second Edition, Harcourt Brace Jonanovich, San Diego, California.
- WHC, 1991, *PORMC: A Model for Monte Carlo Simulation of Fluid Flow, Heat, and Mass Transport in Variably Saturated Geologic Media: Theory and User's Manual*, WHC-EP-0445, Westinghouse Hanford Company, Richland, Washington.

This page intentionally left blank.

**DISTRIBUTION**

Number of copies

OFFSITE

5	<p><u>U.S. Department of Energy</u>          1000 Independence Avenue, S.W.          Forrestal Building          Washington, D.C. 20585</p> <p>L. P. Duffy          C. Frank          S. Lien          J. Lytle          R. P. Whitfield</p>	<p>EM-1          EM-50          EM-54          EM-30          EM-40</p>
1	<p><u>U.S. Department of Energy</u>          12800 Middlebrook Road          Germantown, Maryland 20874</p> <p>K. Hain</p>	<p>EM-54</p>
12	<p><u>U.S. Department of Energy</u>          Office of Scientific and          Technical Information          P.O. Box 62          Oak Ridge, Tennessee 37831</p>	
1	<p><u>Bureau of Economic Geology</u>          University of Texas at Austin          University Station          Box X          Austin, Texas 78713</p> <p>B. Scanlon</p>	
1	<p><u>Colorado School of Mines</u>          Department of Chemistry and Geochemistry          Golden, Colorado</p> <p>D. Langmuir</p>	
3	<p><u>EG&amp;G Idaho, Inc.</u>          P.O. Box 1625          MS 2107          Idaho Falls, Idaho 83415</p> <p>R. G. Baca          R. R. Seitz          J. C. Walton</p>	

DISTRIBUTION (cont)

Number of copies

OFFSITE

2	<p><u>Lawrence Livermore National Laboratory</u>                  University of California                  P.O. Box 808                  Livermore, California 94550</p> <p>K. Pruess                  L. D. Ramspott</p>	
1	<p><u>Los Alamos National Laboratory</u>                  P.O. Box 1663                  Los Alamos, New Mexico 87545</p> <p>B. J. Travis</p>	
1	<p><u>New Mexico State University</u>                  Department of Agronomy                  P.O. Box 30003                  Las Cruces, New Mexico 88003</p> <p>R. Hills</p>	
1	<p><u>Oak Ridge National Laboratory</u>                  P.O. Box Y                  Oak Ridge, Tennessee 37830</p> <p>J. O. Blomeke</p>	
1	<p><u>Office of Civilian Radioactive                  Waste Management</u>                  Forrestal Building                  Washington, D.C. 20585</p> <p>K. H. Kale</p>	RW-20
3	<p><u>Office of Remedial Action                  and Waste Technology</u>                  19901 Germantown Road                  Germantown, Maryland 20585</p> <p>J. A. Coleman                  T. W. McIntosh                  H. F. Walter</p>	<p>EM-35                  EM-343                  EM-343</p>



**DISTRIBUTION (cont)**

Number of copies

OFFSITE

1	<u>U.S. Geological Survey</u> Water Resources Division Low-Level Radioactive Waste Program 12201 Sunrise Valley Drive Reston, Virginia 22092  J. Fischer	
2	<u>U.S. Nuclear Regulatory Commission</u> Division of Engineering Safety Washington, D.C. 20555  S. Coplan T. J. Nicholson	NLS-260 NL-005
2	<u>Westinghouse Idaho Nuclear Company, Inc.</u> P.O. Box 4000 Idaho Falls, Idaho 83401  J. A. Berreth D. A. Knecht	

ONSITE

7	<u>U.S. Department of Energy-Richland</u> <u>Operations Office</u>  J. C. Bartlett R. D. Freeberg M. J. Furman J. M. Hennig S. K. Moy G. W. Rosenwald Public Reading Room	A1-10 A5-19 A5-21 A5-21 A5-21 A4-02 A1-65
25	<u>Pacific Northwest Laboratory</u>  S. Q. Bennett M. P. Bergeron P. W. Eslinger J. W. Falco M. J. Fayer (2) M. D. Freshley P. C. Hays	K6-35 K6-77 K6-96 K6-78 K6-77 K6-77 K6-86

**DISTRIBUTION (cont)**

Number of copies

ONSITE

Pacific Northwest Laboratory (cont)

D. J. Holford	K6-77
J. F. Keller	K1-51
C. T. Kincaid	K6-77
W. E. Nichols	K6-77
K. R. Roberson	K6-77
M. L. Rockhold	K6-77
C. S. Simmons	K6-77
R. L. Skaggs	K6-77
J. L. Smoot	K6-77
J. A. Stottlemire	K6-75
S. K. Wurstner	K6-77
Technical Files (5)	K1-11
Clearance Office	K1-02

24

Westinghouse Hanford Company

L. C. Brown	H4-51
J. W. Cammann	H4-14
L. B. Collard	H4-14
M. P. Connelly	H4-14
J. D. Davis	H4-55
K. R. Fecht	H4-56
R. L. Jackson	H4-56
R. Khaleel	H4-14
N. W. Kline	H0-31
D. Langford	B2-25
A. G. Law	H4-56
T. LeGore	B4-63
R. E. Lerch	B2-35
J. C. Sonnichsen (5)	H4-14
D. D. Wodrich	B3-72
R. D. Wojtasek	L4-92
Central Files	L8-04
Document Clearance Administration	H4-17
Document Processing and Distribution (2)	L8-15

This page intentionally left blank.

**END**

**DATE  
FILMED**

**11 126 191**

