Sandia National Laboratories
Albuquerque, New Mexico 87185

date: February 22, 1989

to: Distribution

[illegible]

from: M. B. Stephenson, T. D. Blacker, 1523

subject: FASTQ Newsletter No. 6, *Version 1.30*

This newsletter is to document continuing changes and updates to FASTQ. The changes documented in this memo are available in *Version 1.30* of the code. The following topics will be discussed:

1. Processing of holes in mesh primitives.
2. Laplacian-Isoparametric smoothing.
3. Material layers in a region.
4. Digitizer enhancements.

1 Processing of holes in mesh primitives

The inclusion of circular holes in FASTQ primitives provides an easy method for analysts to mesh around holes without having to explicitly subdivide the region into smaller primitives. The new *Z* scheme is used to process a hole in a region. This section describes:

Copy to:

1265 W. A. Johnson	1523 J. H. Biffle & Staff
1412 J. C. Bushnell	1524 A. K. Miller & Staff
1412 Y. T. Lin	1530 D. B. Hayes
1412 J. L. Mitchiner	1550 C. W. Peterson
1412 L. R. Phillips	2541 A. Gonzales
1510 J. W. Nunziato	5165 N. R. Hansen
1511 D. K. Gartling	5165 R. K. Thomas
1512 J. C. Cummings	6314 L. S. Costin
1513 D. W. Larson	6323 H. R. Yoshimura
1513 R. E. Hogan	6332 T. M. Torres
1520 L. W. Davison	7133 E. L. Hoffman
1521 R. D. Kreig & Staff	7133 H. C. Walling
1522 R. C. Reuter & Staff	8241 K. J. Perano

1. The method used to define circular holes within regions,
2. The technique **FASTQ** uses to process the holes,
3. The schemes available to control hole region processing,
4. The procedure used for meshing the interior of holes, and
5. Other hole related commands.

1.1 Defining Holes in Regions

The HOLE CARD or the *KEYIN HOLE* command are the two methods used to define holes.

Hole Card. The hole card has 3 or more fields:

1. The string HOLE to identify the card.
2. The number of the region containing the hole(s).
3. One or more region numbers that define the hole(s).

The hole card may appear anywhere in the **FASTQ** input deck as long as it follows the definition of the region that contains the holes.

Examples:

```
HOLE 1 2
HOLE 9 13 14 29 30 32 33 34 35
```

In the first example above, one hole, Region 2, is defined to be within the surrounding Region 1. In the second example, Region 9 has eight holes in it defined by Regions 13, 14, 29, etc. Regions 1 and 9 must be defined before these hole cards are encountered in the input deck.

Currently, the hole region must be defined as a single *CIRC* line. The four lines below would be sufficient to define the hole region in the first card above.

```
POINT 5 1.5 1.5
POINT 8 1.5 5.0
LINE 5 CIRC 8 8 5 1 1.0
REGION 2 2 -5
```

Interactive Hole Specification. Interactive hole specification occurs in the *KEYIN* option of **FASTQ**. If the user chooses the *HOLE* suboption of the *KEYIN* option, **FASTQ** will display the following prompt:

```
ENTER HOLE DATA IN THE FOLLOWING FORMAT:
[ REGION NO., HOLE 1, HOLE 2, ..., HOLE N ]
HIT RETURN TO END INPUT
```

A response of:

```
> 200,7,3,22
```

specifies that Region 200 contains hole Regions 7, 3 and 22. Region 200 must be defined previously for this hole command to be accepted.

1.2 The How of Hole Processing

FASTQ generates a mesh around a hole using the following procedure:

1. It deletes all elements that have one or more nodes within the hole, forming a *hole-void* in the mesh slightly larger than the hole.
2. It *squares* the boundary of the hole-void by deleting additional elements left in the corners. The number of element sides exposed by this deletion process determines the resulting number of intervals assigned to the circle.[1]

   The derived number of intervals for the hole is stored so that the hole region itself may be meshed, see Figure 1.[2] This implies that the region containing the hole should be meshed before the hole region.
3. It subdivides the hole perimeter, and generates new elements by connecting the hole perimeter to the hole-void boundary.
4. If additional rows of elements are specified by the hole scheme, it inserts these element rows between the hole perimeter and the hole-void boundary (see L - Laminate hole).

Once the hole is generated, all of the smoothing and restructuring techniques available in **FASTQ** [1] may be used on the mesh.

1.3 Schemes available for hole processing

Defining a hole in a region does not automatically mean that the hole will be processed.

Z - Process hole. The scheme for the region must include a *Z* for each hole that is added to the mesh. Using multiple *Z*'s allows the user to interact with the hole processor when in

[1]The user may specify a minimum number of intervals on the line forming the circle, and the algorithm will continue the deletion of additional rows until this minimum is met or all elements are deleted.

[2]For interval assignments less than 4, **FASTQ** will not mesh the interior of a hole region. It will detect a perimeter with not enough intervals. Processing the region a second time will correctly mesh the hole's interior, since the hole processor assigned intervals to the perimeter during the first pass.

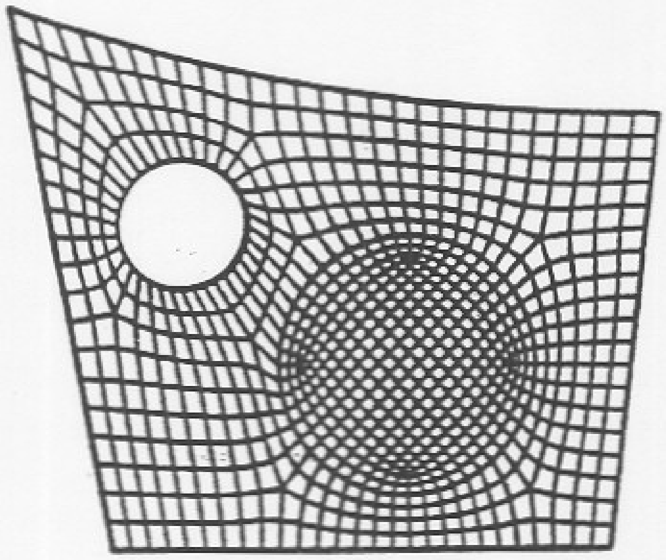


Figure 1. Irregular Rectangle Mesh with Two Holes

stepping mode. The mesh may be plotted, smoothed, etc. between the processing of each hole. On the other hand, if the user does not want to exercise that much control, the *Z* scheme may be enclosed in parentheses, i.e. *(Z)*, which will cause all holes within the region to be processed. The following hole processing schemes produce the same result:

MZZZZZ2S
M(Z)2S

In both cases, FASTQ generates a rectangular mesh, punches five holes in it, and then reshapes it using area-pull and Laplacian smoothing.

L - Laminate hole. If the scheme for the hole, i.e. the region defining the hole, includes one or more *L*'s, then a row of elements is laminated between the hole and the hole-void for each *L*. For example the hole scheme for the holes in Figure 1 is given below.

SCHEME 2 M5NSNSNSL
SCHEME 3 L

Both schemes include one *L* meaning that one additional row of elements will be inserted between the hole perimeter and the hole-void. The rest of the scheme for region two is used when that region is meshed, and the *L* scheme is ignored.

In stepping mode, the program reports the current hole scheme, as it does the region scheme, and asks if the user wants to change it. The example below shows how to add two rows of

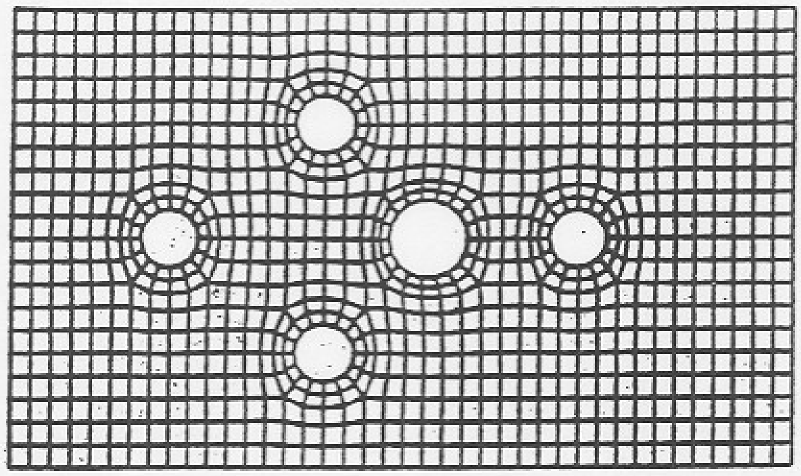


Figure 2. Rectangle Mesh with Five Holes

elements around the large hole in Figure 1 instead of the one specified in the initial scheme.

```
INITIAL MESH DEFINED USING THIS HOLE SCHEME:
M5NSNSNSL

USE CURRENT HOLE SCHEME? N
ENTER HOLE PROCESSING SCHEME: LLM5NSNSNS
```

The order of the *L*'s is not important. They may be first, last or anywhere else in the scheme.

