SAND 90-0566 Revised
Unlimited Release
Printed June 1993

Distribution
Category UC-405

Supersedes SAND90-0566 dated April 1990
Last Modified 5/20/98

GREPOS:
A Genesis Database Repositioning Program

Version 1.21, 1998/12/08 (exodus 1)

Version 1.11, 1999/10/18 (exodus 2)

Gregory D. Sjaardema
Engineering and Manufacturing Mechanics
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

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

Intentionally Left Blank

Contents

Intentionally Left Blank

1 Introduction

GREPOS is a mesh utility program that repositions or modifies the configuration of a two-dimensional or three-dimensional mesh. *GREPOS* is one of the mesh generation tools in the Sandia National Laboratories Engineering Analysis Code Access System (SEACAS) [1].

1.1 Modifications Since First Printing

This document is a revised version of SAND90-0566 printed April 1990. Chapter 1 has been completely rewritten to more closely integrate the report with the other SEACAS mesh generation documentation. The following additional commands have been added to *GREPOS* since that printing:

- INCREMENT, ADJUST, RANDOMIZE, SMOOTH, SWAP, and LIMITS

The following commands have additional options:

- LIST: Added the options SSETS, NSETS, INFORMATION, and QA
- DELETE: Added the options QAINFO, NSETS, and SSETS
- CHANGE: Added the options TYPE, NSETS, and SSETS

The new commands and options are discussed in Chapter 2. Chapter 3 has been added to provide usage examples for *GREPOS*. Chapter 4 (formerly Chapter 3) is unchanged from the previous report, and the information formerly in Chapter 4 has been moved to Appendix C which has been updated to reflect the move of the primary computing environment from VMS/CTSS centered to Unix centered.

1.2 Modifications Since Second Printing

The following commands have been added to *GREPOS* since the second printing:

- SNAP (page 18)
- WARP (page 19)
- EXECUTE (page 21)
- MOVE (page 14)
- COMBINE (page 20)
- EQUIVALENCE (page 21)

The following commands have had additional options added:

- ADJUST: Added the option CENTER. (page 14)

This version of the document has not yet been published. Change bars in the left margin indicate modifications since the second printing*.

1.3 Sandia Engineering Analysis Code Access System Mesh Generation Toolbox

GREPOS is typically used with the other SEACAS mesh generation codes *FASTQ* [2], *GEN3D* [3], *GENSHELL* [4], *GJOIN* [5], and *Aprepro* [6]. Figure 1 shows the structure of the SEACAS mesh generation toolbox. The basic premise underlying this toolbox is that complicated geometries can be generated using a set of small specialized codes.

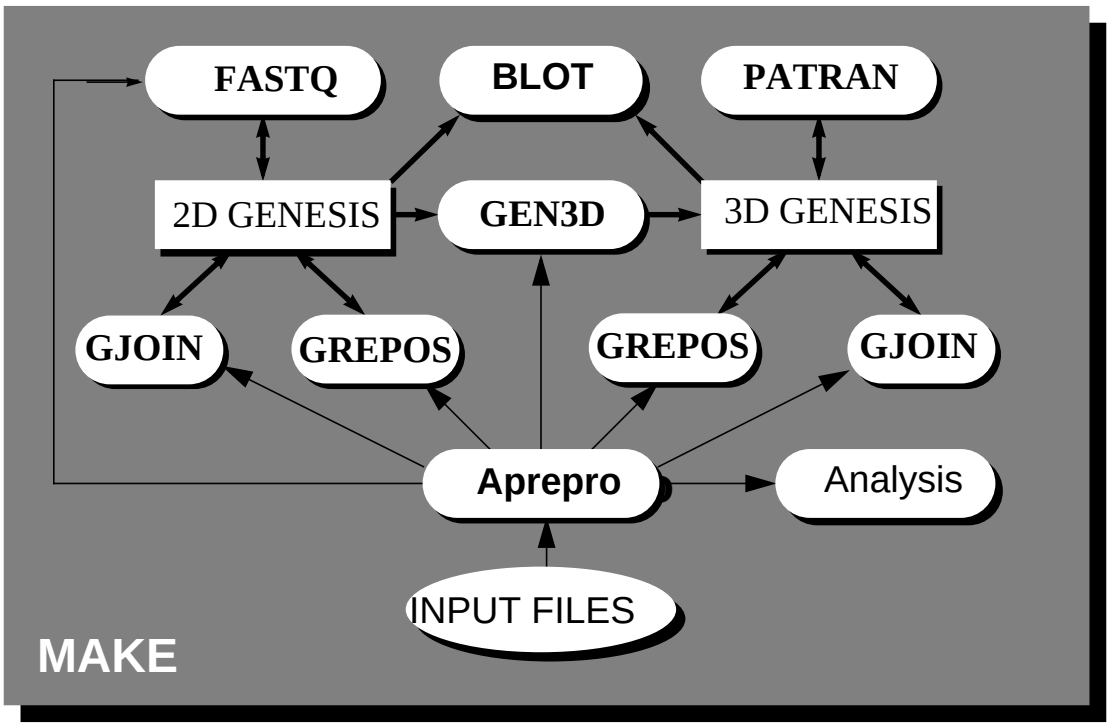

Figure 1. Schematic of SEACAS Mesh Generation Process

Each of these codes has a specialized purpose, a short synopsis of each code is given below, for more information consult the referenced documentation.

GEN3D Transforms a two-dimensional *GENESIS* database into a three-dimensional *GENESIS* database. Several transformations are supported and additional transformations can be easily added.[3]

GREPOS Transforms the geometry of a *GENESIS* database by scaling, mirroring, offsetting, or rotating. It can also modify the database by deleting or renaming material blocks, sideset identifications, or nodeset identifications.[7]

---

* Some of the change bars are due to internal FrameMaker changes and do not actually indicate changes in text. These "false" changes are typically associated with cross-references.

