2 GETTING STARTED

This section provides the first-time user with an introduction to *FLAC*. *Getting Started* contains instructions for program installation and start-up on your computer. It also outlines the recommended procedure for applying *FLAC* to problems in geo-engineering and includes simple examples that demonstrate each step of this procedure.

If you are familiar with the program but only use it occasionally, you may find this section (in particular, Section 2.6) helpful in refreshing your memory on the mechanics of running *FLAC*. More-complete information on problem solving is provided in Section 3.

FLAC can be operated in *command-driven* or graphical, *menu-driven* mode. For most of the examples in this manual, input is entered and results are viewed using the command-driven mode. We believe this is the clearest way for you to understand the operating procedures for *FLAC*. As explained previously in Section 1.1, the command-driven structure allows *FLAC* to be a very versatile tool for use in engineering analysis. However, this structure can present difficulties for new, or occasional, users. Command lines must be entered as input to *FLAC*, either interactively via the keyboard or from a remote data file, in order for the code to operate. There are over 40 main *commands* and nearly 400 command modifiers (called *keywords*) which are recognized by *FLAC*.

The menu-driven mode is an easy-to-use alternative to the command-driven procedure. All the commands in *FLAC* can be accessed by point-and-click operation from the graphical mode. We call this mode the "*GIIC*" for *Graphical Interface for Itasca Codes*; eventually the *GIIC* will operate with all Itasca software.

Getting Started contains the following information.

1. A step-by-step procedure to install and start up *FLAC* on your computer is given in Section 2.1.

2. This is followed in Section 2.2 by instructions on running *FLAC*. Section 2.2.1 describes the procedure for running *FLAC* in the command-driven mode, and includes installation tests (Section 2.2.1.1) and a tutorial (Section 2.2.1.2) to help you become familiar with common input commands. Section 2.2.2 introduces the *GIIC* and provides a tutorial on running *FLAC* in menu-driven mode (Section 2.2.2.4).

3. There are a few things that you will need to know before creating and running your own *FLAC* model — i.e., you need to know the *FLAC* terminology. The nomenclature used for this program is described in Section 2.3. The definition of a *FLAC* finite difference grid is given in Section 2.4. You should also know the syntax for the *FLAC* input language when running in command-driven mode; an overview is provided in Section 2.5.

4. The mechanics of running a *FLAC* model are described in separate steps; in Section 2.6, each step is discussed separately and simple examples are provided.

5. The sign conventions, systems of units and precision limits used in the program appear in Sections 2.7, 2.8 and 2.9, respectively.

6. The different types of files used and created by *FLAC* are described in Section 2.10.

2.1 Installation and Start-up Procedures

2.1.1 System Requirements

To install and operate *FLAC*, your computer must meet the following minimum requirements.

Hard Drive — At least 35 MB of hard disk space must be available to install *FLAC* with the *GIIC*. In addition, a minimum of 100 MB disk space should be available for model save files.

RAM — The minimum amount of RAM required to load *FLAC* with the *GIIC* is 40 MB. Of this memory, approximately 22 MB are used for the Java(TM) Runtime Environment (JRE) to run the *GIIC*, 6 MB for the *GIIC* class files, and 12 MB for the *FLAC* executable code and dynamic linked libraries (DLLs). The executable code loads with 8 MB allocated by default for model generation. The memory allocated for a *FLAC* model can be adjusted by the user to increase the number of zones (size of model) to be analyzed (see Section 2.1.4).

Generally, the combined RAM needed by *FLAC* and its model storage should leave 4 to 6 MB available to Windows; otherwise, Windows starts swapping into virtual RAM (on disc) — this swapping causes a dramatic performance loss in *FLAC*. The more applications running simultaneously, the smaller the *FLAC* model should be. For fast operation of typical geo-engineering models, it is recommended that the computer have at least 128 MB RAM. The operation of the *GIIC* will be noticeably sluggish if the computer has only 64 MB RAM.

Display — For best performance, a screen resolution of 1024×768 pixels and a 16-bit color palette is recommended.

Operating System — *FLAC* is a 32-bit native Windows application. Any Intel-based computer capable of running Windows 95 and upward is suitable for operation of *FLAC*. The code will not run on 16-bit systems such as Win 3.x. Also, computers based on the DEC Alpha Chip are not supported by Itasca and may not execute *FLAC* properly.

Output Device — By default, plots from *FLAC* are sent directly to the Windows native printer. Plots can also be directed to the Windows clipboard, or files encoded in PostScript, Enhanced Metafile format, and several bitmap formats (PCX, BMP, or JPEG). See the **SET plot** command for the selections of output format.

Operation on PC Networks — *FLAC* can be installed on a server for operation over a PC network. However, a hardware lock must be installed locally on any computer running *FLAC*.

2.1.2 Installation Procedure

The *FLAC* package is installed in Windows from a CD-ROM using standard Windows procedures.*

The CD-ROM also contains the complete *FLAC* manual and Acrobat Reader for viewing. The online manual can be accessed directly from the CD-ROM, or it can be copied to another location. The manual requires approximately 30 MB of disk space.

Insert the CD-ROM into the appropriate drive. The installation procedure will begin automatically.

When installing *FLAC*, the install program (“SETUP.EXE”) will activate and guide you through the installation. Make your selections in the dialogs that follow. Please note that the CD-ROM contains all of Itasca’s software products. You *must* click on the *FLAC* box in the *Select Components* dialog in order to install *FLAC* on your computer. The online manual will also be copied to your computer during the installation. You have the option to avoid copying the manual in the *Select Components* dialog. The *FLAC* package can be uninstalled via the Add/Remove Programs icon in the Windows Control Panel.

A default directory structure will be created when using the install program. The root directory is “\ITASCA”; the sub-directories and their contents are summarized in Table 2.1. You will find that all the references made in the *FLAC* manual to files use the default directory structure, and all data files that are described in the manual are contained in these directories.

To use the online manual, double-click on the file “CONTENTS.PDF” that is located in the directory “ITASCA\MANUALS\FLAC.” (If you copy the online manual directly from the CD-ROM, be sure to include the index sub-directory.)

Finally, be sure to connect the *FLAC* hardware key to your LPT1 port before beginning operation of the code.

---

* Note that the installation program works *only* under 32-bit Windows. The term “Windows” in this section means Windows 95 and upward. The files are in uncompressed form on the CD, so you can also copy them directly. However, this procedure will not install the hardware key drivers that are required for execution.

Table 2.1 Contents of Itasca directories

Directory	Sub-directory		Section Files
Fishtank			general *FISH* function files (in *FISH* Functions Library)
FLAC			Single- and Double-Precision executable codes
	Backgrnd		Theory and Background — data files
		Grid	Grid Generation
		Intface	Interfaces
		Models	Constitutive Models: Theory and Use
		Theory	Background — The Explicit Finite Difference Method
	FISH		*FISH* in *FLAC* — *FISH* functions and data files
		FIN	Program Guide
		Library	Library of *FISH* Functions
		Tutorial	*FISH* Beginner’s Guide and *FISH* Reference
	Fluid		Fluid-Mechanical Interaction — data files
		TwoPhase	Two-Phase Flow Option
	GUI		Graphical User Interface — JAVA class files
	Options		Optional Features — data files
		Creep	Creep Material Models
		Dynamic	Dynamic Analysis
		Thermal	Thermal Option
	Problems		Verification Problems and Example Applications — data files
		Examples	Example Applications
		Verify	Verification Problems
	Struct		Structural Elements — data files
	Tutorial		User’s Guide — data files
		Beginner	Getting Started
		FISH	*FISH* Beginner’s Guide
		Solving	Problem Solving
JRE			JAVA runtime environment
Manuals			FLAC online manual
Models			Constitutive model DLLs
System			“FLAC.INI” files, hardware key drivers
Utility			README files, movie viewer

2.1.3 Components of FLAC

Two versions of *FLAC* are provided: a single-precision version, "FLACW_SP.EXE," and a double-precision version, "FLACW_DP.EXE." These executable codes are stored in the "\ITASCA\FLAC" directory. All examples in this manual are run with the single-precision version. The double-precision version is provided for models in which single-precision calculations may not be adequate (see Section 2.9). The single-precision version is recommended for most common analyses; the double-precision version runs approximately 2 times slower, and requires approximately 3 times more RAM, than the single-precision version for similar sized models. Each version can be accessed from the *FLAC* icon in the "Itasca Codes" group.

Both the single- and double-precision executable codes are described as *Windows-console* applications because they operate in a text mode in Windows. Both codes communicate with the *GIIC* via the JAVA(TM) Runtime Environment. The user can switch from the text (command-driven) mode to graphics mode by typing

```
giic
```

from the command line in text mode, and by pressing the FILE / EXIT button in graphics mode to return to text mode.

The Windows-console version of *FLAC* is compiled with the Watcom Fortran 11.0 compiler. The *GIIC* is written in JAVA using JAVA(TM) Runtime Environment, standard edition version 1.2.2.

2.1.4 Memory Allocation

Automatic memory allocation logic has been implemented in *FLAC* for Intel-based computers. When loaded, *FLAC* will, by default, adjust the size of the main array to take up 8 MB RAM, or the maximum amount available, if it is less than 8 MB. This means that if other programs are resident when *FLAC* is executed, the size of the main array may be decreased and smaller allowable problem sizes will result.

You can change the amount of memory used by *FLAC* by modifying the shortcut to *FLAC* from the *FLAC v4.0* icon in the *Itasca Codes* group. In the shortcut properties, add the amount of memory (in MB) to the end of the target string. If the amount of memory requested is more than that available, *FLAC* will still load, but with the maximum available memory. The amount of memory allocated for *FLAC* is printed in the start-up screen.

As a guide, Table 2.2 summarizes the approximate maximum numbers of zones (of Mohr-Coulomb material) that can be created for different sizes of available RAM, in the single-precision version of *FLAC*.

Table 2.2 Maximum number of elements in available RAM

Available RAM (MB)	Maximum number of zones (single-precision)
8	30,000
16	60,000
32	120,000
64	240,000

2.1.5 Utility Software and Graphics Devices

Several types of utility software and graphics devices are available that can be of great help while operating *FLAC*.

Editors — A text editor is used to create *FLAC* input data files. Any text editor that produces standard ASCII text files may be used. Care must be taken if more-"advanced" word processing software (e.g., WordPerfect, Word) is used: this software typically encodes format descriptions into the standard output format; these descriptions are not recognized by *FLAC* and will cause an error. *FLAC* input files must be in standard ASCII format.

When running *FLAC* from the *GIIC*, an input data file is created automatically as the model is generated in the graphical mode. This data file can be saved and edited in order to reproduce or modify the model in later analyses.

Screen Capture — Graphics software can assist in the production/presentation of *FLAC* results. *FLAC*'s **MOVIE** option allows graphics images to be stored and later displayed in series. A movie viewer is contained in the "\ITASCA\Utility" directory.

2.1.6 Start-up

The default installation procedure creates an "Itasca Codes" group with icons for the single-precision and double-precision versions of *FLAC*. An environment variable pointing at the "ITASCA" directory is created, and the necessary drivers for the hardware key are installed — be sure that the *FLAC* hardware key is attached to the LPT1 port on your computer.

To load *FLAC*, simply click the appropriate icon in the Itasca Codes group. The first time you load *FLAC* you will be asked to specify a customer title. This title will appear on all hardcopy output plots generated from *FLAC*. The customer title can be modified at any time using the commands **SET cust1** and **SET cust2**.

FLAC will start up in command-driven mode, and then immediately switch to the graphics mode. The graphics mode is initiated from the "FLAC.INI" file — see below. The graphics mode may take several seconds to initialize while the JRE is being loaded to run the *GIIC*. The initialization time can be affected by other programs running in the background. If you notice a significant delay in the initialization of the graphics mode, it may be necessary to close other Windows applications.

2.1.7 Program Initialization

On start-up, *FLAC* will look for the file named “FLAC.INI” in the current directory and then, if not found, in the directory pointed to by the ITASCA environment variable. (By default, this is the “\ITASCA\System” directory.) The “FLAC.INI” file can contain any *FLAC* commands that preset attributes of the program that you may wish to apply every time *FLAC* is used. When *FLAC* is installed from the CD-ROM, the “FLAC.INI” file is automatically installed in the “\ITASCA\FLAC” directory and contains the command **GIIC**. This will cause the *GIIC* to start up each time *FLAC* is loaded.

If the file “FLAC.INI” does not exist, *FLAC* simply continues in command-driven mode without error. Note that some commands in a “FLAC.INI” file may result in an error message. For example, if you attempt to give properties to a grid prior to defining the grid, the normal error message will arise.

2.1.8 Version Identification

The version number of *FLAC* follows a simple numbering system that identifies the level of updates in the program. There are three numerical identifiers in the version number — that is,

```
Version I.JK
```

where

`I` is an integer starting with 1 that identifies a major release of the code;

`J` is an integer that is incremented whenever a modification is made that requires a major change to the code structure for a supplemental upgrade release of *FLAC*; and

`K` is an integer that is incremented when minor modifications are officially released as an update to the current version.

In addition to the version number, *sub-version* numbers are also used to identify minor changes to *FLAC* that have been made since the official version was released. Users may access the latest sub-version of the current version of *FLAC* via the Internet. (Contact Itasca for further information.) However, *FLAC* with a sub-version number greater than that of the officially-released version should be used with caution, because not all features have been fully tested.

By typing the command

```
print version
```

at the `flac:` command-line prompt, the complete version number, including the sub-version number, can be obtained.

2.2 Running *FLAC*

FLAC can be run in command-driven mode or in menu-driven mode. We recommend that you first use the command-driven mode to run the installation tests and to become familiar with the procedure for creating a model with input commands. This procedure is described below, and a simple tutorial in command-driven mode is given in Section 2.2.1.2.

If you wish to test the graphical interface first, turn to Section 2.2.2. The *GIIC* is introduced in this section, and a menu-driven tutorial is provided in Section 2.2.2.4.

2.2.1 Running FLAC in Command-Driven Mode

FLAC can be run interactively or from an input data file in command-driven mode. If you wish to run the code interactively, just begin typing in commands at the `flac:` prompt. *FLAC* will execute each command as the <ENTER> key is pressed. If an error arises, an error message will be written to the screen.