The scheme entered during step processing is saved for later use when and if the hole region itself is meshed. Therefore, if you intend to mesh the hole interior, remember to include its scheme when modifying the number of *L*'s for hole processing.

1.4 Examples

The two meshes in Figure 2 and Figure 3 illustrate the use of the new hole processing capability.[3] Figure 2 is a rectangular mesh with five holes. The scheme for this mesh was used as an example in Section 1.3. Figure 3 shows an irregular shaped region meshed as a semi-circle. It has thirty-nine holes of four different sizes punched in it. Using the *(Z)* scheme provides an easy way to mesh all of the holes without first counting them.

[3]The FASTQ data files used to create these two meshes are stored on the 1520 VAX cluster in FASTQ$DIR:HOLE1.FSQ and FASTQ$DIR:HOLE2.FSQ respectively. Please feel free to copy and experiment with them.

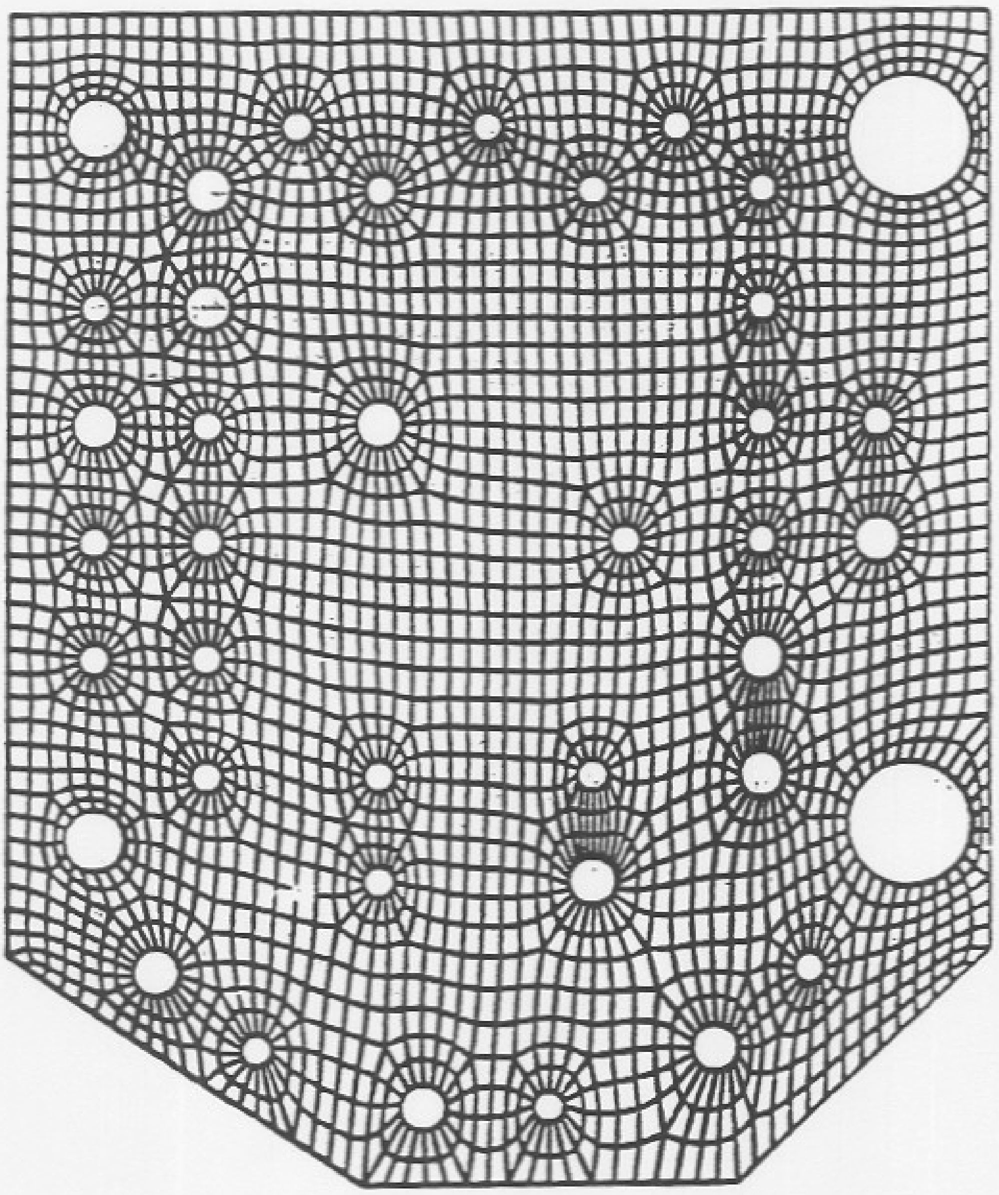


Figure 3. Semi-Circle with Thirty-Nine Various Sized Holes

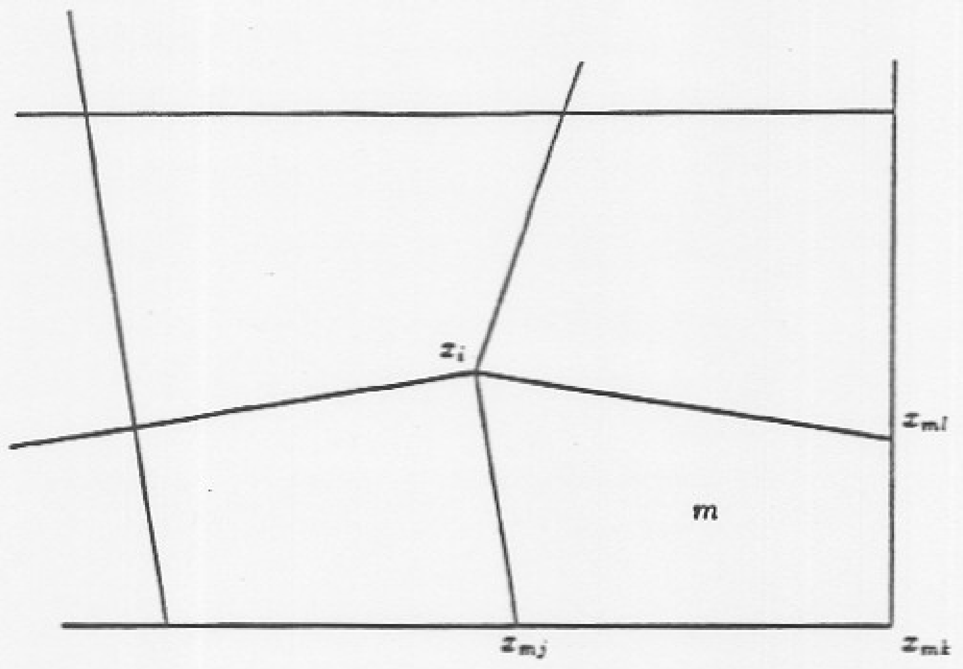


Figure 4. Laplacian-Isoparametric Equation Variables

1.5 Other hole related commands

The *READ*, *WRITE*, *LIST*, and *DELETE* commands now support holes as well. The changes to *READ* and *WRITE* should be transparent to the user. The *LIST* and *DELETE* commands include a new *HOLE* option.

2 Laplacian-Isoparametric smoothing

A new smoothing scheme has been added to FASTQ. It implements the Laplacian-Isoparametric smoothing technique developed by Herrmann [4].

Laplacian-Isoparametric smoothing is defined by Equation 1 where x_i is a coordinate surrounded by n elements. The x_{mj}, x_{mk} and x_{ml} are the other coordinates at nodes j, k and l of the m^{th} element in a clockwise or counter-clockwise order as illustrated in Figure 4.