GJOIN	Join together two or more *GENESIS* databases into a single *GENESIS* database.[5]
FASTQ	Interactive two-dimensional finite element mesh generation program. Includes several mesh generation options including paving.[2]
APREPRO	An algebraic preprocessing program which is used to parameterize finite element analyses. Includes a unit conversion system and material database access routines.[6]
GENSHELL	Transforms a two-dimensional *GENESIS* database into a three-dimensional shell *GENESIS* database. Several transformations are supported and additional transformations can be easily added.[4]
NUMBERS	Calculates several properties of an *EXODUS* file, including mass properties, timesteps, condition numbers, cavity volumes, and others.[8]
BLOT	The primary graphical two-dimensional and three-dimensional postprocessing code. It includes deformed mesh plots, contour plots, shaded fringe plots, variable versus variable and time history plots, and distance versus variable plots. [9]

The SEACAS mesh generation toolbox is a simpler approach to three-dimensional mesh generation than the automatic and general-purpose programs that are available from commercial vendors. Many complicated three-dimensional geometries are composed of several primitives that can be defined in terms of transformations of two-dimensional geometries. Each of the primitives can be meshed using *FASTQ* and *GEN3D*, and then joined together using *GJOIN*.

This approach does; however, have some inherent difficulties. The biggest being managing and synchronizing several related files. For example, the meshes for some large problems can require more than one hundred files containing *FASTQ*, *GEN3D*, *GJOIN*, and *GREPOS* input files; temporary *GENESIS* files; and parameter files. Manually building and modifying a mesh this complicated is obviously very difficult and time consuming. This problem has typically been minimized at SNL through the use of the *UNIX** *make*† program and *Aprepro*. *Make* is used to build a set of dependencies between the various pieces of the finite element model. The analyst can then change files as needed and simply type `make mesh` to generate the mesh. If the dependencies have been entered correctly, *make* will rebuild only those portions of the mesh that are affected by the changed file. The synchronization problem (that is, ensuring that all of the separate pieces have compatible dimensions and discretization) is typically solved by creating a few parameter files which

---

* UNIX is a registered trademark of UNIX Systems Laboratories Inc.

† See your UNIX documentation for more information on make. Typically this is done by entering the command `man 1 make`

contain key dimensions and discretization information. *Aprepro* is then used to preprocess the input files and insert the key dimensions and discretization information into the input files.

1.4 Introduction to the GENESIS File Format

The *GENESIS* mesh database file format is the geometry definition portion of the *EXODUS* database file format used in the Engineering Sciences Center at Sandia National Laboratories. All of the mesh generation programs in the Engineering Sciences Center read and write files in the *GENESIS* format, which allows great flexibility in the choice of mesh generation, file translation, and graphical processing.

The *GENESIS* file contains the data to describe a finite element mesh including the location of the nodal points, the connectivity of the nodes that form each element, the material types of each element, and the boundary condition data which are used to specify load application points and nodal constraints. The reader is referred to References [10] and [11] for more information.

1.5 Overview of Capabilities in *GREPOS*

With *GREPOS*, the user can change the orientation and size of a two-dimensional or three-dimensional mesh; change or delete material block, nodeset, and sideset IDs; "explode" the mesh to facilitate viewing of the various parts of the model; or apply a smoothing process to the generated mesh to produce better shaped elements. *GREPOS* also updates the *EXODUS* QA and information records to help track the codes and files used to generate the mesh.

Orientation:
The orientation of the mesh can be changed by revolving, offsetting, reflecting, zeroing, and scaling the nodal coordinates of the original mesh. A coordinate randomization function is available to try to mimic the naturally occurring imperfections that are present in real-life geometries.

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

Mesh "Explosion" for Viewing Multiple Material Meshes:
An exploded view generation capability has been implemented in *GREPOS*. An exploded view is generated by offsetting each material block by a specified distance in each of the two or three coordinate directions. This allows viewing pieces of the mesh that are hidden by other pieces.

Quality Assurance (QA) and Information Records:
A QA record for the *GREPOS* program is added to the input QA record(s), and the file-name of the input file is added to the information records that are written to the output mesh. The information record is prefaced by *GREPOS* to help in identification. These records, which are explained in Reference [10], are useful in tracing the evolution of a mesh during the mesh generation process.

1.6 Organization of Report

The remainder of this report is organized as follows:

- Chapter 2 describes the command input and valid commands,
- Chapter 3 presents a few short examples illustrating *GREPOS* use, and
- Chapter 4 describes the informational and error messages written by *GREPOS*.

Three appendices are contained in this report.

- Appendix A is a code segment defining the *GENESIS* binary database format,
- Appendix B is a summary of the commands in *GREPOS*, and
- Appendix C describes the specifics of *GREPOS* including executing it, obtaining it, compiling it, and quality assurance.

2 Command Input

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

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

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

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

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

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

The notation conventions used in the command descriptions are:

- The command verb is in **bold** type.
- A literal string is in all uppercase SANSERIF type and should be entered as shown (or abbreviated).
- The value of a parameter is represented by the parameter name in *italics*.
- A literal string in square brackets ("[ ]") represents a parameter option which is omitted entirely (including any following comma) if not appropriate. These parameters are distinct from most parameters in that they do not require a comma

* Arithmetic equations, variables, and control structures can be used in the input files if the -aprepro option is used. See the Aprepro manual for more information.

as a place holder to request the default value.