As an alternative, an input data file may be created using a text editor (see Section 2.1.5). This file contains a set of commands just as they would be entered in the interactive mode. Although the data file may have any name, a common identifying extension (e.g., ".DAT") will help to distinguish it from other *FLAC* files (see Section 2.10).

The data file can be read into *FLAC* by typing the command

```
call  file.dat
```

on the command line, in which "FILE.DAT" is the user-assigned name for the data file. You will see the data entries scroll up the screen as *FLAC* reads each line (if **SET echo** is **on**).

2.2.1.1 Installation Tests

Three simple data files, "TEST1.DAT," "TEST2.DAT" and "TEST3.DAT," are included in the "\Itasca\FLAC" directory so that you can verify whether *FLAC* is properly installed on your computer. These files test the calculation kernel, the graphics screen plotting, and the hardcopy plotting facilities for your computer. In order to run the third test, your current default Windows printer must be connected to the LPT1 port.*

To run these tests, first start up *FLAC* following the procedure in Section 2.1.6. When you start up *FLAC* after installation from the CD-ROM, the code will be operating in the *GIIC*. To change to command-driven mode, press the OK button to close the *FLAC Options* dialog, and then press FILE/EXIT GIIC; *FLAC* will switch to text mode.

* The *FLAC* hardware key is transparent to other uses of the LPT1 port. The printer can be attached directly to the key.

At the `flac:` prompt, type

```
call test1.dat
```

and press <ENTER>. Several data entries should scroll up the screen, and a simple model will be executed for 100 calculation steps. Example 2.1 contains the results of a successful "TEST1.DAT" run.

Example 2.1 FLAC output from "TEST1.DAT"

```
    major stress     (multiply values below by 10^4)

     I  1     2     3     4
J
 1 -9.553-9.493-9.308-8.834
```

Now enter the command

```
call test2.dat
```

A screen plot of this model should appear. The plot is a contour plot of displacements. The plot window may be moved by dragging the title bar, and it may be resized by dragging a border or corner. It may also be maximized, restored or closed. Only one plot screen can be created at a time, and the plot screen *must* be closed in order to continue processing *FLAC* commands. Press the <ENTER> key to close the plot window and return to the `flac:` prompt.

If your current default Windows printer is connected to the LPT1 port, type

```
call test3.dat
```

and the plot shown in Figure 2.1 should be sent to your printer. If you do not have a printer connected, type

```
quit
```

to stop the installation testing.

If you are not able to reproduce the results of any or all of these three tests, you should review the system requirements and installation steps in Sections 2.1.1 and 2.1.2. If you are still having difficulty, we recommend that you contact Itasca and describe the problem you have encountered and the type of computer you are using (see Section 5.2 for error-reporting procedures).

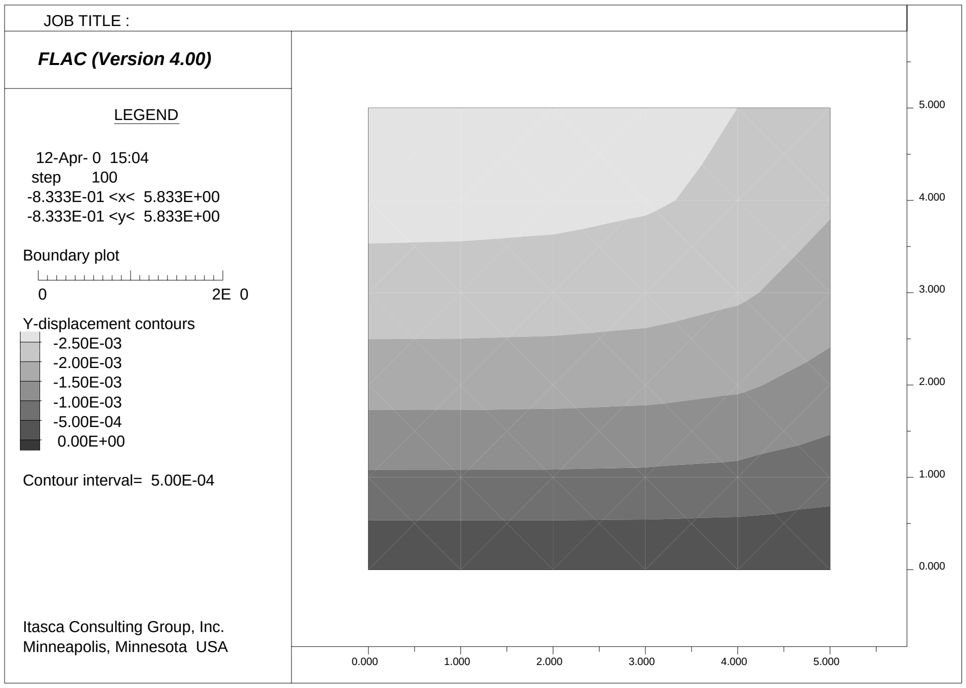


Figure 2.1 Windows plot from "TEST3.DAT"

2.2.1.2 A Simple Tutorial — Use of Common Commands

This section is provided for the new user who wishes to begin experimenting with *FLAC* operation in command-driven mode. A simple example is presented to help you learn some of the basic aspects of solving problems with *FLAC*.

The example problem is a 1 m wide trench excavated to a depth of 3 m in a soil mass. For this tutorial, we excavate the entire trench instantaneously and monitor the resulting movement of the material around the trench. (The data file "TUT.DAT," included in the "\FLAC\Tutorial\Beginner" directory, contains all the commands we are about to enter interactively.)

We run this problem interactively (i.e., by typing the commands from the keyboard, pressing <ENTER> at the end of each command line, and seeing the results directly). To begin, load *FLAC* following the procedure in Section 2.1.6. When you start up *FLAC* after installation from the CD-ROM, the code will be operating in the *GIIC*. To change to command-driven mode, press the OK button to close the *FLAC Options* dialog, and then press FILE/EXIT GIIC; *FLAC* will switch to text mode. Commands are entered at the `flac:` prompt.

To set up the initial finite difference grid, use the **GRID** command:*

```
grid 5,5
```

This command will create an initial grid (or mesh, if you prefer) that is 5 zones (or elements) wide by 5 zones high. Now, give the zones a material model and properties. For this example, we use the Mohr-Coulomb elasto-plastic model. Type in the following commands:

```
model mohr
prop  bulk=1e8 shear=.3e8 fric=35
prop  dens=1000  coh=1e10  ten=1e10
```

Here, we have specified the Mohr-Coulomb model. Every zone in the grid could conceivably have a different material model and property. However, by not specifying a range of zones directly behind the **MODEL** command, *FLAC* assumes that all zones are to be Mohr-Coulomb. The properties are given next — including the bulk modulus (in Pa), shear modulus, the angle of internal friction, the mass density, the cohesion and the tensile strength. Any consistent set of engineering units can be used when assigning properties in a *FLAC* model (see Section 2.8). Note that very high cohesion and tensile strength values are given. These are only initial values that are used during the development of gravitational stresses within the body. In effect, we are forcing the body to behave elastically during the initial development of the gravitational stresses.† This avoids any plastic yield during this initial phase of the analysis. The reasons for this will become obvious once you gain experience with the explicit-simulation procedure.

Now that a grid and model properties have been defined, data pertaining to the simulation can be plotted or printed. Issue the following command:

```
print  x  y
```

The x- and y-coordinates will appear in tabular form in the physical positions of the gridpoints. You will note that the table has i (column) and j (row) going from 1 to 6 along the top and left-hand edge of the table. Therefore, each gridpoint and zone has an i (column) and j (row) associated with it. In this example, the gridpoint range is ***i*** from 1 to 6 and ***j*** from 1 to 6, whereas the zones range is from 1 to 5 for i and 1 to 5 for j. If you require greater clarification on this point, see Figure 2.29 in Section 2.4. To see a plot of the grid, give the following command:‡

```
plot  grid
```

This will create a plot of the grid on the screen. After viewing, press <ENTER> to get back to the `flac:` prompt.

* See the command reference list in Section 1.3 in the **Command Reference** for further details. Note that command words can be abbreviated (see Section 2.5).

† Alternatively, an elastic model could initially be used to set up the virgin stresses, followed by changing the model to Mohr-Coulomb prior to any excavation, applied loads, or other simulations.

‡ The plotting window will be set automatically unless otherwise specified by using the **WINDOW** command.

In order to make a hard copy of a plot, enter the command **COPY** and the plot will be sent (by default) to the current Windows printer connected to the LPT1: port.*

Alternatively, we can send the plot to a file for printing at some later time. For example, the commands

```
set plot emf
copy grid.emf
```

will create a Windows-enhanced metafile plot "GRID.EMF" of the last-viewed plot. The file can then be directly imported to a word processor program such as Microsoft Word.

If a PCX file is desired instead, the **SET pcx** command will allow PCX files to be generated by pressing <`F2`> when the plot is displayed on screen. See Section 1.3 in the **Command Reference** for a full description of this command.

Note that if we do not assign coordinates to the grid (by using the **GENERATE** or **INITIAL** command), then the x- and y-coordinates are assigned equal to the number of the gridpoint minus 1. For example, in the previous grid plot, the lower left-hand gridpoint is assumed to be the origin and is given the coordinate (0,0). The bottom right-hand corner gridpoint (6,1) is given the coordinate (5,0). The user is completely free to assign any chosen coordinates by using the **GENERATE** and **INITIAL** commands. To keep this example simple, we leave the grid at 5 m $\times$ 5 m.

Next, the boundary conditions for the problem are set. In this problem, we want to place roller boundaries on the bottom and sides, apply gravitational forces to the zones, and allow the in-situ stresses to develop as they occur in nature. To fix these boundaries (i.e., no displacement or velocity in the specified direction), use the following commands:

```
fix  y  j=1
fix  x  i=1
fix  x  i=6
```

The commands noted above perform the following functions:

1. The bottom boundary gridpoints (***j*** = 1) are fixed in the y-direction. When *FLAC* sees (***j*** = 1), it automatically assumes that ***i*** ranges from 1 to 6 (i.e., the full range). You can perform the same function by specifying ***j*** = 1, ***i*** = 1,6.
2. The left-hand boundary gridpoints (***i*** = 1) and right-hand boundary gridpoints (***i*** = 6) are fixed in the x-direction. Again, *FLAC* assumes the full range of the j-direction.

* The printer type can be changed with the **SET plot** command; for example, type **SET plot post** before entering the **COPY** command to direct plots to a PostScript-compatible printer. The output port can be changed or a filename can be specified with the **SET output** command — see Section 1 in the **Command Reference**.

Next, we set the gravity by typing

```
set  grav=9.81
```

where 9.81 m/sec^2 is the acceleration due to gravity. Here, gravity is taken as positive downward and negative upward. (If gravity is set negative, objects will rise!)

We wish to see a history of the displacement of a gridpoint on the model to indicate equilibrium or collapse. Type:

```
his nstep=5
his ydis i = 2 j = 6
```

Here, we choose to monitor the y-displacement every five timesteps for a point at the top of the ground surface. Now, we are ready to bring the initial model to equilibrium. Because *FLAC* is an explicit dynamic code, we step the model through time,* allowing the kinetic energy of the mesh to damp out (thus providing the static solution we seek). To allow gravity to develop within the body, we timestep the simulation to equilibrium. Here, the **SOLVE** command is used to detect equilibrium automatically.

Type the commands

```
set force=100
solve
```

The calculation process will begin and the timestep number, maximum unbalanced force and equilibrium ratio will be displayed on the screen. When the unbalanced force falls below the limiting value (a limiting force of 100 N is specified with the **SET** command), the run will stop. Other options for solution limits, such as equilibrium ratio, are discussed later in Section 2.6.4.

Now, we can see what has occurred within the model. Examine the y-displacement history requested earlier:

```
plot his 1
```

A screen plot will be shown which indicates that the model came to equilibrium within 108 timesteps. The final y-displacement at equilibrium is -0.881 $\times$ 10^{-3} m, due to the gravitational loading. For a screen listing of this history, type

```
hist dump 1
```

* calculation time — *not* real time

Let's examine the gravitational stresses developed in the body. The window was automatically defined but, if we wish to enlarge or shrink the plot, we can reset it with the **WINDOW** command. Now, give the plot a title* by typing

```
title
a simple trench excavation example
```

Then type the following:

```
sclin 1  (1,0)  (1,5)
plot  hold  syy  yel  bou  gre
```

This will create a plot (Figure 2.2) of the σ_{yy}-stresses in yellow-brown and the boundary in green.† Similarly, the σ_{xx}-stresses can be plotted by typing

```
plot  hold  sxx  yel  bou  gre
```

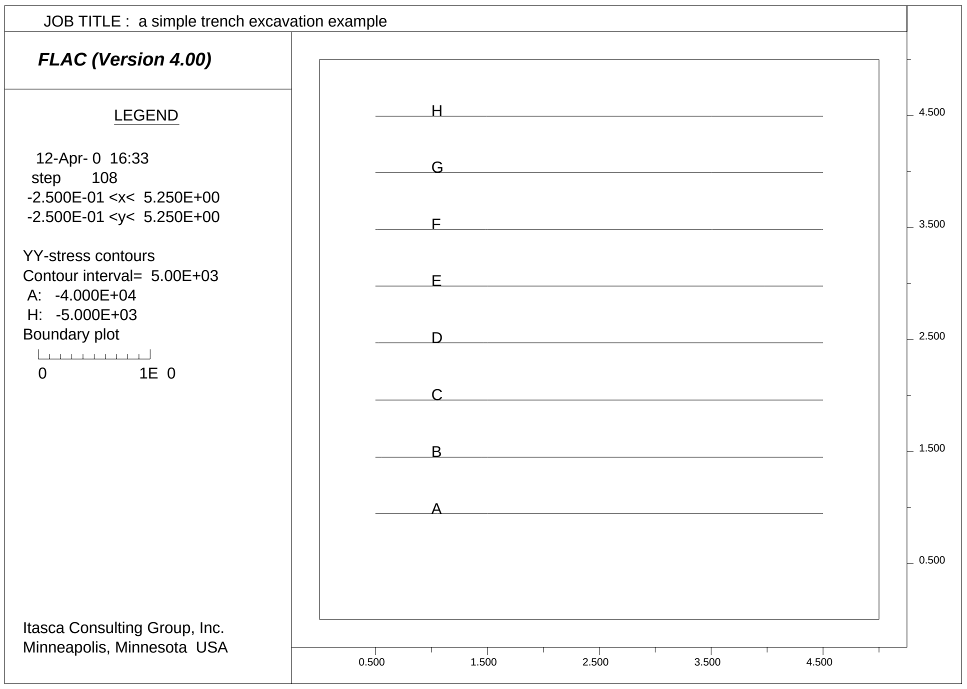


Figure 2.2 The gravitational stresses included in the soil after 108 timesteps

---

* The title and legend appear on hardcopy plots as well as screen plots. There are slight differences between the legend shown on the screen plot and that shown on hardcopy plots.