The weighting factor is w.

$$x_i = \frac{1}{n(2-w)} \sum_{m=1}^{n} x_{mj} + x_{ml} - w x_{mk} \tag{1}$$

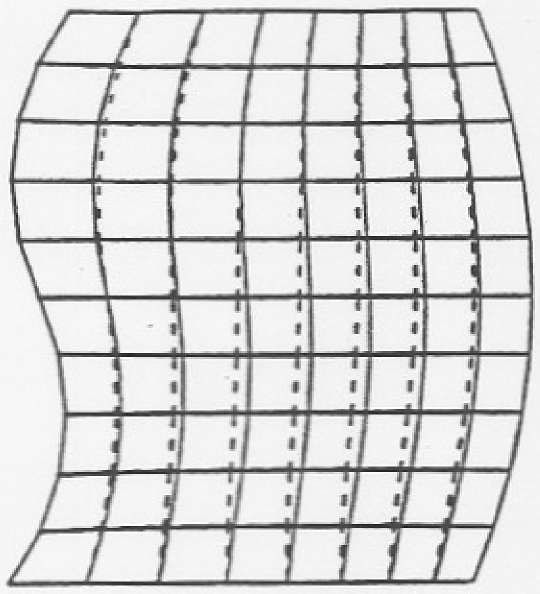


Figure 5. Laplacian-Isoparametric Smoothing

The isoparametric aspect of this smoother attempts to keep the grid lines aligned with the parametric directions of the mesh.

7 - Laplacian-Isoparametric smoother. A *7* in a scheme will cause the smoother type to be set to Laplacian-Isoparametric. The *S* scheme will invoke the active smoother.

2.1 Weighting factor

When w in Equation 1 is 0, the smoothing is the same as scheme *5*, Laplacian. When w is 1, the smoothing is totally isoparametric. The default value is 0.7.

Y - Increase or decrease the isoparametric weight. Use the *Y* or *+Y* scheme to increase the isoparametric weighting by 0.1. Use the *-Y* scheme to decrease the weighting by 0.1.

In Figure 5 the dashed lines show the mesh after smoothing with the straight Laplacian scheme. The solid lines show the mesh after smoothing with the combined Laplacian-Isoparametric scheme using the default weighting factor of 0.7.

3 Material layers in a region

A new meshing capability has been added to allow the meshing of regions containing element rows of differing materials. This capability is useful in modeling logical rectangular[1] regions of sandwiched, layered, or wound materials. The element layers can be placed in any user defined order, and can alternate in any user defined sequence.

To enable this capability, the user must first define a logical rectangular region. The layered direction follows the first side chosen by **FASTQ** for meshing. To insure that **FASTQ** chooses the correct side to begin the layering, this side or line should be listed as the first side or line in the *REGION* definition card, and the region should be given a forced rectangle scheme such as *P*.

The layered material numbers or Block ID's should then be defined as a *SIDE*. [4] This side definition number is then negated and assigned to the region as its Block ID.

When meshing the region, **FASTQ** loops through the Block ID numbers in the specified *SIDE* data set, assigning each subsequent row of elements the next Block ID in the set. Once the entire set has been used, it is repeated until all the element rows have been assigned the appropriate Block ID. Thus if alternating rows of two materials are to be modeled, the *SIDE* data set need only contain two Block ID numbers.

As an example, the following cards in a **FASTQ** file would produce the mesh shown in Figure 6. The BLOCK ID numbers are displayed at the center of the elements.

```
TITLE
EXAMPLE OF A LAYERED MATERIAL REGION
POINT 1 2.0 1.0
POINT 2 4.0 8.0
POINT 3 1.4 1.1
POINT 4 7.0 1.6
POINT 5 1.9 6.0
LINE 1 STR 1 2 0 9 1.0
LINE 2 CIRC 2 3 4 14 1.0
LINE 3 STR 3 5 0 9 1.0
LINE 4 CIRC 1 5 4 14 1.0
SIDE 100 1 4 3 4
REGION 1 -100 -4 -3 -2 -1
SCHEME 1 P
SCHEME 0 M
BODY 1
EXIT
```

[4] *SIDE* definitions may be used to indicate sets of data in **FASTQ**.[1]

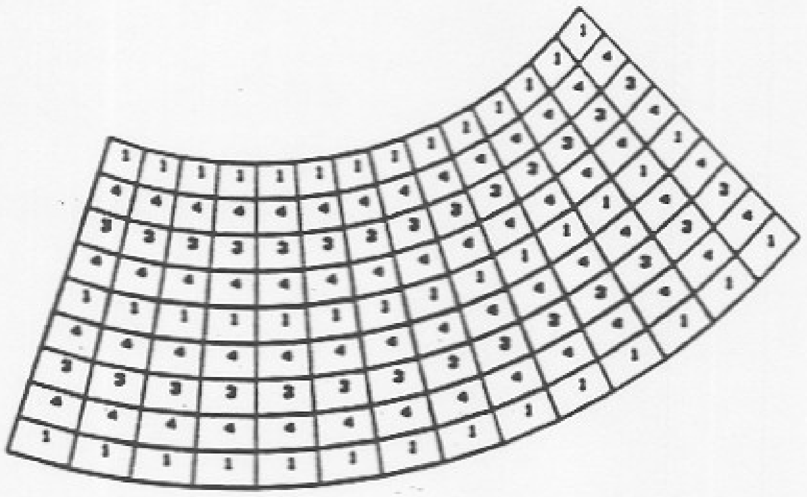


Figure 6. Layered Material Mesh

4 TABLET Option

The previous versions of **FASTQ** contained a digitizer option which allowed the input of geometry from a drawing using a puck and digitizer tablet. This method suffered from tablet input inaccuracies and from the fact that true scale drawings of the geometry were often not available. These concerns have been eliminated with the implementation of *SNAP* grid lines, an improved digitizer initialization technique, and an enhancement of the existing digitizer functionality. The *DIGITIZE* option in **FASTQ** has been replaced with a new *TABLET* option to provide the functionality needed with these enhancements.

4.1 SNAP Grid Lines

SNAP grid lines enable the definition of x and y lines which form a grid across the digitizer tablet. As points are input in the definition of the geometry, these points are moved or *snapped* to the closest grid, or intersection of the grid lines. In this manner, any digitizer error can be eliminated when entering existing scale drawings, and geometries not available as a true scale drawing can be input quickly and accurately. Adjustments or corrections of the input geometry can be made quickly by simply moving grid lines and then *snapping* points back to the appropriate grids.

The *SNAP* functionality can be toggled *ON* or *OFF* using the *TABLET SNAP* option. The default is *ON*. This toggle can also be set from the **FASTQ** file using the SNAP CARD explained in Section 4.2.

Grid lines can be defined in the following ways:

- **Default Grid With Defined Geometry.** If there are less than two grid lines defined in either the x or y direction when entering the *TABLET* option, a default grid is defined for the user. This grid passes through every point in the geometry just as the *TABLET POINT* option explained below.

- **Default Grid With No Defined Geometry.** If there are less than two grid lines defined in either the x or y direction and the *SNAP* toggle is on when entering the *TABLET DIGITIZE* option, **FASTQ** will propose a square uniformly spaced default grid. This grid is based on the current plotting extremes. See Section 4.3 for a discussion on how these extremes are set.

  In determining the uniform x and y spacing, **FASTQ** divides the x and y dimensions by 20 and 15 respectively, rounds the larger of the spacings to the nearest significant digit, and uses this spacing for both the x and y axis. If the proposed spacing is rejected by the user, **FASTQ** asks if the user wishes to continue with the *SNAP* option *OFF*; otherwise, the user must define an acceptable grid before entering the *TABLET DIGITIZE* option.

- **Uniform Square Grid.** A square uniformly spaced grid can be defined using the *TABLET UNIFORM* option. The user is prompted for a minimum and maximum x and y value, and a spacing size. The maximum need not be an exact multiple of the spacing size increment from the minimum. **FASTQ** generates grid lines in both the x and y directions by incrementing the minimum value until the grid being added exceeds or equals the maximum value in that direction.

- **Uniform X Spaced Grid.** Uniformly spaced x grid lines can be defined using the *TABLET UX* option. The user is prompted for a minimum and maximum x value, and a spacing size. The maximum need not be an exact multiple of the spacing size increment from the minimum. **FASTQ** generates x grid lines by incrementing the minimum x value until the grid being added exceeds or equals the maximum x value.

- **Uniform Y Spaced Grid.** Uniformly spaced y grid lines can be defined using the *TABLET UY* option. The user is prompted for a minimum and maximum y value, and a spacing size. The maximum need not be an exact multiple of the spacing size increment from the minimum. **FASTQ** generates y grid lines by incrementing the minimum y value until the grid being added exceeds or equals the maximum y value.