- The default value of a parameter is in angle brackets ("< >"). The initial value of a parameter set by a command is usually the default parameter value. If not, the initial setting is given in the command description.

2.1 Mesh Orientation

ADJUST MINIMUM|MAXIMUM|CENTER *axis*$_1$,*value*$_1$,…

ADJUST calculates the required offset such that the specified minimum, maximum, or bounding box center X, Y, or Z coordinate in the generated mesh is equal to *value* (ignoring any other mesh orientation commands). Only one minimum, maximum, or center per coordinate axis can be specified. The last specification for each coordinate axis is retained if the ADJUST command is used multiple times. The center is defined to be the center of the bounding box of the model.

The OFFSET RESET will cancel the ADJUST command.

EXPLODE <no parameters>

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

MIRROR *axis*$_1$, *axis*$_2$,… <No Default>
MIRROR RESET <no reflections>

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

The MIRROR RESET command resets to no reflection.

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

MOVE *slave_sideset_id* TO *master_sideset_id*
[X|Y|Z|MINUSX|MINUSY|MINUSZ] [TOLER *tolerance*]
MAXDELTA *maxdelta* [GAP *gap]*

MOVE *slave_sideset_id* TO *master_sideset_id* [VECTOR *vx vy vz*] [TOLER *tolerance*] MAXDELTA *maxdelta* [GAP *gap*]

MOVE moves the nodes in all element blocks connected to the sideset specified by *slave_sideset_id* the minimum distance required such that a node in *slave_sideset_id* touches an element face of the sideset specified by *master_sideset_id*. The nodes are moved in the direction specified by the displacement vector. Several options are available for specifying the displacement vector of the slave nodes. The options X, Y, Z, MINUSX, MINUSY, and MINUSZ specify the displacement vector as the respective coordinate direction. An explicit displacement vector can be specified if the VECTOR option is entered.

The TOLER option is used to specify how close to the master element face the slave vector must be to detect a match. If a slave node does not intersect any element face within the specified tolerance, the maximum tolerance required for a match is output.

The MAXDELTA option specifies the maximum distance that a node can be moved to move it to the master surface. This is needed in cases where the master surface folds back on itself and more than one intersection point is possible (e.g. a tire). The maxdelta distance defaults to 1.0e15 if not entered.

The GAP option specifies the distance between the slave sideset nodes and the master_sideset_id element faces. If GAP=0.0, then the closest node on the sideset specified by *slave_sideset_id* will lie directly on an element face of the *master_sideset_id* sideset. If GAP<0.0, then the node will penetrate a master_sideset_id element face, and if GAP>0.0, then the minimum distance between the master and slave sidesets will equal the specified gap distance. The specified gap distance is applied after the search phase of the algorithm, so it doesn't affect the MAXDELTA or displacement vector.

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

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

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

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

OFFSET SPLINE

OFFSET SPLINE is a special form of the OFFSET command in which the offsets in the X and Y coordinates are given in terms of a spline curve as a function of the Z coordinate. OFFSET SPLINE initiates an input mode in which the spline curve is defined. Valid commands in this input mode are:

BOTTOM *xslope yslope* - the slope at the minimum Z coordinate of the curve
BACK *xslope yslope* - the slope at the minimum Z coordinate of the curve
TOP *xslope yslope* - the slope at the minimum Z coordinate of the curve
FRONT *xslope yslope* - the slope at the minimum Z coordinate of the curve
zcord, xcord, ycord - points defining the curve, must be entered from minimum Z to maximum Z.
LIST COMMANDS - list valid commands in spline input mode.
EXIT or END - terminate spline input mode.

This command can only be used with three-dimensional *GENESIS* files.

RANDOMIZE $axis_1$, $magnitude_1$, $axis_2$, $magnitude_2$, …
RANDOMIZE ALL *magnitude* <0.0>
RANDOMIZE RESET <initial condition>
RANDOMIZE BLOCK <no parameters>

RANDOMIZE specifies random offsets to be added to the coordinates. If a REVOLVE command has been issued, the mesh is rotated before it is offset. The specified coordinates are offset by a random amount in the range from -*magnitude* to +*magnitude*.

RANDOMIZE ALL randomly offsets all of the coordinates by the specified magnitude, and RANDOMIZE RESET cancels the offsets.

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

If there are contact surfaces in the mesh description, the nodes on both surfaces will be moved using a different random offset; therefore, if the two surfaces are initially in contact, it is almost certain that they will overlap after they are offset.

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

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

REVOLVE *axis*$_1$, *ndeg*$_1$, *axis*$_2$, *ndeg*$_2$, … <last selection >
REVOLVE RESET <initial condition>

REVOLVE causes the mesh to be rotated. Each (*axis*$_i$, *ndeg*$_i$) parameter pair specifies an axis (X or Y or Z) and the number of degrees to rotate. The rotations are according to right-hand rule. The center of the rotation is specified by the REVCEN command.

The rotations are according to right-hand rule. The center of the rotation is specified by the REVCEN command.

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

For a two-dimensional mesh, *axis*$_i$ must be Z.

SCALE *axis*$_1$, *scale*$_1$, *axis*$_2$, *scale*$_2$, …
SCALE ALL *scale_factor* <1.0 >
SCALE RESET <initial condition>

SCALE causes the specified coordinates of axis *axis*$_i$ to be multiplied by the scaling multiplier *scale*$_i$. The scaling multiplier must be greater than zero; the MIRROR command must be used with the SCALE command if a negative scaling multiplier is required. For example, SCALE X 2.54 MIRROR X will scale the X coordinates by -2.54.