† The color switch also controls the line style on hardcopy plots. See Table 1.6 in the **Command Reference** to select line styles based on color keywords. Note that all plots shown in the figures in Section 2 have a line style set by the default line style, which is a solid line.

We note that the gravitational stresses increase linearly with depth. The values can be printed by typing

```
print  sxx  syy
```

It is wise to save this initial state so that we can restart it at any time for performing parameter studies. To save this, type

```
save  trench.sav
```

A save file will be created on the default drive. The *FLAC* prompt will then return.

Now we can excavate a trench in the soil. First, type

```
prop coh=0
```

With a zero cohesion and vertical, unsupported trench walls, collapse will certainly occur. Because we want to examine this process realistically, the large-strain logic must be set in the code. This is done by typing

```
set large
```

Finally, for plotting purposes, we wish to see only the change in displacements from the trench excavation and not the previous gravitational setting — so we can zero out the *x*- and *y*-displacement components:*

```
init  xdis=0  ydis=0
```

To excavate the trench, enter

```
model  null  i=3  j=3,5
```

Since we purposely set the cohesion low enough to result in failure, we do not want to use the **SOLVE** command with a limit for out-of-balance forces (which checks for equilibrium). Our simulation will never converge to the equilibrium state. Instead, we can step through the simulation process one step at a time and plot and print the results of the collapse as it occurs. This is the real power of the explicit method. The model is not required to converge to equilibrium at each calculation cycle, because we never have to solve a set of linear algebraic equations simultaneously, as is the case in the implicit codes with which many engineers are familiar.

In *FLAC*, we use the **STEP** command:

```
step 100
```

FLAC will now step through 100 timesteps. When it is finished, the prompt will reappear. Now, examine the results thus far by plotting some variables — e.g.,

```
plot  hold  plastic  boundary
```

* This will not affect the calculations since the model does not require displacements in the calculation sequence. They are kept only as a convenience to the user.

The present state of each zone will be indicated by symbols which represent the type of failure condition. This plot indicates that the zones adjacent to the trench are actively yielding in shear.* The Mohr-Coulomb failure model is discussed in detail in Section 2.4.2 in **Theory and Background**.

Now, try plotting some parameters:

```
plot  hold  grid
```

We notice some grid distortion beginning at the trench.

Next, try some plot overlays to distinguish the failure area (to identify this plot, we could first retitle the plot using the **TITLE** command):

```
plot hold xv z yell int=5e-6 dis red max=1e-2 bou green
```

This will produce a plot of the x-velocity contours (in yellow, contour interval of 5×10^{-6} m, zero contours removed) overlaid by the displacement vectors (in red, scaled to a maximum vector length of 1×10^{-2} m) and the boundary (in green). This is shown in Figure 2.3. The velocity contours are given here to help visualize those areas of active yield, because this material is flowing.

The collapse process can be examined as it occurs by timestepping 100 steps at a time. We encourage you to step ahead in this fashion, creating plots at each stage and experimenting with the **max**, **int** and color keywords at each stage. Try plotting the stresses, velocities and displacements to produce meaningful results. In this example, we will jump ahead to a convenient spot in the collapse process:

```
step 400
```

Again, try

```
plot hold grid
```

There is a drastically different picture at this stage as the trench collapses (Figure 2.4).

By typing

```
plot  hold  plas  bou
```

we note that the zones are still at the yield failure point. Examine the σ_{yy}-state and displacements by requesting

```
plot hold syy zero int=2500 disp max=0.2 mage bou gree
```

---

* Note that we have made our boundaries on this problem small in order to speed operation; thus, some boundary interference occurs.

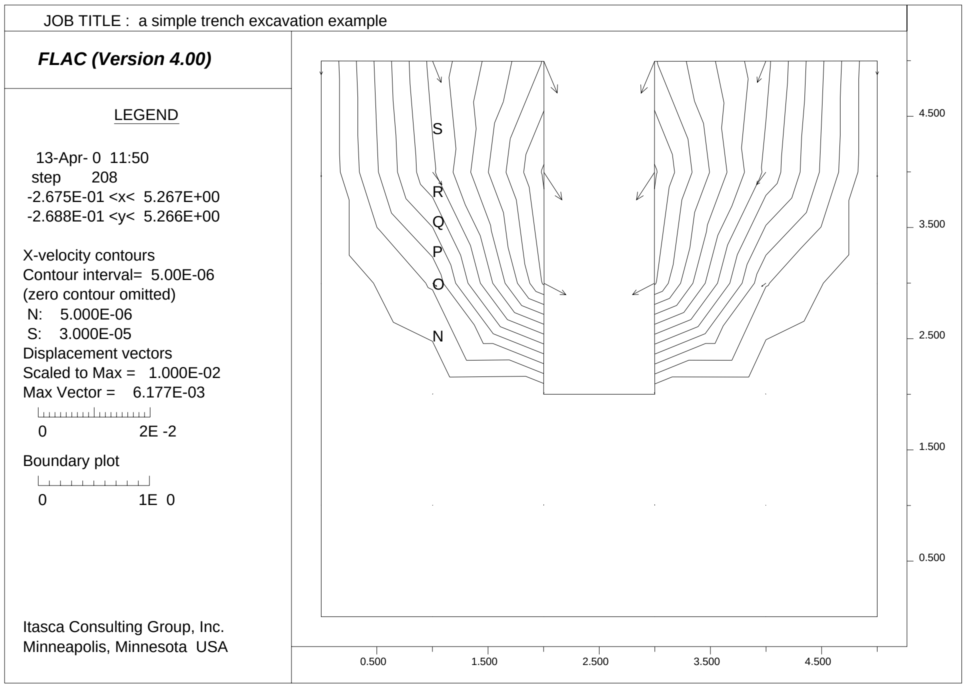


Figure 2.3 A plot of the displacement vectors and x-velocity at timestep 208

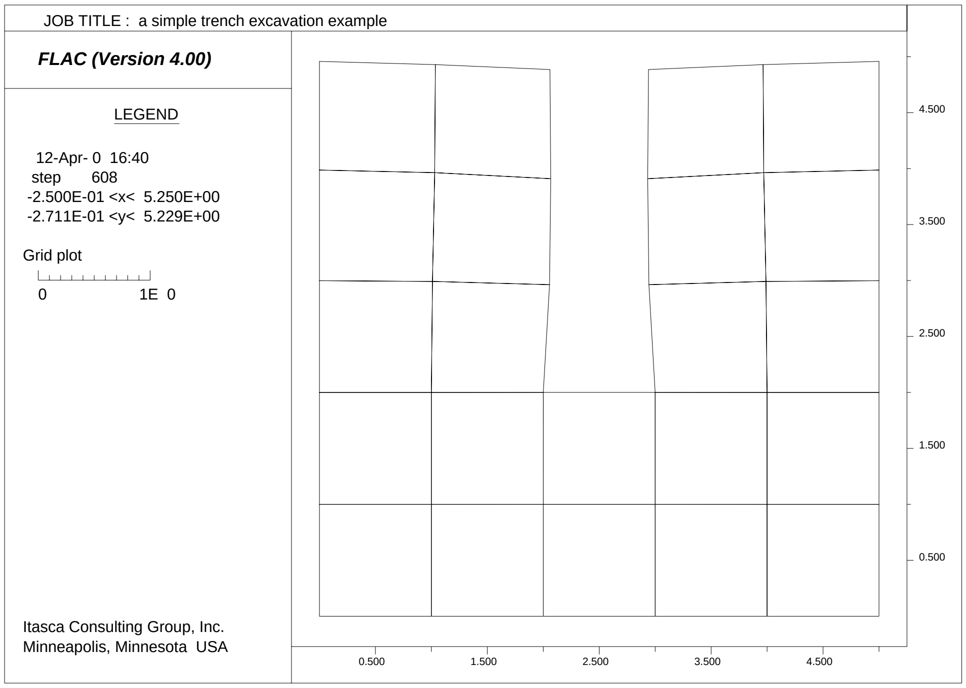


Figure 2.4 Deformed mesh after 608 timesteps

We observe distortion of the stress contours due to the excavation and an increase in magnitude (by approximately 100 times) of the displacement vectors (Figure 2.5). Also note that stress contours, unlike displacement and velocity contours, are not plotted to the external and excavation boundaries, because stresses are constant within a zone.*

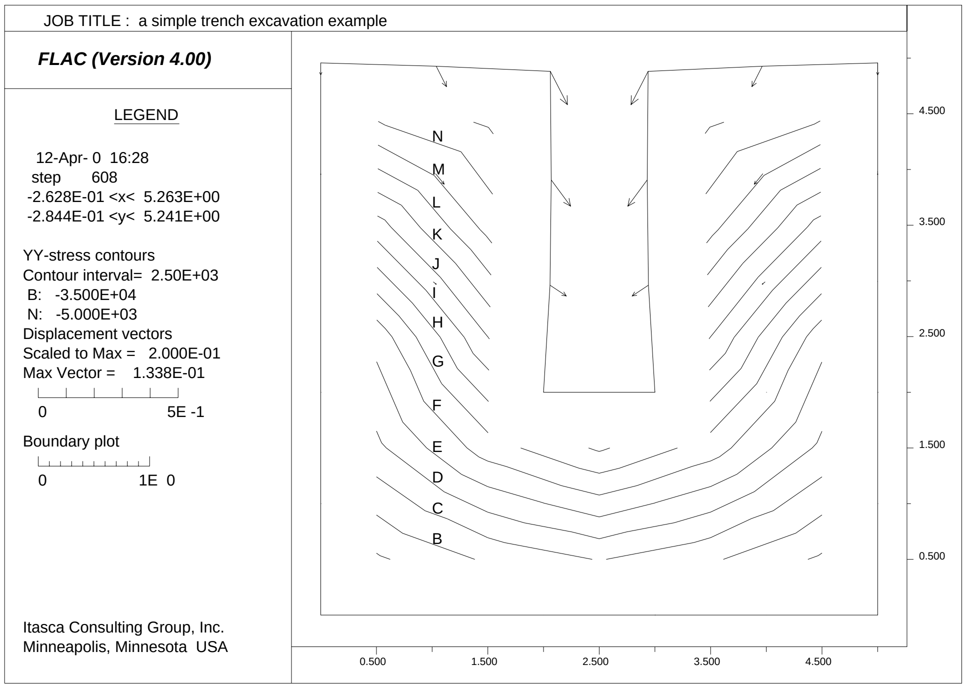


Figure 2.5 yy-stress contours and displacement vectors after 608 timesteps

From this point, you may wish to play with the various features of *FLAC* in an attempt to stabilize the excavation. Try restarting the previous file you created by entering

```
rest  trench.sav
```

Excavate the trench as before, but try using the structural element logic described in Section 1 in **Structural Elements** to model bracing or tieback anchors.

You will see that *FLAC* is virtually bullet-proof — an error-trapping function recognizes most commonly-occurring errors.

This ends the command-driven tutorial. We recommend that you now try running *FLAC* in menu-driven mode, following the instructions in Section 2.2.2. Then, for comparison of command-driven versus menu-driven operation, try the *GIIC* tutorial given in Section 2.2.2.4. After that, read the rest of Section 2 for a beginner's guide to the mechanics of using *FLAC*. As you become more familiar with the code, turn to Section 3 for additional details on problem solving with *FLAC*.

* It is possible to use an interpolation function to extend the stress contour lines to the boundaries, using the *FISH* built-in language (see "EXTRAP.FIS" in Section 3 in the ***FISH* volume**).

2.2.2 Running FLAC in Menu-Driven Mode

The *Graphical Interface for Itasca Codes* (*GIIC*) is a menu-driven graphical interface developed to assist users in operating Itasca codes. The *FLAC-GIIC* is easy to use with a point-and-click operation that accesses all commands and facilities in *FLAC*. The structure of the *GIIC* is specifically designed to emulate expected Windows features and allows general mouse manipulation of displayed items that correspond to *FLAC* operations. You should be able to begin solving problems with *FLAC* immediately, without the need to wade through commands to select those necessary for your desired analysis. This section provides an introduction to the *GIIC* and includes a simple tutorial to help you get started. You will notice that a HELP menu is provided in the main menu bar for the *GIIC*. HELP buttons are also included with each tool in the *GIIC*, and HELP panes can be opened by right-clicking on model tooltabs. Consult these *Help* views for detailed information on specific *GIIC* features. All of the components of the *GIIC* are described in the ***FLAC-GIIC* Reference**.