- **Arbitrary X Spaced Grid.** Arbitrarily spaced x grid lines can be defined using the *TABLET XGRID* option. The user is prompted for x grid values. These values need not be input in increasing numerical order and multiple values can be input on the same line. A carriage return will end input.

- **Arbitrary Y Spaced Grid.** Arbitrarily spaced y grid lines can be defined using the *TABLET YGRID* option. The user is prompted for y grid values. These values need not be input in increasing numerical order and multiple values can be input on the same line. A carriage return will end input.

- **Point Location Grid.** Grid lines can be defined which pass through all the existing geometric data points using the *TABLET POINT* option.

All grid line definitions are cumulative in that new grids are added to the existing grid line definitions. If a grid line is defined where one already exists, the new definition is ignored. All current grid lines may be cleared using the *TABLET CLEAR* option, only x or y grid lines may be cleared using the *TABLET XCLEAR* or *TABLET YCLEAR* options respectively. Individual x or y grid lines may be cleared using the puck as explained in Section 4.4.

It is sometimes helpful to display an axis on the plotting screen to aid in locating the cursor. This has been enabled as in the other graphics options[1] with an *AXIS* toggle. The tick marks on the axis will most likely not correspond directly with the grid lines being used, but the relative value of the grid lines can be determined.

4.2 Defining Grids in the Input Deck

Three new cards have been added to the **FASTQ** input deck in order to input and output the snap grid state and the grid line definitions. The three new cards are the SNAP CARD, the XGRID CARD and the YGRID CARD. These cards are described below.

Snap Card. The snap card has 2 fields:

1. The string SNAP to identify the card.
2. Either ON or OFF to indicate if grid *snap* is to be enabled or disabled.

The snap card may appear anywhere in the **FASTQ** input deck.

Examples:

```
SNAP ON
SNAP OFF
```

In the first example above, snapping points to the nearest grid line intersection is enabled while in the second example it is disabled.

Grid Cards.

The x-grid or y-grid card has 2 or more fields:

1. The string XGRID or YGRID to identify the card.
2. One or more numbers indicating the location of the x or y grid lines.

An x-grid or y-grid card may appear anywhere in the **FASTQ** input deck.

Example:

```
XGRID -0.234 -0.159 0.0 0.120 0.8 1.41 2.2 2.45 2.52 13.8 *
16.6 16.9 17.12 17.4 17.5 17.58 17.7 17.83 18.1 18.7
YGRID 6.35 6.455 6.65 1.3 1.44 1.69 1.76 1.97 2.20 2.295 2.42 *
2.685 2.795
YGRID -30.414 -30.35 0.8 1.065 1.25 2.94 3.0 3.11 3.17 3.26 3.341 *
3.53 3.66 6.14 6.34
XGRID 13.98 14.3 14.8 14.925 15.05 15.21 15.57 15.7 15.825 15.95
```

The example above illustrates that the x and y grids may be entered in any order. SUPES free field input is used to read the cards so that standard continuation and comment characters are valid.

4.3 Tablet Initialization and Plotting Limits

Tablet initialization maps locations on the digitizer tablet to the physical coordinates of the geometry being defined. This mapping can be performed in automatic mode or it can be imposed by the user. The *TABLET INITIALIZE* option is used to toggle between automatic and imposed mode, or to set a new user imposed mapping. The *TABLET ZOOM* option as well as the puck (see Section 4.4) can be used to set the extremes of the plotting on the screen. For effective tablet input, the plot on the screen should somehow correspond to the tablet initialization. This correspondence is defined by the initialization mode and the current zoom limits. The *SNAP* grid capability can be used with either initialization mode.

Automatic Tablet Initialization. In automatic initialization mode, the tablet is automatically mapped to the current plotting limits. This is the default mode. Upon entering the *TABLET* option, if no plotting limits have been defined, the limits are set to the extremes of the data. If no data has been defined, the extremes default to 0. to 20. in the x direction and 0. to 15. in the y direction. Whenever these plotting limits are changed with the *TABLET ZOOM* option or the puck, the tablet mapping is adjusted accordingly, i.e. a point plotted at the center of the screen can always be accessed by placing the puck in the center of the digitizer, etc.

Imposed Tablet Initialization. Imposed mode allows the user to impose his/her own mapping onto the tablet. This imposed mapping does not change as the zoom limits on the screen change. This option is used to digitize directly from a scaled drawing. Whenever the mapping is set or changed, the default plotting limits are set to the mapping. This direct correspondence is lost when plotting limits are subsequently adjusted. At times this may be beneficial in that the user may pan to various areas of the drawing for *close-up* views of the geometry as he inputs or manipulates it. The imposed initialization is accomplished by choosing the *TABLET INITIALIZE* option. The user is then requested to enter the values of two points on the geometry as:

INPUT THE DRAWING XMIN, XMAX, YMIN, YMAX:
>

The values input should correspond to a lower left extreme (xmin, ymin) and an upper right extreme (xmax, ymax). These values need not be the absolute extremes of the geometry since they are only used to set the relative scale and position of the imposed mapping to the tablet coordinates.

FASTQ next initializes the tablet by requesting that the two points entered be digitized with the puck, as the prompt indicates:

NOW DIGITIZE THOSE TWO POINTS:
PUSH "PUCK-1" FOR LOWER LEFT
PUSH "PUCK-2" FOR UPPER RIGHT
PUSH "PUCK-E" TO END

Placing the puck cross-hairs over the lower left location and pushing the *1* key will initialize the lower left location. Likewise a *2* key initializes the upper right location. To end initialization, the *E* key must be pushed. If the puck input has been recognized by the program, **FASTQ** will notify the user with one of the following messages:

LOWER LEFT INPUT
UPPER RIGHT INPUT
INITIALIZATION COMPLETE

If the cursor happens to slip, and bad input has possibly occurred when entering the lower left or upper right points, the user simply pushes the appropriate key again at the desired location. Input order is not important, and the latest digitized location for each point will be used when the *E* key ends the imposed initialization.

If an imposed initialization is currently active when the *TABLET INITIALIZE* option is chosen, the user is asked if the imposed initialization should be toggled *OFF*. A *NO* response will result in a redefinition of the imposed mapping, while a *YES* response toggles the initialization to automatic mode.

4.4 Digitizer Enhancements

The tablet input process has been enhanced with new capabilities and conveniences. For one, many of the function keys on the puck have been reassigned. Although these changes will cause some initial confusion for previous users, we believe that the new positions of the keypad functions make them more useable than before.

The digitizer functions have been upgraded as follows:

- **Enhanced Region Closures.** The region definition mechanism allows the user to input the set of lines which enclose the current location of the cursor as a new region.

The initial implementation approximated arcs as straight lines when attempting to close a region about a point. This worked well for arcs with relatively small subtended angles, but otherwise, created problems. This region picking mechanism has been extended to handle arcs as arcs, and thus eliminates previous problems.

- **Point Moving.** The capability to *grab* a point with the puck and to place it in a new location has been added. This allows exploration of various geometric sensitivities in a problem, as well as simple definition corrections.
- **Deletion Enhancements.** The digitizer could previously be used to delete only points and the lines, sides, and regions defined using those points. This has been enhanced to allow the deletion of lines and the sides and regions using those lines and snap grid lines. Also, as a precautionary measure, the point, line, or grid to be deleted is highlighted, and the user is asked to confirm the highlighted item before deletion occurs.
- **Repaint Screen.** A new key definition allows the user to repaint the screen without exiting from the *DIGITIZE* command. This is helpful when the deletion of points, lines, or grid lines causes some of the remaining features to be drawn over in black.
- **Zoom Picks.** Another new key definition allows the user to pick the limits of the zoom window. The user may also toggle between the current and last zoom limits thus making it easy to move between two zoomed views of the model. Finally, the user may reset the zoom limits to their default values by using a simple key combination.

Keypad Definitions. Table 1 lists the digitizer puck keypad functions and their key assignments. This may be used as a quick reference. Note that the *0*, *C* and *D* keys are *prefix* keys and are always used in combination with another key. The following is a complete explanation of each defined puck key sequence.

1 **Enter Point.** *1* enters the current x, y location as a new point, and assigns the next sequential point number to that point. The point location is displayed on the screen with an X. This point number is then stored as the *last point* for further reference.