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

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

SCALE ATTRIBUTE *number* **BLOCK** *id scale_factor*
SCALE ATTRIBUTE ALL BLOCK *id scale_factor*
SCALE ATTRIBUTE *number* **BLOCK ALL** *scale_factor*
SCALE ATTRIBUTE ALL BLOCK ALL *scale_factor*
SCALE ATTRIBUTE RESET

SCALE ATTRIBUTE causes the specified attribute of the specified element block to be multiplied by the scaling multiplier *scale_factor*.

SCALE ATTRIBUTE ALL multiplies all of the attributes of the specified element block by the specified multiplier, and SCALE ATTRIBUTE *number* BLOCK ALL scales the specified attribute of all element blocks (with attributes).

SCALE ATTRIBUTE RESET resets all attribute scale factors to 1.0

Attribute scalings are not cumulative.

SCALE BLOCK

SCALE BLOCK initiates an input mode where *GREPOS* prompts for the scale factor to be applied to the nodal coordinates of each material block. Before performing the material block scaling, *GREPOS* checks for connections between the material blocks. All connected material blocks will be assigned the same scaling to avoid distorting the mesh. An informational message will be output to the standard output device for each connected material block.

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

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

SMOOTH *tolerance* <1.0E-6>, *iterations* <100>, *relax* <1.0>

SMOOTH performs a length-weighted Laplacian smoothing using Gauss-Siedel iterations [2] which moves a node along a vector sum of weighted factors from the node to nearest neighbor nodes. The weighting is based on the relative distance to the neighboring nodes. Nodes on material boundaries are moved along the tangent to the boundary, and corner nodes are not moved. The iterations will continue until either the maximum node movement in one iteration is less than *tolerance*, or until the iterations exceed *iterations*.

SNAP *slave_sideset_id* TO *master_sideset_id*
[NORMAL|X|Y|Z|MINUSX|MINUSY|MINUSZ] [TOLER *tolerance*]
MAXDELTA maxdelta
SNAP *slave_sideset_id* TO *master_sideset_id* [VECTOR *vx vy vz*] [TOLER *tolerance*]
MAXDELTA maxdelta
SNAP *slave_sideset_id* TO *master_sideset_id* [CENTER *cx cx cz*] [TOLER *tolerance*]
MAXDELTA maxdelta

SNAP adjusts the positions of the nodes on the sideset specified by *slave_sideset_id* to lie on the element faces of the sideset *master_sideset_id*. Several options are available for specifying the displacement vector of the slave nodes. The default option is NORMAL. In this case, the slave node normal is calculated as the average of the element face normals for all element faces connected to the slave node. The options X, Y, Z, MINUSX, MINUSY, and MINUSZ specify the displacement vector as the respective coordinate direction. An explicit displacement vector can be specified if the VECTOR option is entered. The last option is to specify a radial displacement vector originating from the specified CENTER location.

The TOLER option is used to specify how close to the master element face the slave vector must be to detect a match. If a slave node does not intersect any element face within the specified tolerance, the maximum tolerance required for a match is output.

The MAXDELTA option specifies the maximum distance that a node can be moved to snap it to the master surface. This is needed in cases where the master surface folds back on itself and more than one intersection point is possible (e.g. a tire). The maxdelta distance defaults to 1.0e15 if not entered.

WARP XAXIS|YAXIS|ZAXIS|ORIGIN **RADIUS** *radius* **NORMAL** X|Y|Z

WARP takes an input 3D mesh and warps (or wraps) it around the specified axis. The RADIUS value is the 'reference radius' which means a plane in the original mesh a distance of 'radius' from the specified axis will be mapped into the new space with no stretching or shrinking. Planes farther from the axis will be stretched, closer to the axis will be shrunk.

The NORMAL along with the warping axis specify another plane. The new and old coordinates of the mesh nodes lying on this plane will be equal. The normal also specifies the 'zero angle' or origin of the angle space for the mapping.

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

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

2.2 Modification of Attributes or Material, Nodeset, or Sideset IDs

CHANGE MATERIAL|NODESET|SIDESET|NSETS|SSETS *old_id new_id*
CHANGE TYPE *block_id new_type*

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

CHANGE TYPE changes the element type of the specified material block to the specified type. For example, CHANGE TYPE 1 SHELL changes the element type of material block 1 to SHELL. Note that only the identification of the element type is changed. This command does not change the connectivity or any other descriptor of the element block. Valid identifiers for element types are analysis code dependent; consult the documentation for the target analysis code for more information.

CHANGE ATTRIBUTE *number* **BLOCK** *id value*
CHANGE ATTRIBUTE ALL BLOCK *id value*
CHANGE ATTRIBUTE *number* **BLOCK ALL** *value*
CHANGE ATTRIBUTE ALL BLOCK ALL *value*
CHANGE ATTRIBUTE RESET

CHANGE ATTRIBUTE causes the specified attribute of the specified element block to be set to *value.*

CHANGE ATTRIBUTE ALL sets the value of all of the attributes of the specified element block to the specified value, and CHANGE ATTRIBUTE *number* BLOCK ALL sets the specified attribute of all element blocks (that contain attributes).

CHANGE ATTRIBUTE RESET resets all attributes to their original value.

Attribute changing cannot be combined with attribute scaling of the same attribute.

COMBINE BLOCK *id_final* **WITH BLOCK** *id1, id2, ..., idn*
COMBINE MATERIAL *id_final* **WITH MATERIAL** *id1, id2, ..., idn*
COMBINE NODESET *id_final* **WITH NODESET** *id1, id2, ..., idn*
COMBINE SIDESET *id_final* **WITH SIDESET** *id1, id2, ..., idn*

COMBINE combines the specified entities (element blocks, materials, nodesets, or sidesets) into a single entity of the same type with the specified id. An entity with the specified final id must exist and the other entities are added to this id. The combined ids of the entities combined into the final id will not appear in the output database. There is currently no capability to undo the COMBINE command.