2.2.2.1 Entering the GIIC and Selecting Analysis Options

You can enter the *GIIC* simply by typing the command

```
giic
```

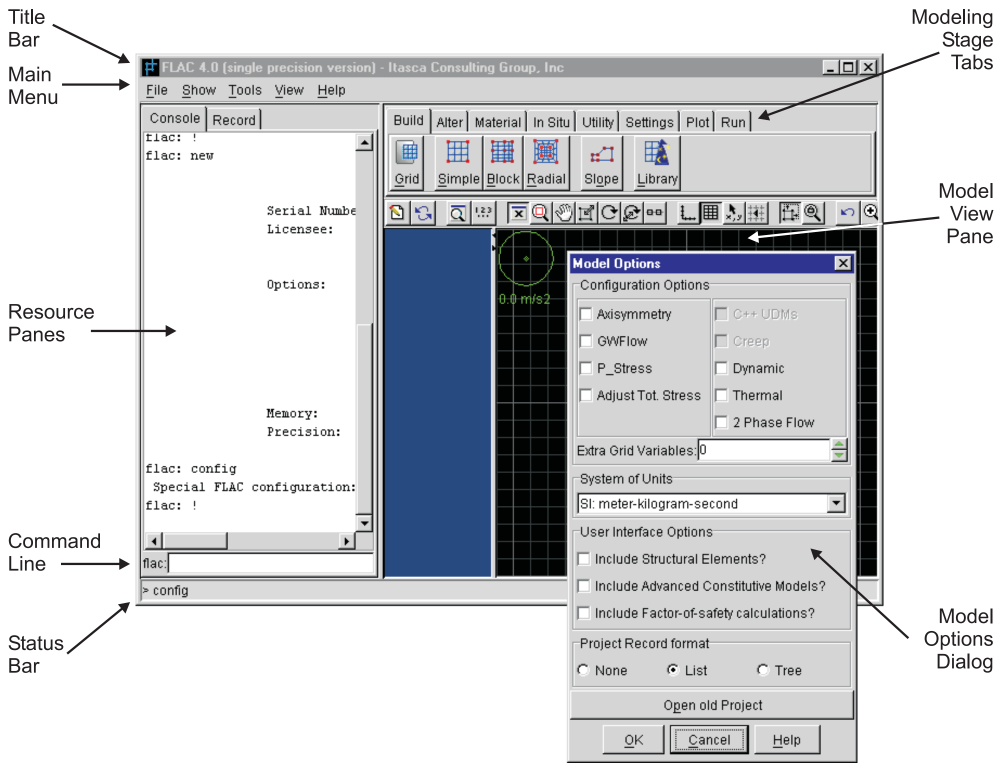
from the `flac:` prompt if you are in the text mode after loading *FLAC*. The **GIIC** command is automatically included in the "FLAC.INI" file (see Section 2.1.7) when you install *FLAC* from the Itasca CD, so that the graphical interface will start up every time *FLAC* is loaded. The *GIIC* main window is shown in Figure 2.6.

The code name and current version number are printed in the title bar at the top of the window, and a main menu bar is positioned just below the title bar. Beneath the main menu bar are two windows: a *resources* pane and a *model-view* pane. The resources pane contains two tabbed panes with text-based information. A *Console* pane shows text output and allows command-line input (at the bottom of the pane). A *Record* pane shows a record of commands needed to generate the current model project state. This record can be exported to a data file as a set of *FLAC* commands that represent the problem being analyzed.

The model-view pane shows a graphical view of the model. Additional tabbed views can be added to this window which display user-defined plots. At the top of the model-view pane is a tab bar containing modeling-stage tabs. When you click on a modeling-stage tab, a tool bar will open; this contains buttons that access model tool panes. The tool bar for the model BUILD tool is shown in Figure 2.6. When you click on a button, this opens a modeling-stage pane; these panes contain all the tools you will need to create and run your model.

You can use the VIEW menu to manipulate any view pane (e.g., translate or rotate the view, increase or decrease the size of the view, turn on and off the model axes). The *View* menu is also available as a tool bar that can be turned on from the SHOW menu. The *View* tool bar is shown on the model-view pane in Figure 2.6.

An overview of the *GIIC* operation is provided in the HELP menu. The menu also contains a list of Frequently Asked Questions about the *GIIC* and an index to all *GIIC Help* files.



```
flac: !
flac: new

                 Serial Numbe
                 Licensee:


                 Options:






                 Memory:
                 Precision:

flac: config
 Special FLAC configuration:
flac: !
```

flac:

> config

Extra Grid Variables: 0

System of Units: SI: meter-kilogram-second

Project Record format: None [✓ List] Tree

Figure 2.6 The GIIC main window

The text field with the `flac:` prompt located at the bottom of the *Console* pane allows you to enter *FLAC* commands directly from the *GIIC*. The *Console* pane will echo the commands that you enter. You should not need to use the command line at all; it is provided as a shortcut if you prefer to type a command rather than use the graphical interface. A status bar is located at the bottom of the main window and displays information related to the currently active view or tool.

There is also a *Fish Editor* pane available in the *GIIC* that allows you to create new *FISH* functions, edit existing functions and specify *FISH* parameters. This window can be opened from the SHOW menu.

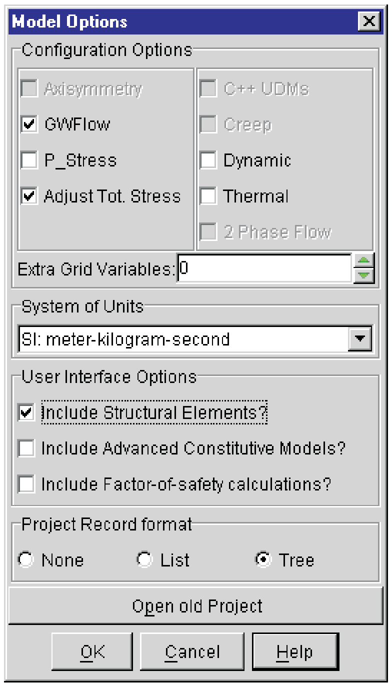
A *Model Options* dialog box will appear every time you start the *GIIC* or begin a new model project. The dialog is shown in Figure 2.6. This dialog identifies which optional modes of analysis are available to you in your version of *FLAC*. (Note that dynamic analysis, thermal analysis, two-phase flow analysis, creep models and C++ user-defined models are separate modules that can be activated at an additional cost per module.) The *FLAC* Configuration Options must be selected at the beginning of a new analysis, while the User Interface Options (structural elements, advanced material models and factor-of-safety calculation) can be included at any time in the model run.

You can select a system of units for your analysis in the *Model Options* dialog. Many parameters will then be labeled with the corresponding units, and predefined values, such as gravitational magnitude and properties within the material database, will be converted to the selected system. The selection for system of units must be done at the beginning of the analysis.

If you are a new user, or only intend to perform a simple static analysis, we recommend that you click the OK button in the *Model Options* dialog to access the basic *FLAC* features. In this case, only the null, isotropic elastic and Mohr-Coulomb models are active, and a static, plane-strain analysis is performed in the *GIIC*. If you wish to come back later in the analysis and, for example, add structural elements, click FILE / MODEL OPTIONS in the main menu. This will reopen the *Model Options* dialog. Check *Include Structural Elements?* and click OK. A STRUCTURE tab will be added to the modeling-stage tab bar, and structural elements can now be included in your model.

The final model option that can be selected is the format for the project record that is used in the *Record* pane. Two types of format are provided: a *List Record* format and a *Project Tree Record* format. The List format is a simple record with independent save files. Each save file includes a record of all the commands needed to generate the state. The Project Tree format shows changes between save files. Save states are displayed in a tree structure.

The *Model Options* dialog is shown below in Figure 2.7 with the following model options selected: groundwater configuration option with automatic adjustment of total stresses for external pore-pressure change (**CONFIG gw ats**), structural elements user-interface option, Project Tree Record format and SI system of units.


Model Options

Configuration Options

- [ ] Axisymmetry
- [✓] GWFlow
- [ ] P_Stress
- [✓] Adjust Tot. Stress
- [ ] C++ UDMs
- [ ] Creep
- [ ] Dynamic
- [ ] Thermal
- [ ] 2 Phase Flow

Extra Grid Variables: 0

System of Units

SI: meter-kilogram-second

User Interface Options

- [✓] Include Structural Elements?
- [ ] Include Advanced Constitutive Models?
- [ ] Include Factor-of-safety calculations?

Project Record format

None List [✓ Tree]

Figure 2.7 The Model Options dialog box

2.2.2.2 Changing GIIC Preferences

After you have selected which *Model Options* you wish to have operating during your analysis, you can save these preferences, so that these selections are active each time you enter the *GIIC*. Also, you can save your preferences for the look-and-feel of the *GIIC* on start-up. You can select which resource pane you wish to have open, as well as the size of this pane and the *Model*-view pane. Preferences for the *GIIC* appearance can be changed. Open the SHOW menu in the main menu to change the look-and-feel of the *GIIC* panes and tool bar. Once you are satisfied, click FILE / SAVE PREFERENCES in the main menu. The *GIIC* start-up preferences are stored in the file "STARTUP.GPF," located in the "ITASCA\FLAC\GUI" directory.

2.2.2.3 Modeling-Stage Tabs

The model tools are accessed from the modeling-stage tab bar located above the model-view pane. The tabs are arranged in a logical progression for building and solving your model. The order follows the recommended procedure for problem solving discussed in Section 2.6. The first two modeling-stage tabs contain tools to generate and shape the grid to fit the problem domain.

- The grid is first created via the BUILD tab, and
- then shaped to fit the problem geometry via the ALTER tab.
- Next, material models and properties are assigned to the zones in the model, using the tools accessed from the MATERIAL tab.
- Boundary and initial conditions are applied via the IN SITU tab.
- The UTILITY tab provides tools to monitor model variables and access existing *FISH* functions.
- The SETTINGS tab allows model conditions to be set or changed during the analysis.
- All plotting facilities in *FLAC* are accessible via the PLOT tab.
- Calculations are performed using tools from the RUN tab.

Note that model conditions can be changed at any point in the solution process by re-entering a modeling-stage tab. For example, model properties can be changed at any time via the MATERIAL tab, and pressure or stress alterations can be made via the IN SITU tab. Also, if you select structural elements in the *Model Options* dialog, a STRUCTURE tab will be included in the modeling-stage tab bar to access structural support for the model.

When you click on each of the modeling-stage tabs, a tool bar will appear that provides access to model tool panes in which you can perform operations related to that tool. The BUILD tab tool bar is shown in Figure 2.6. Next, a simple tutorial is given to provide an introduction to the model tools and to help you become acquainted with the *GIIC* operation.

2.2.2.4 A Simple Tutorial — Use of the GIIC

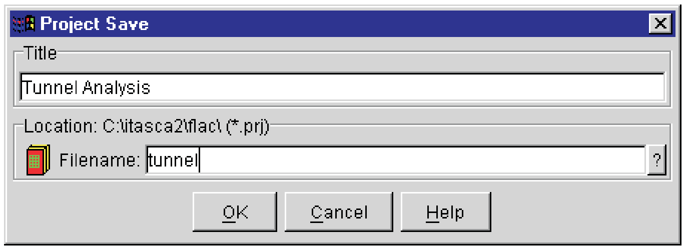
In this section we provide a simple tutorial to help you get started using the *GIIC*. The tutorial demonstrates the use of several modeling tools to create and solve a simple geotechnical problem.

The example is a circular tunnel excavated at a shallow depth in rock. Two rock types are evaluated: a strong rock and a weak rock. We excavate the tunnel instantaneously and monitor the movements of the rock around the tunnel for both rock types. This tutorial is similar in scope to the command-driven tutorial presented in Section 2.2.1.2, and is provided to allow you to compare command-driven versus menu-driven operation of *FLAC*.

To begin, start up the *GIIC* by following the procedure given in Section 2.2.2.1. (If you have loaded *FLAC* by double-clicking on the *FLAC* icon in the Itasca Codes group, the *GIIC* will start up automatically — see Section 2.1.6.)

We are performing a simple, static, plane-strain analysis, so we click the OK button in the *Model Options* dialog to access the basic *FLAC* features. (See Figure 2.6.) By default, the *List Record* format is selected.

When beginning a modeling project, we first select the SAVE PROJECT AS... menu item from the FILE menu in order to set up a project save file. After we select this menu item, we are asked to assign a project title and filename for this project. The dialog is shown in Figure 2.8. We click on ? in this dialog to select a directory in which to save the project file. We save the project as “TUNNEL.PRJ” (Note that the “.PRJ” extension is assigned automatically.) The project file contains the project record and allows access to all the model save (“.SAV”) files that we will create for the different stages of this analysis. (See Section 2.10 for a discussion of the differences between a model save (“.SAV”) file and the project save (“.PRJ”) file.) We can stop working on the project at any stage, save it and re-open it at a later time simply by opening the project file (from the FILE/OPEN PROJECT menu item); the entire project and associated model save files will be accessible in the *GIIC*.



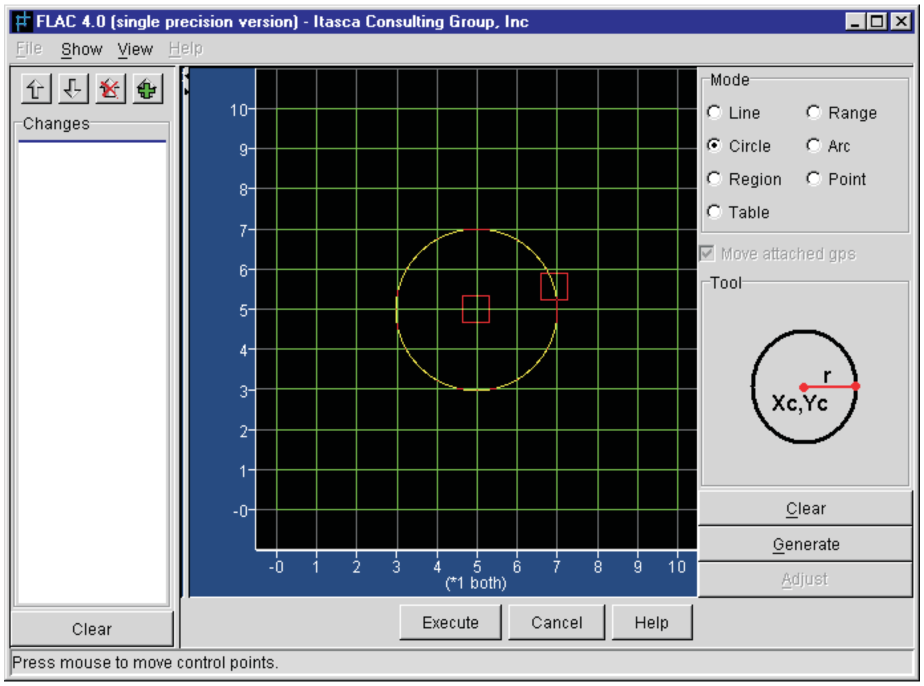
Tunnel Analysis

Filename: tunnel

Figure 2.8 Project Save dialog

We now begin the model creation. To set up the initial finite difference grid, we click on the GRID button from the BUILD modeling-tool tab. This tool invokes the **GRID** command. We press OK in the *How many zones?* dialog to select the default grid of 10 zones in the i-direction by 10 zones in the j-direction. A plot of the grid will immediately be shown in the *Model*-view pane. We will use SI units for this example (see Section 2.8 for selection of system of units). The model domain is then 10 m by 10 m. Click on the VIEW / SHOWAXIS VALUES menu item to show the x- and y-axes for the model.