01 **Get Closest Point.** *01* searches the database for the closest point to the current x, y location of the cursor, draws a box around the point to indicate the point selected, and enters that point as the *last point* for further reference. No new point is entered into the data base or displayed. This allows reference to existing points for line and arc generation and for region closures.

C1 **New Point At Closest Point.** *C1* finds the closest point just as *01* does, but it adds a new point to the data with the same coordinates as the closest point. This new point is stored as the *last point*. This option is used for defining slide line endpoints.

D1 **Delete Closest Point And Definitions.** *D1* searches the database for the point closest to the current x, y location. Once found, a box is drawn around this point to

0 Closest *Prefix*	1 Point 0- Closest Point C- Point at Point D- Delete Point	2 Straight Line 0- Closest Line C- Line on Line D- Delete Line	3 Cursor
4 Bisect Line	5 CCW Arc 0- Closest CCW C- CCW Arc ON Arc	6 CW Arc 0- Closest CCW C- CW Arc ON Arc	7 Region
8 Move Point	9 Repaint Screen	A Toggle Snap D- Delete Grid	B Zoom (Box) 0- Previous Zoom D- Reset Zoom
C Slide-Line *Prefix*	D Delete *Prefix*	E Exit	F *Not Used*

Table 1. Keypad Definitions

indicate the point selected, and the user presses *1* again to delete the point and all lines, sides, and regions which are defined using this point. Pressing any other key will abort the point delete command. The items deleted are erased from the screen.

2 **Straight Line To New Point.** *2* enters the current x, y location as a new point, with a new number, just as *1* does, but also enters a straight line into the database drawn from the *last point* to this *new point*. This line is also automatically numbered and displayed on the screen. The *new point* is then stored as the *last point* for further reference. If no *last point* has been stored, no line is input.

02 **Straight Line To Closest Point.** *02* searches the database as *01* does to find the closest point. This point is used as the *new point* for straight line generation as in *2*. If the closest point found is the *last point*, no line is generated. This option allows for closing the region, or for connecting existing points into lines.

C2 **Straight Line To New Point Over Closest Point.** *C2* enters a new point with the closest point's coordinates as *C1* does, and then enters a new line as in *2*. Again this is used for slide line definitions.

D2 **Delete Closest Line And Definitions.** *D2* searches the database for the line closest to the current x, y location. Once found, this line is drawn in another color, and the user presses *2* again to delete this line and all sides and regions which are defined using this line. Pressing any other key will abort the delete line command. The items deleted are erased from the screen.

3 **Show Current Location.** *3* draws a full screen cross-hair to show the current location of the cursor relative to the data drawn on the screen. Since the screen cursor does not automatically track with the puck, pressing the *3* key after moving the puck will update the cursor location on the screen.

4 **Division of Closest Line.** *4* divides the line closest to the current cursor location into two lines at the perpendicular intersection of a line from the cursor to this line. The common end point of the two new lines is located on the original line. The point does not snap to a grid intersection *even if grid lines are on*. The point may be moved to a grid intersection by using the *8* key as explained later.

All line attributes of the original line are inherited by the two new lines including line factor, line type, and boundary flags. If intervals had been specified on the original line, these are divided between the new lines based on relative lengths. Also any regions or sides which used the original line in their definition will be redefined to include the two new lines.

5 **Ccw Arc To New Point.** *5* enters the current x, y location as the *new point* and designates a counterclockwise circular arc to be drawn from the *last point* to this *new point* about a *center point* to be entered next. Upon entering *5*, a *center point* must be entered for the arc definition to be completed.

The center may be entered with either a *1* to designate a new point or a *01* to refer to the closest existing point. This *center point* will be input into the data base without regard to it being a true center or not. FASTQ uses a spiral representation to smooth the arc between the *new point* and the *last point* about the center given.

After the arc has been drawn, the *new point*, and not the *center point*, is stored as the *last point* for further reference. Thus, the line input sequence can continue without re-entering the arc endpoint with a *01*.

05 **Ccw Arc To Closest Point.** *05* finds the closest point to the current location, and uses it as the arc endpoint, or *new point*. No additional points are added to the database, and arc generation continues as in *5*. If the closest point found is the *last point*, no arc is generated.

C5 **Ccw Arc To New Point Over Closest Point.** *C5* enters a new point with the closest point's coordinates, as in *C1*, and draws a counterclockwise arc as in *5*. Again this is used for slideline generation.

6 **Cw Arc To New Point.** *6* designates a clockwise arc as opposed to the counterclockwise arc in *5*.

06 **Cw Arc To Closest Point.** *06* designates a clockwise arc as opposed to the counterclockwise arc in *05*.

C6 **Cw Arc To New Point Over Closest Point.** *C6* designates a clockwise arc as opposed to the counterclockwise arc in *C5*.

7 **Enter Enclosing Region.** *7* defines a region as the set of connecting lines from the database which most tightly enclose the current x, y location of the cursor. If such a definition of lines does not exist in the database, the terminal *beeps* and no region is input. If such a line connectivity can be made, the region will be sequentially numbered and entered into the data base.

Also, a diamond will be displayed on the screen. This diamond will be drawn half way along the diagonal between the maximum x, y location and the minimum x, y location. Depending on the skewness of the region, this may or may not fall within the boundary of the region.

8 **Move Point.** *8* highlights the closest point to the current cursor position and prompts the user for a new position. A second *8* will place the point at the new position while pressing any other key, except the *3* key, will abort the move command.

9 **Repaint Screen.** *9* repaints the screen which may have become corrupted because of multiple point, line or grid line deletions.

A **Toggle SNAP.** *A* toggles the *SNAP* setting. With the *SNAP* on, grid lines are displayed, and all points entered or moved, except those generated by line division, automatically snap to the closest grid intersection.

DA **Delete A Grid Line.** *DA* searches the database for the grid line closest to the current x, y location. Once found, this grid line is drawn in another color, and the user presses *A* again to delete this grid line. Pressing any other key will abort the delete grid line command. The deleted grid line is erased from the screen.

B **Zoom On Area-of-Interest.** *B* selects the current cursor position as the lower-left corner of the zoom area. This corner is drawn on the screen. A second *B* selects the upper-right corner of the zoom area. The screen is repainted to show only the area selected. The previous zoom limits are saved for use by the *0B* command.

0B **Toggle Between Current Zoom and Last Zoom.** *0B* exchanges the current zoom limits for the previous ones and redraws the screen with these new limits. This is an easy way to move between two areas of interest in the model.

DB **Reset Zoom Limits To Default.** *DB* resets the zoom limits to their default values, i.e. those set upon entering the *DIGITIZE* option. The previous zoom limits are saved for use by the *0B* command.

E **Exit.** *E* exits the tablet input routines and returns the user to the main menu.

Keypad Example. A sequence such as *3–01–3–3–05–3–3–1* will:

1. Display the current cursor location,
2. Pick the point closest to the cursor location and save it as the *last point*,

3. Update the cursor location twice,
4. Pick the point closest to the cursor location as the end point of a counterclockise arc from the *last point*,
5. Update the cursor location twice, and
6. Pick the the current cursor location as the center point of the CCW arc.

References

[1] T. D. Blacker, **FASTQ** *Users Manual Version 1.2*, SAND88-1326, Sandia National Laboratories, June 1988.

[2] Unclassified memo to Distribution from T. D. Blacker, 1522; dtd 6/9/88; subject: **FASTQ** Newsletter No. 4.

[3] Unclassified memo to Distribution from T. D. Blacker, 1522; dtd 6/30/88; subject: **FASTQ** Newsletter No. 5.

[4] L. R. Herrmann, "Laplacian-Isoparametric Grid Generation Scheme," *Journal of Engineering Mechanics*, EM5, October 1976, pp. 749-756.

Sandia National Laboratories
Albuquerque, New Mexico 87185

date: June 30, 1988

to: Distribution

Ted Blacker

from: T. D. Blacker, 1522

subject: **FASTQ** Newsletter No. 5

References

[1] T. D. Blacker, **FASTQ** *Users Manual Version 2.1*, SAND88-1326, Sandia National Laboratories, June 1988.

[2] Unclassified memo to Distribution from T. D. Blacker, 1522; dtd 6/9/88; subject: **FASTQ** Newsletter No. 4.