DELETE MATERIAL|NODESET|SIDESET|NSETS|SSETS id_i …
DELETE QAINFO| QA| INFORMATION

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

DELETE QAINFO, QA, or INFORMATION deletes the *EXODUS* quality assurance records and information records, only the quality assurance records, or only the information records, respectively. After deleting the specified records, a new INFO record is added that documents the deletion†.

---

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

† This command is not intended to undermine or in any way reduce the Quality Assurance trail associated with an analysis. The command was added as a work-around for bugs found in some of the other codes that could only handle a fixed number of QA or INFO records.

EQUIVALENCE *tolerance*
EQUIVALENCE RESET

EQUIVALENCE results in all nodes in the model that are with *tolerance* distance of each other to be combined into a single node. The equivalencing will be performed after the EXECUTE or EXIT command is specified. There is currently no way to limit the combined nodes based on membership in a nodeset.

EQUIVALENCE RESET removes previous EQUIVALENCE commands and no nodes will be equivalenced.

INCREMENT MATERIAL|NODESET|SIDESET|NSETS|SSETS *increment*…

INCREMENT increments the identification numbers associated with the material blocks, nodesets, or sidesets by the specified *increment* before creating the output *GENESIS* file. This command cannot be undone once it is entered.

SWAP SIDESET|SSETS id,…

SWAP reverses the direction of the sideset normal. It should only be used on sidesets that are defined on the faces of SHELL elements.

2.3 Information and Processing

END|EXIT

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

EXECUTE

EXECUTE performs all currently specified transformations and then returns to the command handler for additional transformations.

HELP *command* <general help>

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

LIMITS

LIMITS displays the minimum, maximum, and range of the X, Y, and Z coordinates of the input mesh.

LIST *option*

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

SSETS, QA, INFORMATION, and COMMANDS.

LIST {VARS|VARIABLES}

displays a summary of the database. The summary includes the database title; the number of nodes, elements, and element blocks; the number of node sets and side sets; and the number of each type of variable.

LIST {BLOCKS|MATERIALS}

displays a summary of the element blocks. The summary includes the block identifier, the number of elements in the block, the element number of the first and last element in the block, and the block status, either selected or not selected.

LIST {NSETS|NODESETS}

displays a summary of the node sets. The summary includes the set identifier and the number of nodes in the set.

LIST {SSETS|SIDESETS}

displays a summary of the side sets. The summary includes the set identifier, the number of elements in the set, and the number of nodes in the set.

LIST {QA|INFORMATION}

displays the quality assurance (QA) records of the information (INFO) records which maintain the history of the *GENESIS* file.

QUIT

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

SHOW *command* <no parameter>

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

2.4 Order of Transformation

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

SNAP or MOVE, DELETE, REVOLVE, OFFSET, MIRROR, RANDOMIZE, ZERO, SCALE, EXPLODE, and SMOOTH.

Processing of transformation commands does not occur until an EXECUTE, EXIT or END command is entered.

If order-dependent processing is required, the EXECUTE command must be used. The EXECUTE command performs all currently specified transformations and then returns to the command handler for additional transformations. An example of this follows:

```
scale x 1.1074
execute
revolve x 30 y 30
execute
zero x 1.0e-5 y 1.0e-5
execute
scale y 12.5
```

Unlike the transformation commands, the CHANGE operations are performed in the order they are entered, and at the time they are entered.

3 GREPOS Example Input and Output

A few examples of *GREPOS* commands are shown below. In the examples below, lines beginning with the string "**GREPOS>**" show the user input and the lines following are the *GREPOS* output. Lines in Italic type indicate comment about the command or the output and will not appear during a run of *GREPOS*.

```
GREPOS> revolve x 90 y 30 z 20
 Rotation matrix for generated mesh:
    .8138    .2962   -.5000
    .4698    .1710    .8660
    .3420   -.9397    .0000

GREPOS> revcen 0.0 0.0 0.0
 Center of revolution =  .0000  .0000  .0000

GREPOS> offset  x 10 y -10 z 13
 Coordinate offsets =  10.00 -10.00  13.00

GREPOS> adjust min x 10 max y 12 center z 20
 Coordinate offsets = 10.00 12.00 10.00
```

(Note: the previous offset command is overwritten by this command.)

```
GREPOS> offset reset
 Coordinate offsets =  .0000  .0000  .0000

GREPOS> scale x 2.54
 Coordinate scale factors = 2.540 1.000 1.000

GREPOS> scale all 2.54
 Coordinate scale factors = 6.452 2.540 2.540
```

(Note: Scalings are cumulative)

```
GREPOS> scale reset all 2.54
 Coordinate scale factors = 2.540 2.540 2.540

GREPOS> mirror x z
    New X = - Old X
    New Y =   Old Y
    New Z = - Old Z

GREPOS> randomize x 1.0e-5
 Coordinate random factors = 10.00E-6   .00E-6   .00E-6

GREPOS> smooth
 Smoothing Type = LAPLACIAN
 Tolerance      = 1.000E-6
 Iterations     = 100
 Relaxation Par = 1.000
```

GREPOS> list block

```
Block    1 (#1):   230 elements   4-node   0 attributes
   Element block type = "QUAD    "
Block    2 (#2):    52 elements   2-node   0 attributes
   Element block type = "TRUSS   "
```