We next create the circular tunnel by shaping the grid to fit the tunnel boundary. To avoid errors in calculation of gridpoint masses, all grid shaping should be done before the computational process begins; these errors may occur if the grid is shaped after computational stepping (see Section 2.6.1 for further discussion). Grid shaping is done by clicking on the SHAPE button from the ALTER modeling-tool tab. A plot of the grid appears with a set of tools that we can use to add shapes to the grid. We select the CIRCLE radio button, move the mouse to a position on the grid corresponding to the tunnel center, and press and hold the left mouse button while moving the mouse. A circle tool will appear with two red boxes, one at the centroid and one along the circle periphery (see Figure 2.9). We can move the circle and adjust its radius by pressing and holding the left mouse button while the mouse is positioned within each box. Alternatively, we can select values for the centroid coordinates and the circle radius with dialogs that open when we right-click the mouse while it is positioned within each box. The circle in Figure 2.9 is centered at x = 5.0, y = 5.0 and has a radius of 2.0 m.

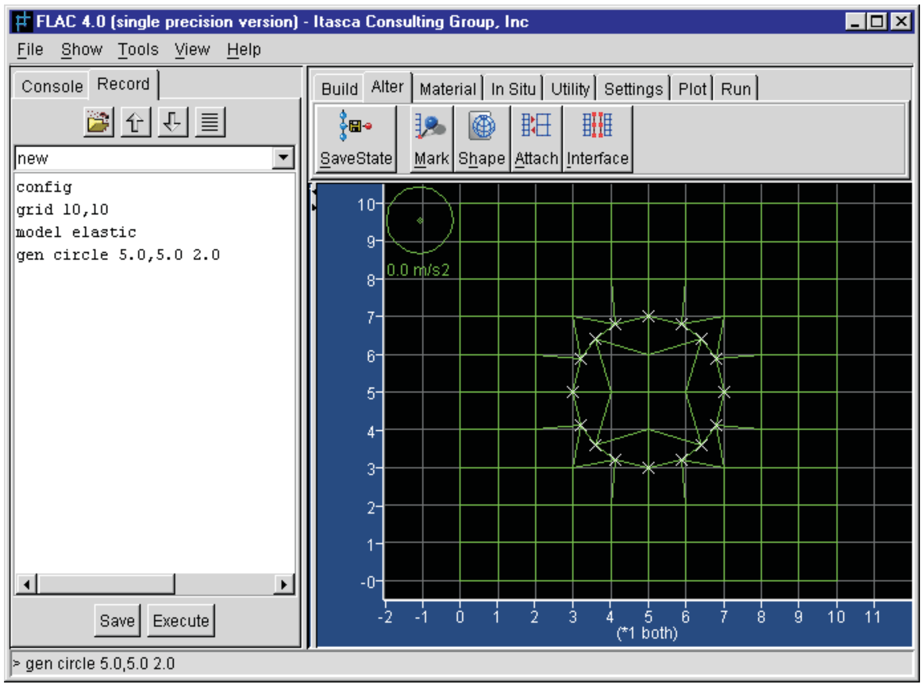


Press mouse to move control points.

Figure 2.9 GIIC virtual grid with CIRCLE button active

When we press GENERATE, the grid is deformed to fit the boundary of the circle, and the corresponding **GENERATE** command is displayed in the *Changes* sub-pane to the right of the grid plot. Note that this is a "virtual" grid: any alterations we make within this grid can be undone or changed. We simply press one of the arrow keys above the *Changes* pane to remove (or add) a command corresponding to the shape created in the virtual grid.

Once we are satisfied with the alteration, we press the EXECUTE button. This sends the command(s) to *FLAC* and returns to the *Model*-view pane. The *FLAC* commands are processed, and the altered *FLAC* grid with marked gridpoints is displayed, as shown in Figure 2.10. The *FLAC* commands created thus far are shown in the *List Record* pane in this figure.

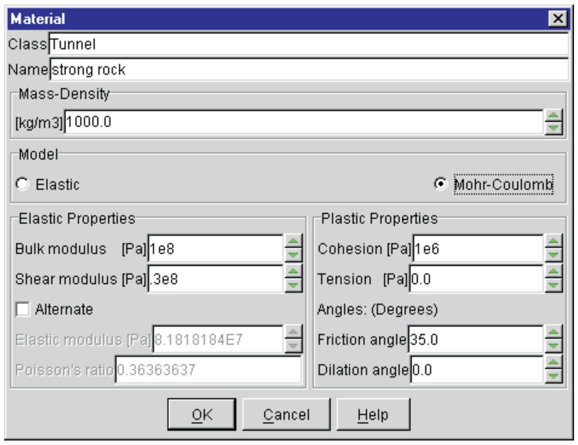

FLAC 4.0 (single precision version) - Itasca Consulting Group, Inc
File Show Tools View Help
Console Record
Build Alter Material In Situ Utility Settings Plot Run
SaveState Mark Shape Attach Interface
0.0 m/s2
(*1 both)
Save Execute

```
new
config
grid 10,10
model elastic
gen circle 5.0,5.0 2.0
```

```
> gen circle 5.0,5.0 2.0
```

Figure 2.10 FLAC grid with zones shaped for circular tunnel

We next move to the MATERIAL modeling-tool tab and press the ASSIGN button to create and assign materials and their properties to the zones within the grid. Materials are created from a *Material* dialog that is opened by pressing the CREATE button in the *Assign* pane. Within this dialog, we can assign a classification and material name, prescribe a constitutive model type (elastic or Mohr-Coulomb) and assign material properties. Soils and rocks can be divided into different classifications, such as "Tunnel" rock, with separate material names within a classification, such as "strong rock" and "weak rock." The classification and material name are used to associate a **GROUP** name with each material. We will create two different materials for this analysis: a strong rock and a weak rock. The *Material* dialog with the selected properties for strong rock is shown in Figure 2.11. The dialog for weak rock is similar except that the cohesion is zero.*



Class: Tunnel
Name: strong rock
Mass-Density [kg/m3]: 1000.0
Model: Elastic [✓ Mohr-Coulomb]

Elastic Properties
Bulk modulus [Pa]: 1e8
Shear modulus [Pa]: 3e8
Alternate: ☐
Elastic modulus [Pa]: 8.1818184E7
Poisson's ratio: 0.36363637

Plastic Properties
Cohesion [Pa]: 1e6
Tension [Pa]: 0.0
Angles: (Degrees)
Friction angle: 35.0
Dilation angle: 0.0

Figure 2.11 Material properties dialog in the ASSIGN tool

We press OK in the *Material* dialog to create the material. The material is added to a material *List* shown on the right side of the *Assign* pane. Once all the materials required for an analysis have been created and added to the list, they can be assigned to the grid. It is possible to assign different materials to different zones in the grid, or to different marked regions of the grid, using the *Range* tools provided in the *Assign* pane. In our example, we will evaluate the response of the tunnel in strong rock versus weak rock, so we begin by assigning strong rock material to all zones. We highlight the *Tunnel:strong rock* item and press the SETALL button to assign this material to all non-null zones in the grid. Figure 2.12 shows the *Assign* pane with the strong rock material assigned. **GROUP**, **MODEL** and **PROPERTY** commands are listed in the *Changes* pane when the materials are assigned. We now press EXECUTE to send these commands to *FLAC*.

* A database of common soil and rock materials and properties is also available by pressing the DATABASE button in the lower-right corner of the *Assign* pane. The database is divided into classification groups and material names. You can also create your own database of common materials within this database tool, which can be saved and loaded for other projects. Database materials are stored in a file with extension ".GMT" — see Section 2.10.

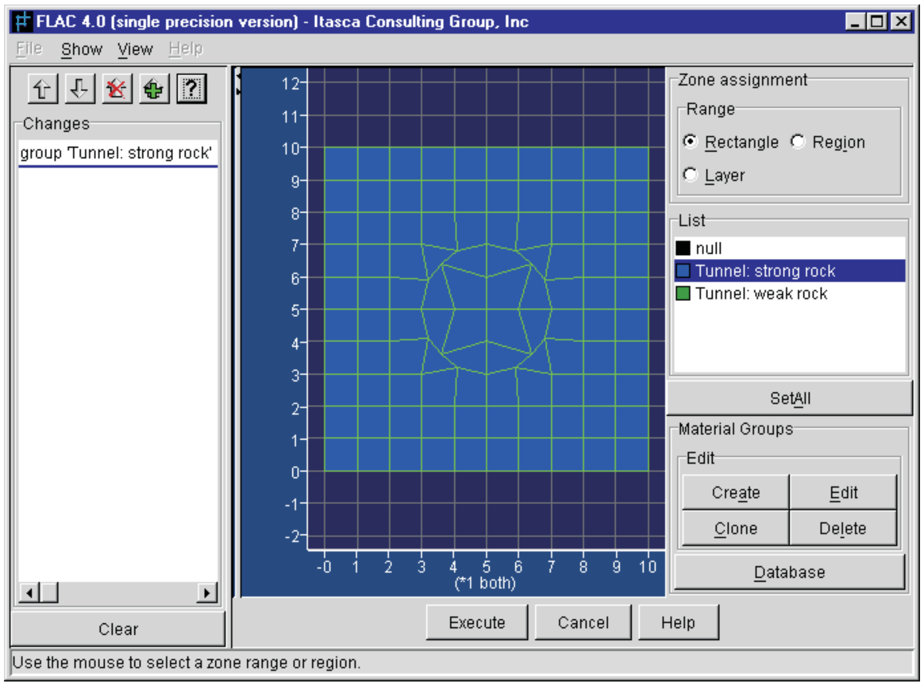
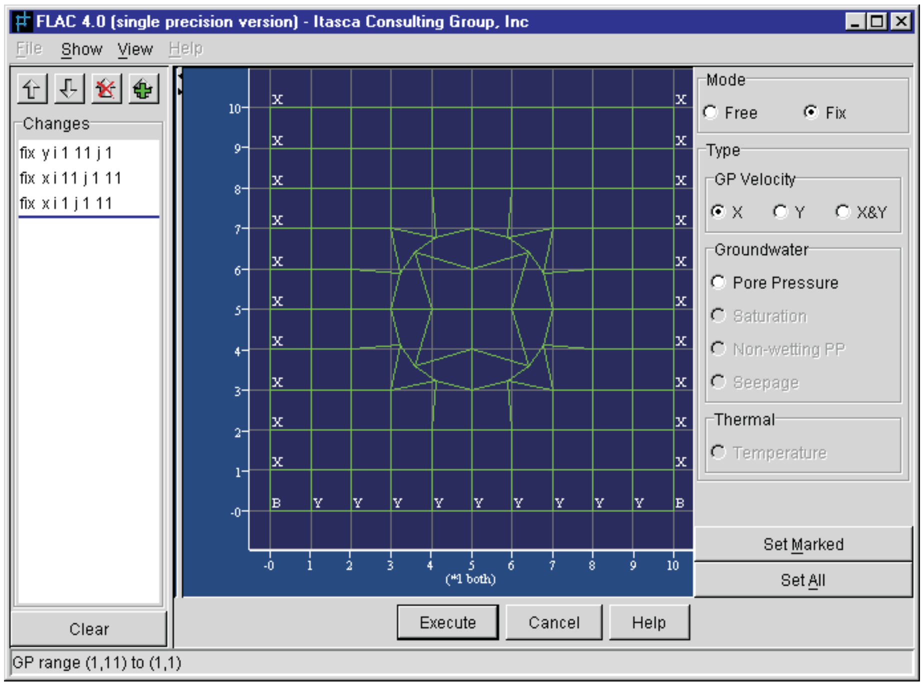


Figure 2.12 Strong rock material assigned to all zones with the SETALL ***button***

The next step is to assign the boundary conditions for our model. We select the FIX button from the IN SITU modeling-tool tab. We wish to have roller boundary conditions applied to the bottom and sides of the model. To prescribe a roller boundary on the bottom, we press the Y radio button to specify a fixed-gridpoint velocity in the y-direction. By default, the y-velocity is zero and, by specifying that this velocity is fixed at the selected gridpoints, we are preventing any movement in the y-direction. We hold down the left mouse button while dragging the mouse along the bottom boundary. Gridpoints are marked and, when we release the button, a letter denoting the y-fixity condition is printed at the selected gridpoints. We repeat the process using the X radio button to specify a fixed-gridpoint velocity in the x-direction along the left and right boundaries. The resulting boundary conditions are shown in Figure 2.13. Press EXECUTE to send these commands to *FLAC*.