This newsletter is to document continuing changes and updates to **FASTQ**. The following topics will be discussed:

1. Extension of Weighting Boundary Conditions For Closed Boundaries
2. Name Change For *LINEBC* And *SIDEBC*
3. Redimensioning Improvement When Reading Large Files

Copy to:
1265 W. A. Johnson
1412 S. J. Weissman
1412 J. C. Bushnell
1412 Y. T. Lin
1412 J. L. Mitchiner
1412 L. R. Phillips
1510 J. W. Nunziato
1511 D. K. Gartling
1512 J. C. Cummings
1513 D. W. Larson
1513 R. E. Hogan
1520 L. W. Davison
1521 R. D. Kreig & Staff
1522 R. C. Reuter & Staff
1523 J. H. Biffle & Staff
1524 A. K. Miller & Staff
1530 W. Herrmann, Actg.
1550 C. W. Peterson
2541 A.. Gonzales
5165 N. R. Hansen
5165 R. K. Thomas
6314 L. S. Costin
6323 R. H. Yoshimura
6332 T. M. Torres
8241 K. J. Perano

1 Extension of Weighted Boundaries

The weighted boundaries implemented in [2] allow the user to specify nonuniform weights for nodes along a boundary. To insure that the weighting distribution is applied along the boundary in the correct direction, the user specifies a *beginning point*. The specified *beginning point* is weighted with the value of the weighting function corresponding to the minimum x location of the function. Typically this would be the value of the function at zero. From the *beginning point*, the weights are assigned progressively along the boundary. The next node will get the next proportional weight, etc. Thus, the beginning node is intended to be the indicator of directionality along the boundary.

However, if the boundary being loaded is a closed loop, such as an interior hole or the complete boundary of a region. the *beginning point* does not supply enough information to indicate directionality since the node in either direction is a member of the same boundary flag set. Thus, for closed loop boundary weighting, additional information is needed. The user must specify (either by default or explicitly) the line which is attached to the *beginning point*, and whose nodes are to be used first in the boundary weighting. This *beginning line* parameter is optional. For non-closed boundaries, it is not used by FASTQ. The default for closed loop boundaries is the first line in the boundary flag listing. If the boundary flag definition is read from a file, there is no chance of ambiguity. However, there is no guarantee on the order of lines in a boundary flag when interactively changing or inputting the definition. Thus, it is probably good practice to explicitly define the *beginning line* for closed boundaries.

As an aid in verifying that closed loop boundaries have been handled appropriately. **FASTQ** outputs a warning message whenever a closed loop boundary is weighted. The message indicates the flag being weighted, the *beginning point* number the first node number (which is located at the *beginning point*) to be weighted, the *beginning line* number, and the second node number (which is located on the *beginning line*) to be weighted.

The weighting card in the **FASTQ** file and the process for inputting the weighting information in an interactive session have been changed to accomodate this added parameter. Note that the formats also use the new designations of *ELEMENT* and *NODE* boundary flags as discussed in Section 2.

1.1 Weight Card

The weight card has 6 fields:

1 The string WEIGHT to identify the card.

2 The string POINT, NODE, or ELEMENT to indicate which type of boundary flag is to receive a weighting.

3 The boundary flag number which is to receive a weighting.

4 The weighting function. For POINT boundary flags this is a point number. For NODE or ELEMENT boundary flags this is a positive side number or a negative line number (if only one line is being used to define the weighting function.)

5 The beginning distribution point on the boundary flag set.

6 The beginning distribution line on the boundary flag set. This parameter is optional and only used when the boundary flag set contains a closed loop boundary. The default, if not specified, is the first line number in the boundary flag line definition. It has no meaning when used with POINT boundary flags.

Examples:

```
WEIGHT POINT 100 12 5
WEIGHT,NODE,200,8,2
WEIGHT,ELEM,20,-18,14,2
```

1.2 Interactive Weight Specification

Interactive weight specification occurs in the *KEYIN* option of **FASTQ**. If the user chooses the *WEIGHT* suboption of the *KEYIN* option, he/she will be presented with the following prompt:

THE FOLLOWING BOUNDARY FLAGS CAN BE WEIGHTED:
*P*OINT FLAGS - FOR NODES AT POINTS*
*N*ODE FLAGS - FOR NODES ON BOUNDARIES*
*E*LEMENT FLAGS - FOR ELEMENT SIDES ON BOUNDARIES*
WHICH BOUNDARY FLAG WOULD YOU LIKE TO WEIGHT:

If the *ELEMENT* boundary flag option is chosen the user will be presented with the following prompt:

ENTER BOUNDARY ELEMENT FLAG WEIGHTS IN THE FOLLOWING FORMAT:
[FLAG NO., WEIGHTING SIDE (OR NEG LINE) NO., BEGINNING POINT NO., BEGINNING LINE NO. (OPTIONAL)]
HIT RETURN TO END INPUT
>

A response of:

```
> 200,7,3,22
```

specifies that *ELEMENT* flag number 200 will receive weights according to the function specified by lines in side 7, and that the minimum x value in the function will be applied starting at the node generated at point 3 in the boundary flag set and progressing along line 22 if it is a closed loop boundary. Interactively weighting other types of boundary flags proceeds in a similar fashion.

2 Renaming of *LINEBC* and *SIDEBC*

Users of **FASTQ** have requested a name change for the boundary flags. This change better reflects their use. The old names of LINEBC and SIDEBC will still be allowed, but the new names of NODEBC and ELEMBC respectively can now be used synonymously. However, all interactive prompts and listings have been changed to only reflect the new naming convention. Eventually the old names should disappear from **FASTQ** files. The new names better reflect the use of the boundary condition flags, and they are not ambiguous with the *LINE* and *SIDE* definitions used in geometry definition.

3 Improvement of Redimensioning For Large Files

FASTQ must redimension memory storage whenever the data being input exceeds the dimensioned size of the array(s.) In the past, array sizes were increased by 50% each time redimensioning was needed. This incremental redimensioning caused **FASTQ** to iterate several times between an attempted read and redimensioning for very large input files. Since redimensioning necessitated a rereading of the file, this was often a time consuming process. **FASTQ** has been changed so that if input exceeds the dimensions when reading a file, **FASTQ** rewinds the file, scans the file to find out how much data is there, and then performs the redimensioning based on the amount of data. This slows the redimensioning process by the time it takes to scan the file, but no iteration is required for large files. If redimensioning is needed in interactive sessions, the array sizes are still only increased by 50%, as **FASTQ** cannot predict the amount of information the user will input.

TDB:1522:tdb

Sandia National Laboratories
Albuquerque, New Mexico 87185

date: June 9, 1988

to: Distribution

[illegible]

from: T. D. Blacker, 1522

subject: FASTQ Newsletter No. 4

References:

1) T. D. Blacker, FASTQ *Users Manual Version 2.1*,SAND88-1326, Sandia National Laboratories, 1988.

This newsletter is to document continuing changes and updates to FASTQ. The following topics will be discussed:

1. Publishing of the Users Manual
2. Addition of Boundary Weighting Capability For Nonuniform Distributed Boundary Conditions
3. Graphics Options Added To Plot Individual Sides And Lines

1 Users Manual

The FASTQ Users Manual [1] has been completed and is in the process of being published. The users manual documents a number of additions to FASTQ since the last newsletter, and makes obsolete all other FASTQ documentation including previous newsletters. I strongly recommend that users of FASTQ review the manual carefully, as a number of recent requests and additions have been added to FASTQ. These include, but are not limited to the following:

- Multiple command input is now allowed at any level in FASTQ. For instance, at the main level of FASTQ, a command of "M,P,G,P" will choose the *MESH* option, processes the mesh, choose the *MESH GRAPHICS* option, and plot the mesh without any additional user interaction.
- The toggling of specific portions of the geometric information is possible in the *GRAPHICS* option. For instance, a plot with just line intervals or just side boundary flags, etc. is possible. Before you got all or nothing.

- Two new meshing primitives have been added. These are the pentagon and the semicircle. The use of all of the meshing primitives is documented in the Users Manual.
- The NASTRAN and ABAQUS interfaces are more fully supported. The user may specify different element types for different element blocks as these element blocks are being written to the respective files.
- The use of sides has been expanded to allow definition of line sets. These sets of lines (sides) need not be continuous portions of the geometric boundary, and may be used to insure that intervals, factors, or boundary flags are consistent on all lines in the set.

Unfortunately, the users manual will be out-of-date before it even gets published as explained in the rest of this memo. This, however, is as expected, since FASTQ continues to be changed and enhanced. This newsletter should be attached to the manual when it arrives to maintain as complete a reference as possible.