Two material blocks in model, IDs are 1 and 2

GREPOS> change material 1 2

```
ERROR - Cannot change to an existing ID
```

Material ID 2 already exists in model

GREPOS> change material 1 10

```
*** Material 1 changed to Material 10
```

GREPOS> change type 2 BAR

```
*** Material Block 2 type changed to BAR
```

GREPOS> list

```
Valid LIST options
  SSETS    SIDESETS  NSETS     NODESETS  VARS      VARIABLE  BLOCKS
   MATERIAL COMMANDS INFORMAT  QA
```

GREPOS> limits

```
Input Mesh Limits:
Min X =  0.00000E+00, Max X =  1.20000E+00, Range =  1.20000E+00
Min Y = -1.00000E-01, Max Y =  0.00000E+00, Range =  1.00000E-01
Min Z =  0.00000E+00, Max Z =  2.78000E+00, Range =  2.78000E+00
```

GREPOS> increment material 10

```
*** Material 10 changed to Material 20
*** Material  2 changed to Material 12
```

GREPOS> exit

No more input, process the file.

4 Informational and Error Messages

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

```
DATABASE ERROR - Reading database item and aborts.
```

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

When the command input is complete, *GREPOS* transforms the mesh and writes the output database. *GREPOS* allocates memory dynamically as it is needed. If the system runs out of memory, the following message is printed:

```
FATAL ERROR - Too much dynamic memory requested
```

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

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

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

```
Cannot change to an existing ID
```

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

```
Node not connected to any elements
```

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

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

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

5 References

[1] G. D. Sjaardema, “Overview of the Sandia National Laboratories Engineering Analysis Code Access System,” Technical Report SAND92–2292, Sandia National Laboratories, Albuquerque, New Mexico, January 1993.

[2] T. D. Blacker, “FASTQ Users Manual, Version 2.1,” Technical Report SAND88–1326, Sandia National Laboratories, Albuquerque, New Mexico, July 1988.

[3] A. P. Gilkey and G. D. Sjaardema, “GEN3D: A GENESIS Database 2D to 3D Transformation Program,” Technical Report SAND89-0485, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

[4] G. D. Sjaardema, “GENSHELL: A GENESIS Database 2D to Shell Transformation Program,” technical report, Sandia National Laboratories, Albuquerque, New Mexico. In preparation.

[5] G. D. Sjaardema, “GJOIN: A Program for Merging Two or More GENESIS Databases,” Technical Report SAND92-2290, Sandia National Laboratories, Albuquerque, New Mexico, December 1992.

[6] G. D. Sjaardema, “Aprepro: An Algebraic Preprocessor for Parameterizing Finite Element Analyses,” Technical Report SAND92–2291, Sandia National Laboratories, Albuquerque, New Mexico, December 1992.

[7] G. D. Sjaardema, “GREPOS: A GENESIS Database Repositioning Program,” Technical Report SAND90-0566, Sandia National Laboratories, Albuquerque, New Mexico, April 1990.

[8] G. D. Sjaardema, “NUMBERS: A Collection of Utilities for Pre- and Postprocessing Two- and Three-Dimensional EXODUS Finite Element Models,” Technical Report SAND88–0737, Sandia National Laboratories, Albuquerque, New Mexico, March 1989.

[9] A. P. Gilkey, “BLOT—A Mesh and Curve Plot Program for the Output of a Finite Element Analysis,” Technical Report SAND88–1432, Sandia National Laboratories, Albuquerque, New Mexico, June 1989.

[10] W. C. Mills-Curran, A. P. Gilkey, and D. P. Flanagan, “EXODUS: A Finite Element File Format for Pre- and Post-processing,” Technical Report SAND87–2977, Sandia National Laboratories, Albuquerque, New Mexico, September 1988.

[11] L. M. Taylor, D. P. Flanagan, and W. C. Mills-Curran, “The GENESIS Finite Element Mesh File Format,” Technical Report SAND86–0910, Sandia National Laboratories, Albuquerque, New Mexico, May 1986.

[12] D. P. Flanagan, W. C. Mills-Curran, and L. M. Taylor, “SUPES A Software Utilities Package for the Engineering Sciences,” Technical Report SAND86-0911, Sandia National Laboratories, Albuquerque, New Mexico, September 1986.

[13] “American National Standard Programming Language FORTRAN,” Technical Report ANSI X3.9–1978, American National Standards Institute, Inc., New York, 1978.

[14] B. Berliner, “CVS II: Parallelizing Software Development,” in *Proceedings of the Winter 1990 USENIX Conference*, 1990.