Changes:

```
fix y i 1 11 j 1
fix x i 11 j 1 11
fix x i 1 j 1 11
```

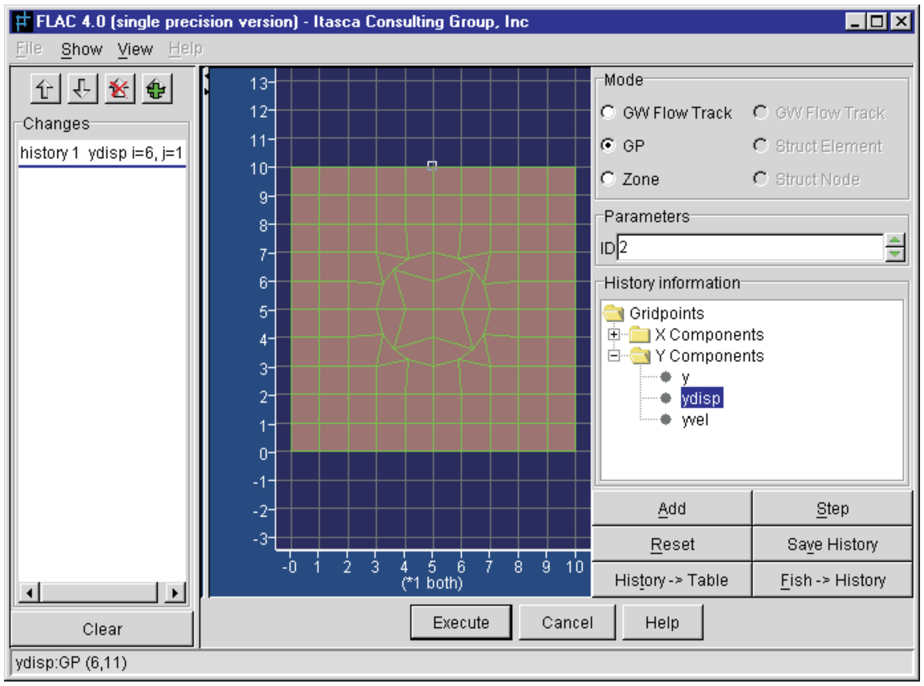
Mode: Free [✓ Fix]

GP Velocity: [✓ X] Y X&Y

GP range (1,11) to (1,1)

Figure 2.13 Boundary conditions specified with the FIX ***tool***

We can access different variables in our model with the UTILITY modeling-tool tab. We wish to monitor the displacement at the ground surface as the tunnel is excavated. To do this we click on the HISTORY button to open the *History* pane, and then click on the GP mode radio button. We select the y-displacement history from the *History Information* sub-menu, and then we point the mouse at a gridpoint on the top of the model. When we click on the gridpoint, a **HISTORY** command is created for the y-displacement history at that gridpoint. Figure 2.14 shows the results of our action in the *History* pane. Press EXECUTE to send the command to *FLAC*.



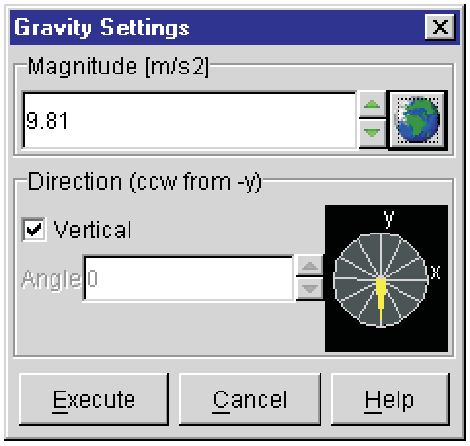
history 1 ydisp i=6, j=1

ID 2

ydisp:GP (6,11)

Figure 2.14 Select variables to monitor with the HIST *tool*

Gravitational loading is specified as a global setting in our model via the SETTINGS modeling-tool tab. We click on the GRAVITY button to access the *Gravity Settings* dialog. Then, by clicking on the globe icon in the dialog, the value of 9.81 m/sec^2 will be listed as the magnitude of gravitational acceleration. (You can also type in a different value for the magnitude.) The dialog is shown in Figure 2.15. Note that the gravitational vector is shown by an icon in the model view.



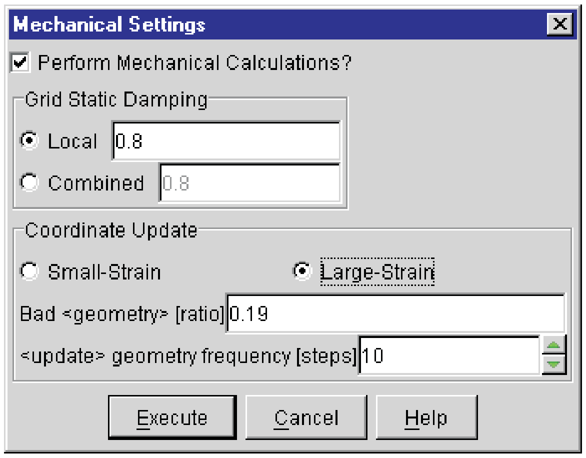
Magnitude [m/s2] 9.81

Vertical [✓]

Angle 0

Figure 2.15 Set gravity settings in the Gravity Settings dialog

We anticipate that large deformations will occur in this analysis, so we click on the MECH button from the SETTINGS tab to access the *Mechanical Settings* dialog. We press the LARGE-STRAIN radio button to set the large-strain logic. Figure 2.16 shows the *Mechanical Settings* dialog.



[✓] Perform Mechanical Calculations?

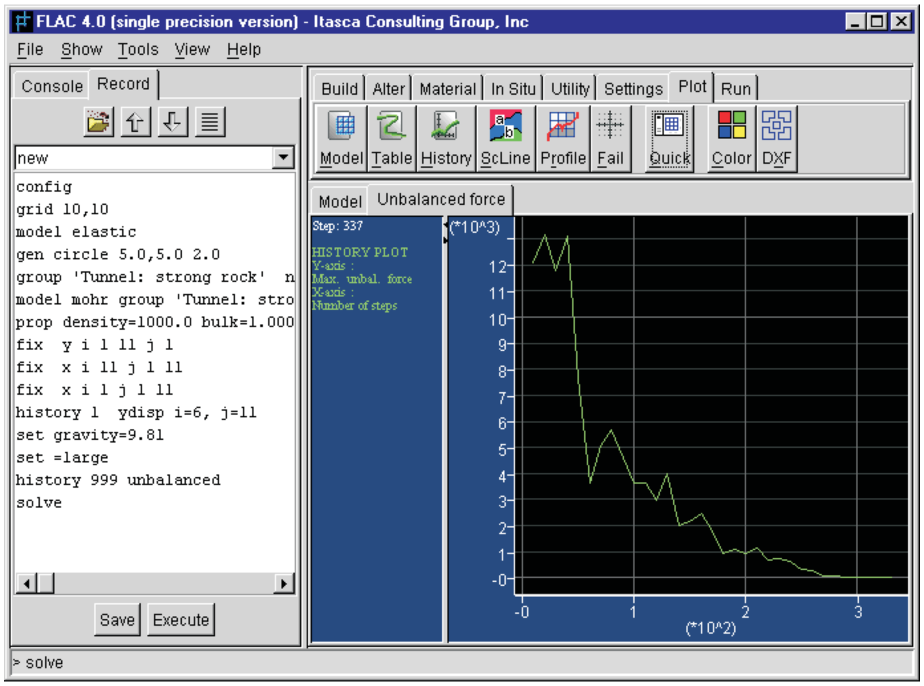
Grid Static Damping
- [✓ Local] 0.8
- Combined 0.8

Coordinate Update
- Small-Strain [✓ Large-Strain]
- Bad <geometry> [ratio] 0.19
- <update> geometry frequency [steps] 10

Figure 2.16 Set global mechanical settings in the Mechanical Settings dialog

We are now ready to bring the model to an initial equilibrium state. We timestep the model to a force equilibrium condition under gravity loading. The solution approach of timestepping to equilibrium is described is Section 2.6.4. We press the RUN modeling-tool tab and then the SOLVE button. This opens a *Solve* tool; we press OK and invoke the **SOLVE** command to detect equilibrium automatically. The *Solve* dialog appears and the timestep number, maximum unbalanced force and equilibrium ratio are displayed. The equilibrium ratio is used to determine equilibrium (see Section 2.6.4 for details). When the ratio falls below the default limiting value of 10^{-3}, the calculation stops. Other limiting conditions can also be prescribed, as described in Section 2.6.4.

There are several ways to check that equilibrium has been reached. A quick check can be made by plotting the change in maximum unbalanced force during stepping. Press the PLOT modeling-tool tab, then the QUICK button, and finally the UNBALANCED FORCE item, and a plot of unbalanced force versus accumulated timestep will appear. The plot given in Figure 2.17 shows that the maximum unbalanced force is approaching zero, which indicates that an equilibrium state is reached.


FLAC 4.0 (single precision version) - Itasca Consulting Group, Inc
File Show Tools View Help
Console Record
new
Save Execute
Build Alter Material In Situ Utility Settings Plot Run
Model Table History ScLine Profile Fail Quick Color DXF
Model Unbalanced force
Step: 337
HISTORY PLOT
Y-axis :
Max. unbal. force
X-axis :
Number of steps
(*10^3)
12 11 10 9 8 7 6 5 4 3 2 1 -0
-0 1 2 3
(*10^2)
> solve

```
config
grid 10,10
model elastic
gen circle 5.0,5.0 2.0
group 'Tunnel: strong rock' n
model mohr group 'Tunnel: stro
prop density=1000.0 bulk=1.000
fix  y i 1 11 j 1
fix  x i 11 j 1 11
fix  x i 1 j 1 11
history 1  ydisp i=6, j=11
set gravity=9.81
set =large
history 999 unbalanced
solve
```

Figure 2.17 History of maximum unbalanced force from the QUICK button

It is a good idea to save the project state at the different stages of our analysis. In this way we can easily return to a given state and make modifications without the need to run the entire simulation again. We can save our project model state at the initial-equilibrium stage by pressing the SAVESTATE button in the RUN tool bar. This opens a dialog box that allows the user to give a descriptive title to the saved state, select a directory in which to save the model-state file, and name the file. By default, the file has the extension ".SAV." The save file is described in Section 2.10.

We choose to save the model state as "TUN1.SAV." We must save this file in the same directory as the project file "TUNNEL.PRJ" so that the project can be opened later and list all associated save files. The save file is added to a list at the top of the *List Record* pane. Each time we save the model state, a new save file will be added to the list. We can click on any file in the list to open that saved state.

We can create plots for a wide variety of variables in a *FLAC* model. Click on the MODEL button from the PLOT tab to open the *FLAC Plot Items* dialog box. The dialog is shown in Figure 2.18. This dialog accesses most of the general plotting facilities in *FLAC*. (Note that separate tools are provided for table, history, profile and failure plots in the PLOT tool bar.) For example, if we wish to examine the gravitational stresses that develop in the model, we can create a contour plot of σ_{yy}-stresses. Click on the CONTOUR-ZONE / TOTAL STRESS / SYY plot item from the *Plot Items* tree, and add this to the *Add Plot Items* list. Then click on the GEOMETRY / BOUNDARY plot item and add this to the list. We can either create a fill-contour plot or a line-contour plot. By default, a filled contour plot is created with the contour range denoted by the fill colors. The resulting fill-contour plot is shown in Figure 2.19.

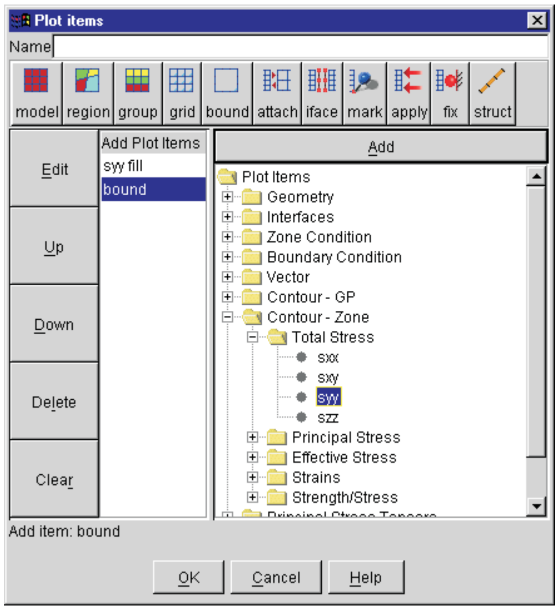
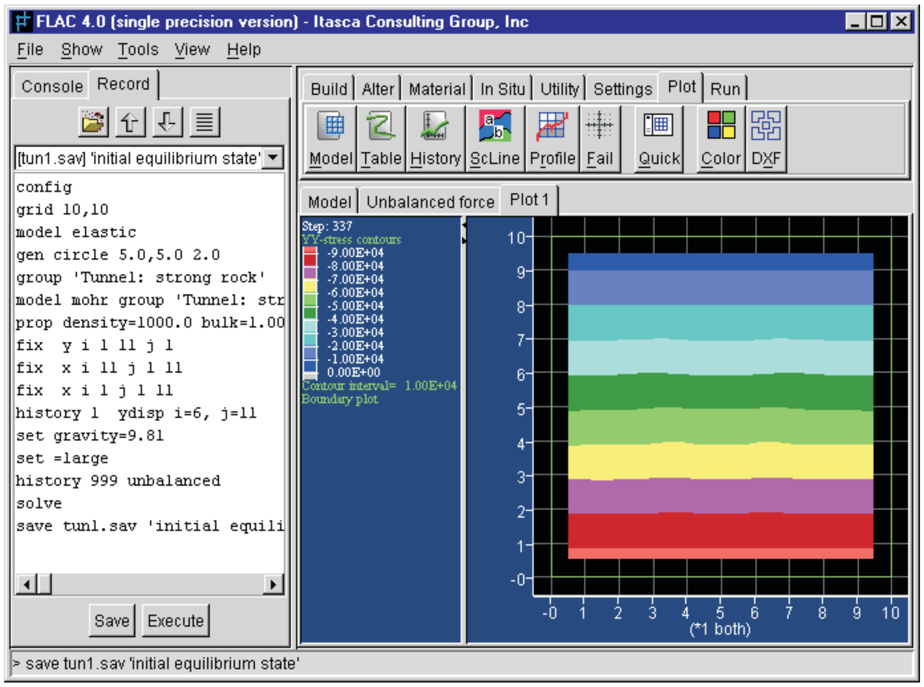


Figure 2.18 Plot Items dialog

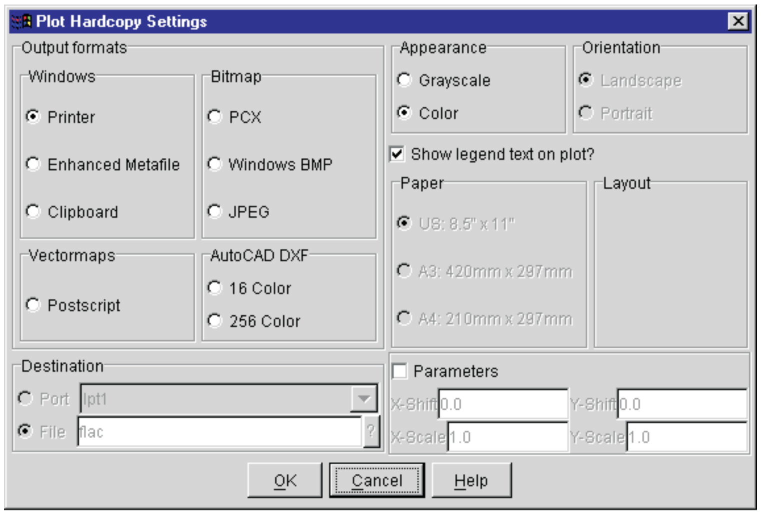


```
config
grid 10,10
model elastic
gen circle 5.0,5.0 2.0
group 'Tunnel: strong rock'
model mohr group 'Tunnel: str
prop density=1000.0 bulk=1.00
fix  y i 1 11 j 1
fix  x i 11 j 1 11
fix  x i 1 j 1 11
history 1  ydisp i=6, j=11
set gravity=9.81
set =large
history 999 unbalanced
solve
save tunl.sav 'initial equili
```

```
> save tun1.sav 'initial equilibrium state'
```

Figure 2.19 Contour plot of σ_{yy}-stresses resulting from gravitational loading

We can make a hardcopy plot of any *FLAC* model plot we choose. To do this, click on the File/Print Plot menu item in the main menu. If the current Windows default printer is connected to the LPT1 port, we can send the currently active plot view directly to the printer by clicking this menu item. The File/Print Plot Setup menu item can be used to change the printer device settings. Figure 2.20 shows the *Plot Hardcopy Settings* dialog with the selections for device settings.