2 Variable Weighted Boundary Flagging

A request to facilitate the loading of structures subjected to loads or boundary conditions of nonuniform distributions has led to the development of weighted boundary flagging in FASTQ. This weighted flagging has been allowed for in the GENESIS database, but until now has not been fully implemented in FASTQ. It involves attaching a scalar weight or scalar value to nodes within a flagged boundary set. These scalar weights can later be translated to actual loads or constraints in the analysis code. Problems involving the application of nonuniform pressures or nonuniform boundary displacements are examples where this capability will be useful.

The implementation in FASTQ allows the user to place weights from an arbitrarily shaped distribution on nodes in either nodal boundary flag sets (*LINEBCs* and *POINBCs*) or element side boundary flag sets (*SIDEBCs*.) Since nodes or element sides may have multiple flags attached to them, they may also have multiple weights attached to them, each weight being associated with a particular flag. The generated weights can also be graphically displayed in the *MESH GRAPHICS* option for user verification. Nodes within boundary flag sets which do not contain explicit weighting functions are defaulted to a weight of 1.0.

The process of applying weights to boundary flags is three fold:

1. The weighting distribution must be described.
2. The distribution must be tied to a particular boundary flag going in a particular direction.
3. The generation of the scalar weights must be verified by the user.

Each of these items will be discussed below.

2.1 Definition of the Weighting Distribution

The weighting distribution is defined by a side (a series of lines) or a single line. These line(s) must form a continuous piecewise function $f(x)$ such that for a side with n lines,

$$f(x) = f_1(x) + f_2(x) + \cdots + f_n(x)$$

where the $f_i(x)$ term represents line i in the definition. Since these lines are actual FASTQ lines, any line type available in FASTQ may be used to define a given portion of the weighting function. Thus, constant or linearly varying functions (straight line types), pieces of exponential spirals (circular arc line types), or the symmetric tip of parabolic functions (parabola line types) may be used to define each $f_i(x)$ term in the completed function. The weighting function has no limits on y value magnitudes, either positive or negative.

The x value of the function is used as a distance measure for mapping against a given boundary flag set. The mapping is accomplished by first determining a given node j's distance along the boundary away from a given starting node. Let d_j be this distance. This distance is determined as the sum of the distances between previous nodes along the boundary. If the distance between neighboring boundary nodes i and $i+1$ along the flagged boundary is given as $s_{i,i+1}$, then

$$d_j = \sum_{i=1}^{j-1} s_{i,i+1}$$

For a flagged portion of a boundary containing n nodes, the total distance is then given as

$$d_t = \sum_{i=1}^{n-1} s_{i,i+1}$$

and the proportional distance p_j is defined as

$$p_j = \frac{d_j}{d_t}$$

The distribution function is then used to determine the weight to be associated with the given proportional distance. For the mapping to be successful, the distribution function must be defined by lines such that any given proportional distance p_j produces only 1 weight or y value. A function $f(x)$ which is monotonically increasing in x would insure this, but such a restriction on the piecewise function $f(x)$ would not allow the user to specify step functions, or to use arcs greater than π or parabolas with a major axis other than vertical in the function's definition. Thus restrictions have been eased somewhat. Let the x value of the unattached end point of the first line in the side be x_1, the x value of the point between line 1 and line 2 be x_2, etc. For a weighting distribution containing n lines, there are $n + 1$ points. For a weighting distribution to be valid the following constraint must hold:

$$x_i \leq x_{i+1}$$

for each i from 1 to n. In other words, the line end points must be monotonically increasing or equal in x. Thus step functions may be used. The function value at a step function where $x_i = x_{i+1}$ is still ambiguous and thus FASTQ defaults to pick a weighting or y value associated with the point with the lowest subscript.

These constraints do not force line segments to be monotonically increasing in x between end points, but the line definitions available in FASTQ afford no real chance at ambiguity. Since circular arcs are limited to less than 2π, for a line i defined by this arc, any x value between the endpoints x_i and x_{i+1} can only contain one y value. This is also true of parabolic lines.

The mapping of the node distance to the function proceeds by determining the proportional x_j value as

$$x_j = [p_j \times (x_{n+1} - x_1)] + x_1$$

From this proportional x_j value, a corresponding unique weight or y_j value is calculated. If x_j falls exactly at a line end point x_i, then y_j is taken as the y value of that end point. As mentioned above, when more than one endpoint exists at that location, the y value of the end point with the lowest subscript at that x location will be used.

2.2 Attaching The Distribution To The Boundary Flag

The distribution is tied to a boundary flag through a new *WEIGHT* card in the FASTQ file, or the *KEYIN WEIGHT* option during an interactive session. The weight definition includes a flag number, a side, line, or point defining the weighting function, and a beginning point on the boundary flag.

The boundary flag may refer to either a *POINT* boundary flag, a *LINE* boundary flag, or a *SIDE* boundary flag. If a *POINT* boundary flag is being weighted, the weighting function is not a side or line, but consists of only one point. The node attached to the designated beginning point will receive the y value associated with the weighting function point as that node's weight. *LINE* and *SIDE* boundary flag weighting functions can be described as either a side or a line. If a line is being used as the weighting function, this line number must be specified as a negative integer.

The beginning point must be a point whose node is flagged with the given flag number. For *LINE* and *SIDE* flags, where a true weighting function is used, the beginning point must be at an end of the boundary flag nodes. Otherwise, the orientation of the boundary flag distribution will be ambiguous. *SIDE* boundary flags refer to element sides, but the weights are attached to the nodes at both ends of these element sides. If any errors are found in the input when weighting is attempted, all weights in the boundary flag set will be reset to the default of 1.0. Again, all boundary nodes not specifically weighted receive a default value of 1.0.

Weight Card. The weight card has 5 fields:

1 The string WEIGHT to identify the card.

2 The string POINT, LINE, or SIDE to indicate which type of boundary flag is to receive a weighting.

3 The boundary flag number which is to receive a weighting.

4 The weighting function. For POINT boundary flags this is a point number. For LINE or SIDE boundary flags this is a positive side number or a negative line number (if only one line is being used to define the weighting function.)

5 The beginning distribution point on the boundary flag set.

Examples:

```
WEIGHT POINT 100 12 5
WEIGHT,LINE,200,8,2
WEIGHT,SIDE,20,-18,14
```

Interactive Weight Specification. Interactive weight specification occurs in the *KEYIN* option of FASTQ. If the user chooses the *WEIGHT* suboption of the *KEYIN* option, he/she will be presented with the following prompt:

```
THE FOLLOWING BOUNDARY FLAGS CAN BE WEIGHTED:
P*OINT FLAGS - FOR NODES AT POINTS
L*INE FLAGS - FOR NODES ON LINES
S*IDE FLAGS - FOR ELEMENT SIDES ON LINES
WHICH BOUNDARY FLAG WOULD YOU LIKE TO WEIGHT:
```

If the *SIDE* boundary flag option is chosen the user will be presented with the following prompt:

```
ENTER BOUNDARY SIDE FLAG WEIGHTS IN THE FOLLOWING FORMAT:
[ FLAG NO., WEIGHTING SIDE (OR NEG LINE) NO., BEGINNING POINT NO.
]
HIT RETURN TO END INPUT
>
```

A response of:

```
> 200,7,3
```

specifies that *SIDE* flag number 200 will receive weights according to the function specified by lines in side 7, and that the minimum x value in the function will be applied starting at the node generated at point 3 in the boundary flag set. Interactively weighting other types of boundary flags proceeds in a similar fashion.

2.3 . Verification of Weighting Values

A toggle has been provided to display the weighting values attached to any nodes in the completed mesh. In the *MESH GRAPHICS* option, specifying the *WEIGHT* suboption will toggle this display from ON to OFF, or OFF to ON depending on the setting currently in effect. Weights for nodal boundary flag sets will be displayed in red, and weights for side boundary flag sets will be displayed in white on a Leir Seigler terminal.

3 Plotting By Side Or Line

The *LPLOT* and *SPLOT* option has been added to the main *GRAPHICS* option of FASTQ. This option allows the user to specify sides or lines for plotting. Any number of sides or lines may be specified, and only those points and line(s) which make up the sides or lines specified will be plotted. These points and lines remain active until specifically changed as in the *RPLOT* and *BPLOT* option explained in [1]. These options are particularly useful when verifying a line or side which is to be used as a boundary set weighting function. The *SPLOT* option is also useful for examination of sets of lines on a side which are to receive the same number of intervals, the same factor, or th: same boundary flags.