Intentionally Left Blank

A The GENESIS Database Format

The following code segment reads a *GENESIS* database.

```
C   --Open the GENESIS database file
      NDB = 9
      OPEN (UNIT=NDB, ..., STATUS='OLD', FORM='UNFORMATTED')
C   --Read the title
      READ (NDB) TITLE
C      --TITLE - the title of the database (CHARACTER*80)
C   --Read the database sizing parameters
      READ (NDB) NUMNP, NDIM, NUMEL, NELBLK,
     &   NUMNPS, LNPSNL, NUMESS, LESSEL, LESSNL
C      --NUMNP - the number of nodes
C      --NDIM - the number of coordinates per node
C      --NUMEL - the number of elements
C      --NELBLK - the number of element blocks
C      --NUMNPS - the number of node sets
C      --LNPSNL - the length of the node sets node list
C      --NUMESS - the number of side sets
C      --LESSEL - the length of the side sets element list
C      --LESSNL - the length of the side sets node list
C   --Read the nodal coordinates
      READ (NDB) ((CORD(INP,I), INP=1,NUMNP), I=1,NDIM)
C   --Read the element order map (each element must be listed once)
      READ (NDB) (MAPEL(IEL), IEL=1,NUMEL)
C   --Read the element blocks
      DO 100 IEB = 1, NELBLK
C      --Read the sizing parameters for this element block
         READ (NDB) IDELB, NUMELB, NUMLNK, NATRIB
C         --IDELB - the element block identification (must be unique)
C         --NUMELB - the number of elements in this block
C         --   (the sum of NUMELB for all blocks must equal NUMEL)
C         --NUMLNK - the number of nodes defining the connectivity
C         --   for an element in this block
C         --NATRIB - the number of element attributes for an element
C         --   in this block
C      --Read the connectivity for all elements in this block
         READ (NDB) ((LINK(J,I), J=1,NUMLNK, I=1,NUMELB)
C      --Read the attributes for all elements in this block
         READ (NDB) ((ATRIB(J,I), J=1,NATRIB, I=1,NUMELB)
 100 CONTINUE
C   --Read the node sets
      READ (NDB) (IDNPS(I), I=1,NUMNPS)
C      --IDNPS - the ID of each node set
      READ (NDB) (NNNPS(I), I=1,NUMNPS)
C      --NNNPS - the number of nodes in each node set
      READ (NDB) (IXNNPS(I), I=1,NUMNPS)
C      --IXNNPS - the index of the first node in each node set
C      --   (in LTNNPS and FACNPS)
      READ (NDB) (LTNNPS(I), I=1,LNPSNL)
C      --LTNNPS - the nodes in all the node sets
      READ (NDB) (FACNPS(I), I=1,LNPSNL)
```

```
C      --FACNPS - the factor for each node in LTNNPS
C   --Read the side sets
       READ (NDB) (IDESS(I), I=1,NUMESS)
C      --IDESS - the ID of each side set
       READ (NDB) (NEESS(I), I=1,NUMESS)
C      --NEESS - the number of elements in each side set
       READ (NDB) (NNESS(I), I=1,NUMESS)
C      --NNESS - the number of nodes in each side set
       READ (NDB) (IXEESS(I), I=1,NUMESS)
C      --IXEESS - the index of the first element in each side set
C      --   (in LTEESS)
       READ (NDB) (IXNESS(I), I=1,NUMESS)
C      --IXNESS - the index of the first node in each side set
C      --   (in LTNESS and FACESS)
       READ (NDB) (LTEESS(I), I=1,LESSEL)
C      --LTEESS - the elements in all the side sets
       READ (NDB) (LTNESS(I), I=1,LESSNL)
C      --LTNESS - the nodes in all the side sets
       READ (NDB) (FACESS(I), I=1,LESSNL)
C      --FACESS - the factor for each node in LTNESS
```

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

```
C   --Read the QA header information
       READ (NDB, END=...) NQAREC
C      --NQAREC - the number of QA records (must be at least 1)
       DO 110 IQA = 1, MAX(1,NQAREC)
          READ (NDB) (QATITL(I,IQA), I=1,4)
C         --QATITL - the QA title records; each record contains:
C         --   1) analysis code name (CHARACTER*8)
C         --   2) analysis code qa descriptor (CHARACTER*8)
C         --   3) analysis date (CHARACTER*8)
C         --   4) analysis time (CHARACTER*8)
  110  CONTINUE
C   --Read the optional header text
       READ (NDB, END=...) NINFO
C      --NINFO - the number of information records
       DO 120 I = 1, NINFO
          READ (NDB) INFO(I)
C         --INFO - extra information records (optional) that contain
C         --   any supportive documentation that the analysis code
C         --   developer wishes (CHARACTER*80)
  120  CONTINUE
C   --Read the coordinate names
       READ (NDB, END=...) (NAMECO(I), I=1,NDIM)
C      --NAMECO - the coordinate names (CHARACTER*8)
C   --Read the element type names
       READ (NDB, END=...) (NAMELB(I), I=1,NELBLK)
C      --NAMELB - the element type names (CHARACTER*8)
```

B Command Summary

Mesh Orientation (page 14)

ADJUST {MINIMUM|MAXIMUM} *axis, value*

set minimum or maximum extent of mesh to *value*

EXPLODE

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

MIRROR $axis_1$, $axis_2$, …
MIRROR RESET

causes the mesh to be reflected about the specified axes, or resets the mesh to no reflections.

OFFSET [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, …
OFFSET [ADD] ALL *offset*
OFFSET RESET
OFFSET *xoff, yoff, zoff*
OFFSET SPLINE

specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for SHIFT command.

RANDOMIZE $axis_1$, $magnitude_1$, $axis_2$, $magnitude_2$, …
RANDOMIZE ALL *magnitude*
RANDOMIZE RESET
RANDOMIZE BLOCK

randomly offset nodes

REVCEN *xcen, ycen, zcen*

sets the center of rotation for the REVOLVE command.

REVOLVE $axis_1$, $ndeg_1$, $axis_2$, $ndeg_2$, …
REVOLVE RESET

causes the mesh to be rotated, or resets the mesh to no rotations.

SCALE $axis_1$, $scale_1$, $axis_2$, $scale_2$, …
SCALE ALL *scale_factor*
SCALE RESET

causes the mesh to be scaled, or resets the mesh to no scaling.

SCALE ATTRIBUTE *number* BLOCK *id scale_factor*
SCALE ATTRIBUTE ALL BLOCK *id scale_factor*
SCALE ATTRIBUTE *number* BLOCK ALL *scale_factor*
SCALE ATTRIBUTE ALL BLOCK ALL *scale_factor*
SCALE ATTRIBUTE RESET

causes the specified attribute of the specified element block to be multiplied by the scaling multiplier *scale_factor*.