Plot Hardcopy Settings

Output formats
- Windows: [✓ Printer] Enhanced Metafile Clipboard
- Bitmap: PCX Windows BMP JPEG
- Vectormaps: Postscript
- AutoCAD DXF: 16 Color 256 Color

Appearance: Grayscale [✓ Color]

Orientation: [✓ Landscape] Portrait

[✓] Show legend text on plot?

Paper: [✓ US: 8.5" x 11"] A3: 420mm x 297mm A4: 210mm x 297mm

Layout

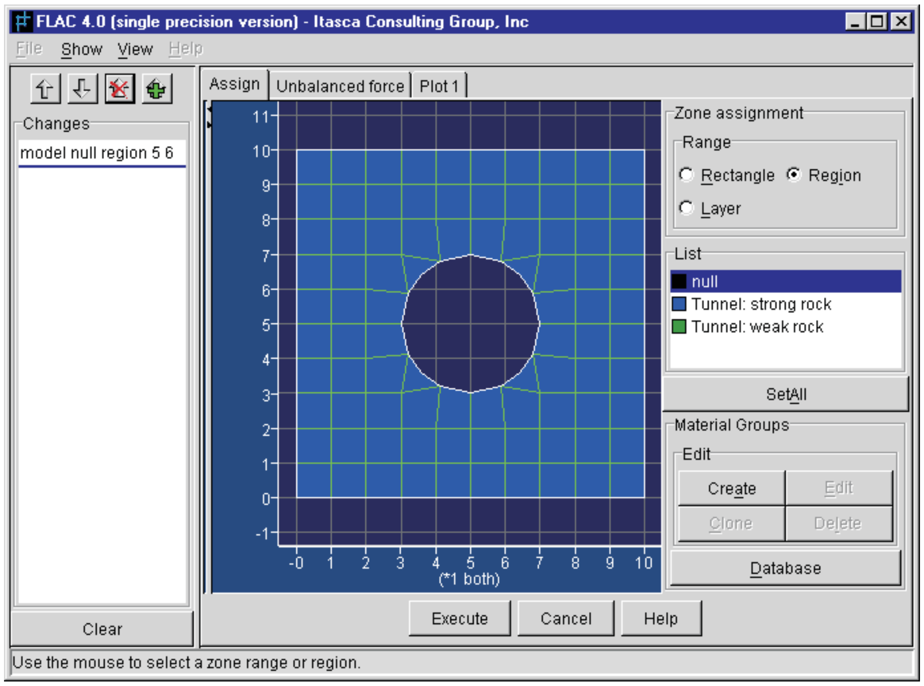
Destination: Port lpt1 [✓ File] flac

[ ] Parameters: X-Shift 0.0 Y-Shift 0.0 X-Scale 1.0 Y-Scale 1.0

Figure 2.20 Plot Hardcopy Settings dialog

We are now ready to excavate the tunnel. We return to the Assign pane from the Material tab. In order to excavate the tunnel, we define the excavated region as null material: highlight the *null* material in the material list and click on the Region radio button in the *Range* sub-pane. Regions are denoted on the virtual grid plot by black lines. By clicking on any zone within the circular tunnel region defined by the black line, all the zones within this region are changed to null material. A **MODEL null** command corresponding to these zones is also listed in the *Changes* sub-pane. Figure 2.21 shows the new model state and lists the commands we have generated. Press Execute to send the new **MODEL** command to *FLAC*.

We save the model state at this stage; “TUN2.SAV” is added to the List Record.



model null region 5 6

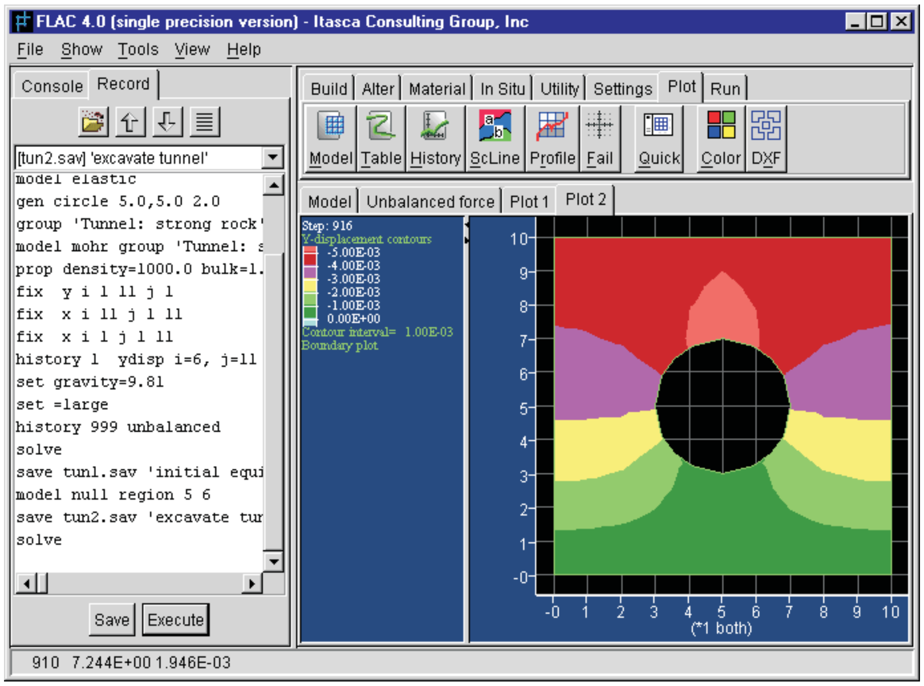
List: null; Tunnel: strong rock; Tunnel: weak rock

Use the mouse to select a zone range or region.

Figure 2.21 FLAC model with tunnel excavated

We now evaluate the behavior of the strong rock material. We perform the analysis by using the SOLVE tool, as we did previously to determine the initial equilibrium state. A stable solution state is calculated, and the resulting displacements are illustrated by the *y*-displacement contour plot shown in Figure 2.22. This plot is created by clicking on the CONTOUR-GP / YDISP plot item from the *Plot Items* tree.

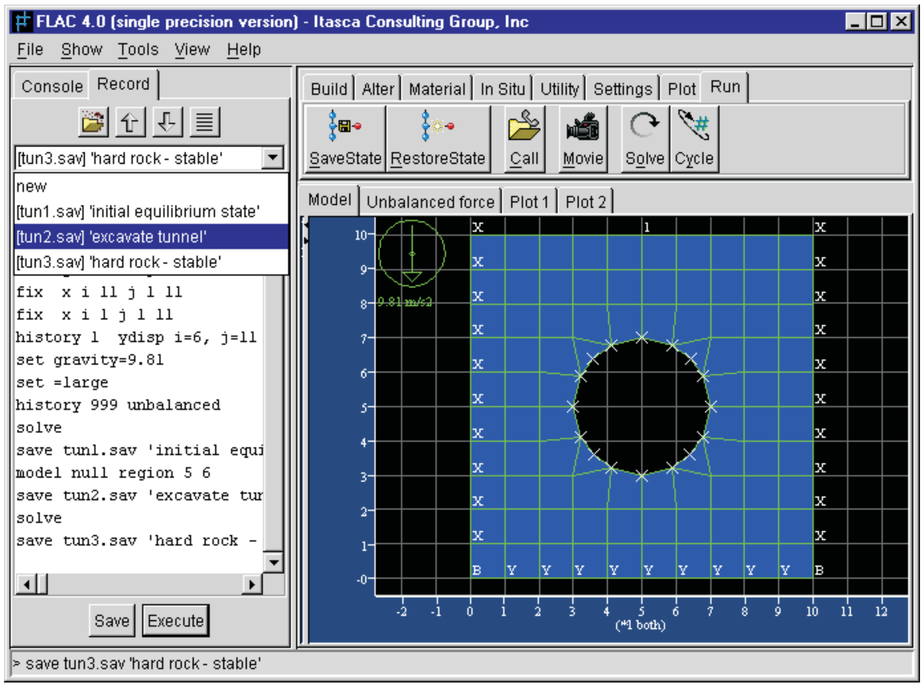
We save this state as "TUN3.SAV"; the *FLAC* commands to create the model to this stage are shown in the List Record in Figure 2.23. The three save states in our project are also shown at the top of the *Record* pane.


FLAC 4.0 (single precision version) - Itasca Consulting Group, Inc
File Show Tools View Help
Console Record
Build Alter Material In Situ Utility Settings Plot Run
Model Table History ScLine Profile Fail Quick Color DXF
Model Unbalanced force Plot 1 Plot 2
Step: 916
Y-displacement contours
-5.00E-03
-4.00E-03
-3.00E-03
-2.00E-03
-1.00E-03
0.00E+00
Contour interval= 1.00E-03
Boundary plot
(*1 both)
Save Execute

```
[tun2.sav] 'excavate tunnel'
model elastic
gen circle 5.0,5.0 2.0
group 'Tunnel: strong rock'
model mohr group 'Tunnel: s
prop density=1000.0 bulk=1.
fix  y i 1 11 j 1
fix  x i 11 j 1 11
fix  x i 1 j 1 11
history 1  ydisp i=6, j=11
set gravity=9.81
set =large
history 999 unbalanced
solve
save tun1.sav 'initial equi
model null region 5 6
save tun2.sav 'excavate tur
solve
```

```
910  7.244E+00 1.946E-03
```

Figure 2.22 Strong rock: y-displacement contours


FLAC 4.0 (single precision version) - Itasca Consulting Group, Inc
File Show Tools View Help
Console Record
Build Alter Material In Situ Utility Settings Plot Run
SaveState RestoreState Call Movie Solve Cycle
Model Unbalanced force Plot 1 Plot 2
9.81 m/s2
(*1 both)
Save Execute

```
[tun3.sav] 'hard rock - stable'
new
[tun1.sav] 'initial equilibrium state'
[tun2.sav] 'excavate tunnel'
[tun3.sav] 'hard rock - stable'
fix  x i 11 j 1 11
fix  x i 1 j 1 11
history 1  ydisp i=6, j=11
set gravity=9.81
set =large
history 999 unbalanced
solve
save tun1.sav 'initial equi
model null region 5 6
save tun2.sav 'excavate tur
solve
save tun3.sav 'hard rock -
```

```
> save tun3.sav 'hard rock - stable'
```

Figure 2.23 Strong rock: state saved as "TUN3.SAV"

Next, we evaluate the tunnel response using the weak-rock material. We first need to return to a previous model stage. We return to the state at which the tunnel is introduced ("TUN2.SAV") by clicking on this file name in the *Record* pane. (Note that we should actually return to the initial equilibrium stage, "TUN1.SAV," and re-calculate an initial stress state that corresponds to the weak-rock strength properties and gravitational loading. In this simple exercise, this initial stress state is the same as that for the strong rock.)

We return to the *Assign* pane and click on *Tunnel: weak rock* in the material list to highlight this material. We click on the SETALL button to change all of the non-null zones from strong rock to weak rock. The result is shown in Figure 2.24.