TDB:1522:tdb

Distribution:

1265	W. A. Johnson
1412	S. J. Weissman
1412	J. C. Bushnell
1412	Y. T. Lin
1412	J. L. Mitchiner
1412	L. R. Phillips
1510	J. W. Nunziato
1511	D. K. Gartling
1512	J. C. Cummings
1513	D. W. Larson
1513	R. E. Hogan
1520	L. W. Davison
1521	R. D. Kreig & Staff
1522	R. C. Reuter & Staff
1523	J. H. Biffle & Staff
1524	A. K. Miller & Staff
1530	W. Herrmann, Actg.
1550	C. W. Peterson
2541	A. Gonzales
5165	N. R. Hansen
5165	R. K. Thomas
6314	L. S. Costin
6323	R. H. Yoshimura
6332	T. M. Torres
8241	K. J. Perano

SANDIA REPORT
SAND88–1326 • UC–705
Unlimited Release
Reprinted August 1993

FASTQ Users Manual Version 1.2

Ted D. Blacker

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

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

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

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

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

NTIS price codes
Printed copy: A06
Microfiche copy: A01

Using the Paving Algorithm in FASTQ

General

Although the Paving algorithm for filling an arbitrary region with all quadrilateral elements is included in the FASTQ code, the FASTQ documentation has not been updated to reflect this addition. The purpose of this addendum is to document the use of the paving algorithm within the FASTQ code.

Setting the Scheme

Using the Paving option is fairly simple: First, the scheme for the region to be paved should be set to X (see sections 3.1.7 and A.7 of the FASTQ manual). For Example:

```
ENTER OPTION: keyin


ENTER KEYIN OPTION: scheme
 NOTE: ENTER A DEFAULT SCHEME BY SPECIFYING
 THE REGION NUMBER AS ZERO
 ENTER A SCHEME IN THE FOLLOWING FORMAT:
 [ REGION NO., SCHEME ]
 HIT RETURN TO END INPUT
>1 x
>


ENTER KEYIN OPTION:
```

The paving scheme will now be used for region 1.

Interval Assignment

The only major restriction for paving is that each boundary of the region (both exterior and interior) must contain an even number of intervals.. The user may set up intervals for paving in two ways: One is by setting the number of intervals for each line or side of the region to be paved(see sections 3.1.3 and A.3 for some ways to set intervals). If the user makes a mistake and the system detects an odd number of intervals, an interval is automatically added if possible to one of the lines or sides to allow paving to proceed. The second method of setting up intervals for paving is the use of a characteristic size desired for elements in the region. For this method, intervals for lines or sides in the region are set to zero. Then, the mesh command size is used for the region as follows:

```
ENTER OPTION: mesh
ENTER MESH OPTION: size
 NOTE: ENTER A DEFAULT SIZE BY SPECIFYING
 A SIZE WITH NO REGIONS.
 ENTER REGION SIZE DATA IN THE FOLLOWING FORMAT:
 [ SIZE, REGION 1, REGION 2, ..., REGION N ]
 HIT RETURN TO END INPUT
>0.5 1
>


ENTER MESH OPTION:
```

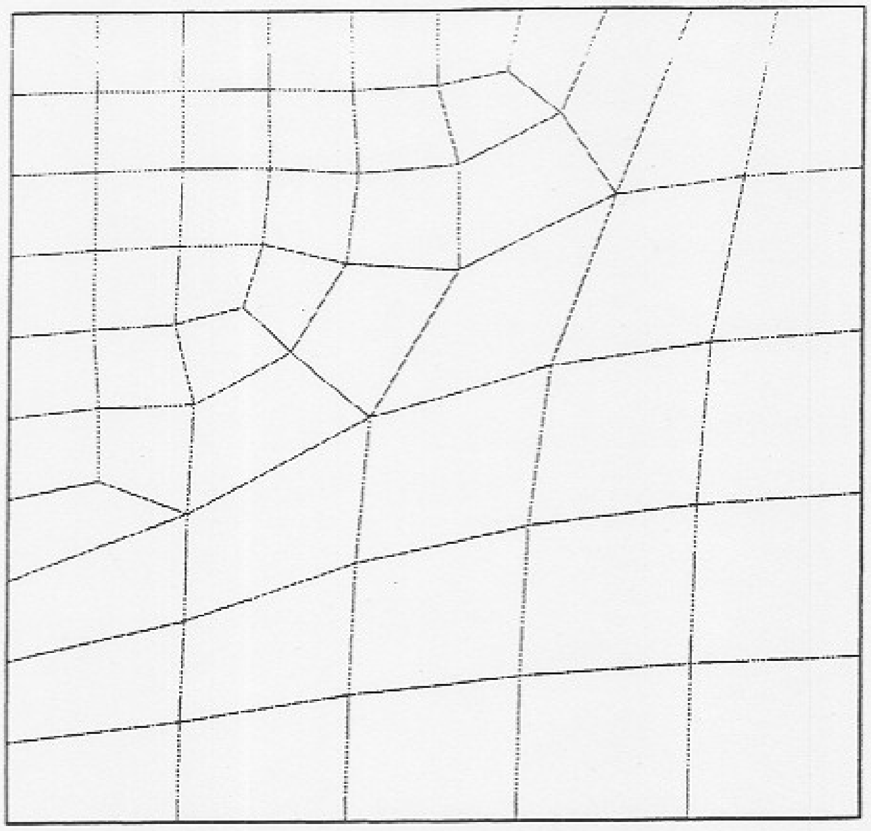
For this example, the mesh element size desired for region 1 was set to 0.5. The system automatically calculates the appropriate intervals for each line or side in the region and performs the meshing.

Meshing of paved areas proceeds as with any other scheme using the Mesh Process or Mesh Step commands as explained in the FASTQ manual.

Sample Problem

Below is a listing of a simple test problem using the paving scheme. The bottom and right lines are assigned and interval of 5. The other two lines are not explicitly assigned and get an interval assignment based on the desired size of 1.0 specified for the region. The finished mesh is also shown.

```
TITLE
 TEST PAVING PROBLEM
 POINT 1 0.0000000E+00 0.0000000E+00
 POINT 2 1.0000000E+01 0.0000000E+00
 POINT 3 1.0000000E+01 1.0000000E+01
 POINT 4 0.0000000E+00 1.0000000E+01
 LINE 1 STR 1 2 0 5 1.0000
 LINE 2 STR 2 3 0 5 1.0000
 LINE 3 STR 3 4 0 0 1.0000
 LINE 4 STR 4 1 0 0 1.0000
 REGION 1 1 -1 -2 -3 -4
 SIZE 1.0000000E+00 1
 SCHEME 1 X
 SCHEME 0 M
 BODY 1
 EXIT
```



Output from Sample Problem

HOW TO OBTAIN HARD COPY OF FASTQ MESH GENERATIONS

If the user simply enters "fastq" from the keyboard, the script which controls the execution of FASTQ executes fastq.x11 to bring the output to the user's monitor window. This is the script's default. However, if the user needs to generate a printed copy during an interactive session, the DUAL version of FASTQ must be active.

To use this version enter fastq with the -dev dual option at the start of a FASTQ session. This allows two simultaneous output possibilities: the X11 window and a hardcopy of the model plotted in the X11 window. The hardcopy will be in the form of a metafile.

The option to obtain hardcopy of plotted models IS on the menus! Unfortunately the menu is currently too long for the window and it tends to be overlooked. (To see it press C to continue the long menu.) Enter h after plotting the model. The hardcopy is put into a metafile and by default the file will be fastq.met. To use another filename, enter fastq with two options, -dev dual and -hard afilename.met, at the start of the session.

After exiting FASTQ, the afilename.met file can be changed to a postscript file by executing the POST utility. The post utility is part of the SVDI installation, and it also is run from a script file. To generate an postscript file use the cps option with post. Post will present a choice of output types. The resulting vdicps.ps file can then be printed by a postscript printer.

Other options such as eps can be used in place of cps. In particular, eps can generate an encapsulated postscript file to be inserted into a document by applications such as Framemaker.

FASTQ can be run in batch mode, or interactively, by replacing the -dev dual option with -dev cps or other -dev parameters. Obtaining hardcopy is still an explicit command. Usually there will be an input file of mesh generation instructions for these -dev options because X11 graphics are not active and no mesh generation output will appear in a screen window.

SUMMARY:

```
fastq -dev dual -hard afilename.met
        (choose hardcopy during the session, more than one plot is OK)
post afilename.met cps
        (choose output type)
lp vdicps.ps
```