SCALE BLOCK

causes each material block in the mesh to be scaled by the specified factor

SHIFT [ADD] $axis_1$, $offset_1$, $axis_2$, $offset_2$, …
SHIFT [ADD] ALL *offset*
SHIFT RESET
SHIFT *xoff, yoff, zoff*
SHIFT SPLINE

specifies the coordinate offsets for the mesh, or resets the mesh to no offsets. Synonym for OFFSET command.

SMOOTH *tolerance, iterations, relaxation-factor*

smooth mesh to improve element shape

SNAP *slave_sideset_id* TO *master_sideset_id*
[NORMAL|X|Y|Z|MINUSX|MINUSY|MINUSZ] [TOLER *tolerance*]
SNAP *slave_sideset_id* TO *master_sideset_id* [VECTOR *vx vy vz*] [TOLER *tolerance*]
SNAP *slave_sideset_id* TO *master_sideset_id* [CENTER *cx cx cz*] [TOLER *tolerance*]

moves nodes in *slave_sideset_id* to lie on the element faces in *master_sideset_id*.

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

sets all $axis_i$ coordinates with an absolute value less than min_i equal to zero, or resets the mesh to no automatic zeroing.

Modification of Material, Nodeset, or Sideset IDs (page 19)

CHANGE MATERIAL|NODESET|SIDESET|NSETS|SSETS *old_id new_id*
CHANGE TYPE *block_id new_type*

modifies the identification number of the material block, nodeset, or sideset; or modifies the element type of the material block.

CHANGE ATTRIBUTE *number* BLOCK *id value*
CHANGE ATTRIBUTE ALL BLOCK *id value*
CHANGE ATTRIBUTE *number* BLOCK ALL *value*
CHANGE ATTRIBUTE ALL BLOCK ALL *value*
CHANGE ATTRIBUTE RESET

causes the specified attribute of the specified element block to be set to the specified *value.*

DELETE MATERIAL|NODESET|SIDESET|NSETS|SSETS *id* …
DELETE|QA|INFORMATION|QAINFO

deletes the material block, nodeset, or sideset with ID *id;* or delete the quality assurance records, informational records, or both.

INCREMENT MATERIAL|NODESET|SIDESET|NSETS|SSETS *increment.*

increment identification numbers

SWAP SIDESET|SSETS id,…

reverse orientation of sideset for shell elements.

Information and Processing (page 21)

END or EXIT

ends command input and starts processing.

HELP *command*

displays information about a *GREPOS* command.

LIMITS

display minimum, maximum, and range of X, Y, and Z coordinates in mesh.

LIST *option*

displays the database information specified by *option*.
LIST {VARS|VARIABLES}
LIST {BLOCKS|MATERIALS}
LIST {NSETS|NODESETS}
LIST {SSETS|SIDESETS}
LIST {QA|INFORMATION}
LIST COMMANDS

QUIT

quits command input and aborts processing.

SHOW *command*

displays the processing parameters set by a command.

Order of operations:
DELETE, REVOLVE, OFFSET, MIRROR, RANDOMIZE, ZERO, SCALE, EXPLODE, and SMOOTH

C GREPOS Details

Version:

The current version of *GREPOS* is 1.20, modified on 1998/05/20.

Execution:

To execute *GREPOS* on a *UNIX** system (with SEACAS), type:

grepos [-options *option*] *input_database output_database*

Input_database is the filename of the input *GENESIS* database.
Output_database is the filename of the output *GENESIS* database

Valid *options* are:

-executable = *alternate-executable* to specify running a different version of grepos,

-aprepro to pipe the input through the program aprepro [6],

-command=*single-line-command* to run grepos with just a single command given on the command line instead of interactively or in an input file.

-help to get a usage synopsis,

-VMS to indicate that the *EXODUS* file is in VAX/VMS binary format†, and

-IEEE to indicate that the *EXODUS* file is in IEEE binary format‡.

User input is read from the terminal keyboard (unless redirected using '<').
User output is directed to the terminal.

Execution Files

The table below summarizes *GREPOS* file usage.

Description	Unit	Type	File Format
User input	std input	input	ASCII
User output	std output	output	ASCII
GENESIS database	9	input	*GENESIS*
GENESIS database	10	output	*GENESIS*

All files must be connected to the appropriate unit before *GREPOS* is run. Each database file is opened with the name retrieved by the EXNAME routine of the *SUPES* [12] library.

---

* UNIX is a registered trademark of UNIX Systems Laboratories Inc.
† Cray Unicos systems only
‡ Cray Unicos systems only

Source Code:

The *GREPOS* source code is maintained in the SEACAS system which is managed by the Concurrent Version System (cvs) [14]. *GREPOS* is written in ANSI standard FORTRAN-77 [13] with the exception of the following extension:

- Include files are used.

GREPOS uses the following software package:

- the *SUPES* [12] package which includes dynamic memory allocation, a free-field reader, and FORTRAN extensions.
- the *SUPLIB* package which is an undocumented internal development library which includes *GENESIS* reading and writing routines, command parsing, string utilities, and other miscellaneous useful functions.

Availability:

GREPOS and all other SEACAS codes are available on a licensed basis. The license agreements for these codes stipulate that (1) the software is to be used solely for internal purposes, (2) the codes are not to be distributed or transferred to any person without written permission, (3) the codes are to be used at a single site and should be copied only for necessary maintenance, development, or backup purposes, and (4) there should be a procedure, or site plan, in place for protecting the provisions of the license agreements.

For more information on obtaining *GREPOS* or other SEACAS codes, contact:

Michael C. Maurer
Engineering Sciences Center, MS-0841
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-5800
(505) 845-0900, FAX: (505) 844-8081