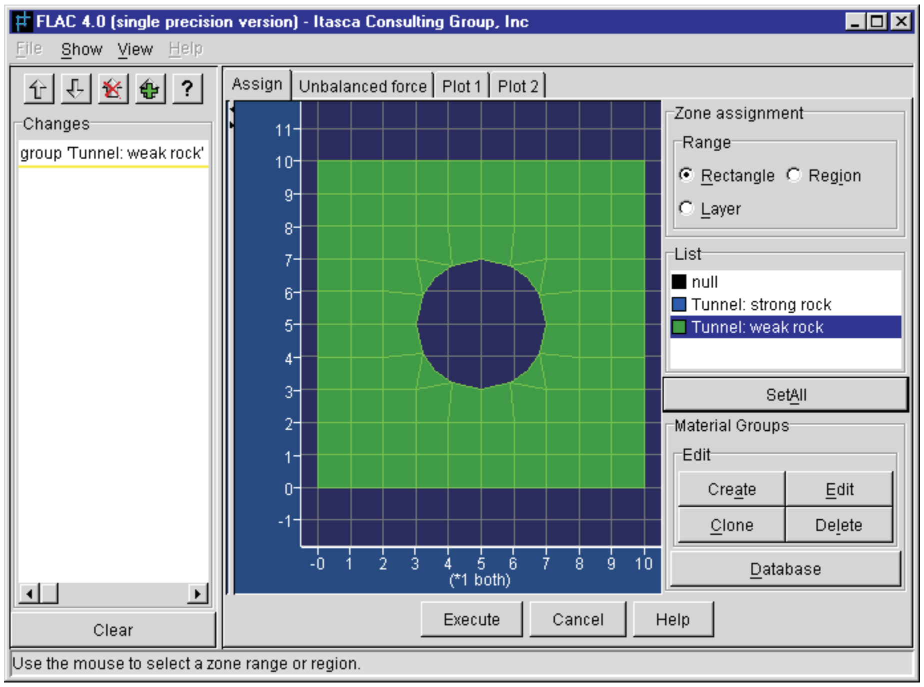
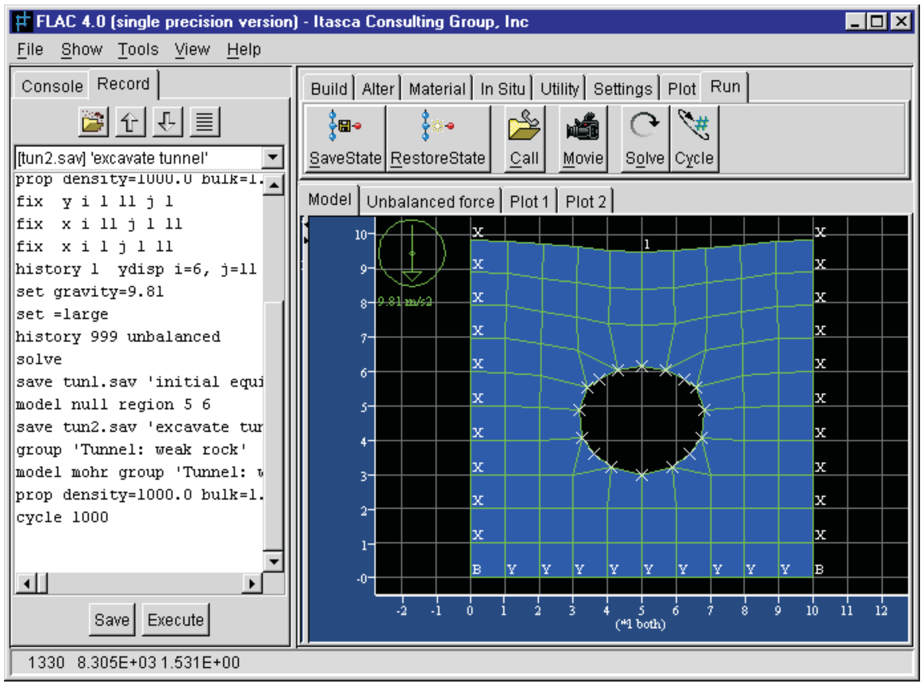


Figure 2.24 Weak rock material assigned to all zones with the SETALL button

For analyses in which we anticipate that material failure can occur and the simulation may never reach an equilibrium state, we do not use the SOLVE tool. Instead, we use the CYCLE tool in the RUN tab in order to step through the simulation and monitor the response as it occurs. After pressing CYCLE, we enter 1000 cycles for the calculation duration and press OK. *FLAC* will now step through 1000 timesteps. When stepping is finished, the model plot is refreshed automatically, and because we are running in large-strain mode, we observe that the top of the grid has begun to deform downward. See Figure 2.25. If we continue stepping, eventually *FLAC* will report an error message ("Bad Geometry") and the calculation will stop. This indicates that zones in the model have reached a limiting distortion; the limiting conditions for zone distortion are described in Section 2.6.1.


FLAC 4.0 (single precision version) - Itasca Consulting Group, Inc
File Show Tools View Help
Console Record
[tun2.sav] 'excavate tunnel'
Save Execute
Build Alter Material In Situ Utility Settings Plot Run
SaveState RestoreState Call Movie Solve Cycle
Model Unbalanced force Plot 1 Plot 2
9.81 m/s2
(*1 both)

```
prop density=1000.0 bulk=1.
fix  y i 1 11 j 1
fix  x i 11 j 1 11
fix  x i 1 j 1 11
history 1  ydisp i=6, j=11
set gravity=9.81
set =large
history 999 unbalanced
solve
save tun1.sav 'initial equi
model null region 5 6
save tun2.sav 'excavate tur
group 'Tunnel: weak rock'
model mohr group 'Tunnel: w
prop density=1000.0 bulk=1.
cycle 1000
```

```
1330  8.305E+03 1.531E+00
```

Figure 2.25 Weak rock: deformed grid at 1000 timesteps after tunnel excavation

There are different ways to monitor the collapse process. For example, if we plot the history of *y*-displacement at gridpoint ***i*** = 6, ***j*** = 11, which we recorded at the beginning of the simulation, we can identify collapse by the increasing displacement that is displayed. Press the History button in the Plot tab, click on *Item ID* number 1 (which is the history number corresponding to the *y*-displacement history we selected), and press OK. A plot of the *y*-displacement history versus accumulated timestep will appear, as shown in Figure 2.26. The displacement is increasing at a constant rate, indicating collapse. (We made this plot view full-screen by clicking off Show / Resources.)

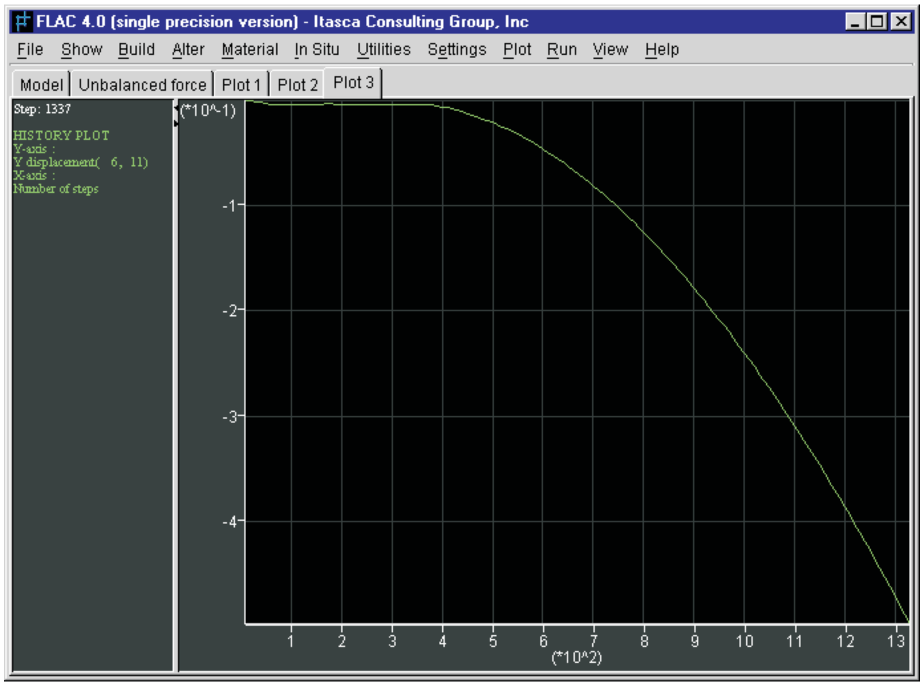


Figure 2.26 Weak rock: y-displacement history at gridpoint 6,11

Finally, we save model stage as "TUN4.SAV," and then update the project file by pressing the FILE / SAVE PROJECT menu item in the main menu. This will automatically update the file "TUNNEL.PRJ." If we desire, we can move the project file and save files to a different directory and restore the project again if we wish to make additional plots or perform other analyses.

This completes the *GIIC* tutorial. We recommend that you now try variations of this example to become more familiar with the *GIIC* operation. For example, begin with the "TUN1.SAV" model state and try adding beam elements along the tunnel periphery in weak rock after nulling the tunnel region to simulate the support provided by a tunnel lining. You can add structural elements via the *Model Options* dialog after restoring the project state. See Section 1 in **Structural Elements** for a description of the beam structural-element logic.

2.3 Nomenclature

FLAC uses nomenclature that is consistent, in general, with that used in conventional finite difference or finite-element programs for stress analysis. The basic definitions of terms are reviewed here for clarification. Figure 2.27 is provided to illustrate the *FLAC* terminology.

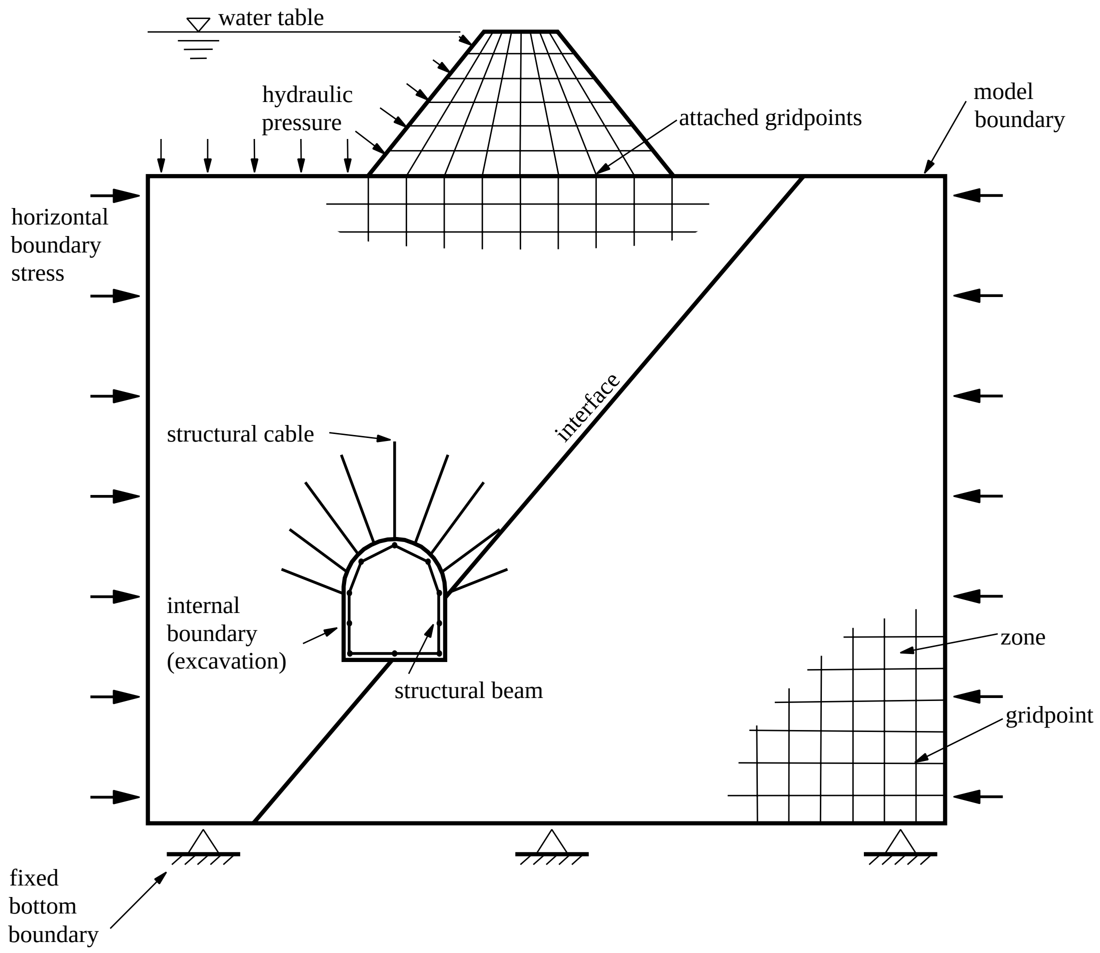


Figure 2.27 Example of a FLAC model

FLAC MODEL — The *FLAC* model is created by the user to simulate a physical problem. When referring to a *FLAC* model, the user implies a sequence of *FLAC* commands (see Section 1 in the **Command Reference**) that define the problem conditions for numerical solution.

ZONE — The finite difference zone is the smallest geometric domain within which the change in a phenomenon (e.g., stress versus strain, fluid flow or heat transfer) is evaluated. Quadrilateral zones are used in *FLAC*. Another term for zone is element. Internally, *FLAC* divides each zone into four triangular "subzones," but the user is not normally aware of these.

GRIDPOINT — Gridpoints are associated with the corners of the finite difference zones. There are always four (4) gridpoints associated with each zone. In the *FLAC* model, a pair of x- and y-coordinates are defined for each gridpoint, thus specifying the exact location of the finite difference zones. Other terms for gridpoint are nodal point and node.

FINITE DIFFERENCE GRID — The finite difference grid is an assemblage of one or more finite difference zones across the physical region which is being analyzed. Another term for grid is mesh.

MODEL BOUNDARY — The model boundary is the periphery of the finite difference grid. Internal boundaries (i.e., holes within the grid) are also model boundaries.

BOUNDARY CONDITION — A boundary condition is the prescription of a constraint or controlled condition along a model boundary (e.g., a fixed displacement or force for mechanical problems, an impermeable boundary for groundwater flow problems, adiabatic boundary for heat transfer problems, etc.).

INITIAL CONDITIONS — This is the state of all variables in the model (e.g., stresses or pore pressures) prior to any loading change or disturbance (e.g., excavation).

CONSTITUTIVE MODEL — The constitutive (or material) model represents the deformation and strength behavior prescribed to the zones in a *FLAC* model. Several constitutive models are available in *FLAC* to assimilate different types of behavior commonly associated with geologic materials. Constitutive models and material properties can be assigned individually to every zone in a *FLAC* model.

SUB-GRID — The finite difference grid can be divided into sub-grids. Sub-grids can be used to create regions of different shapes in the model (e.g., the dam sub-grid on the foundation sub-grid in Figure 2.27). Sub-grids cannot share the same gridpoints with other sub-grids; they must be separated by null zones.

NULL ZONE — Null zones are zones that represent voids (i.e., no material present) within the finite difference grid. All newly created zones are null by default.

ATTACHED GRIDPOINTS — Attached gridpoints are pairs of gridpoints that belong to separate sub-grids that are joined together. The dam is joined to the foundation along attached gridpoints in Figure 2.27. Attached gridpoints do not have to match between sub-grids, but sub-grids cannot separate from one another once attached.

INTERFACE — An interface is a connection between sub-grids that *can* separate (e.g., slide or open). An interface can represent a physical discontinuity such as a fault or contact plane. It can also be used to join sub-grid regions that have different zone sizes.

MARKED GRIDPOINTS — Marked gridpoints are specially designated gridpoints that delimit a region for the purpose of applying an initial condition, assigning material models and properties, and printing selected variables. The marking of gridpoints has no effect on the solution process.

REGION — A region in a *FLAC* model refers to all zones enclosed within a contiguous string of "marked" gridpoints. Regions are used to limit the range of certain *FLAC* commands, such as the **MODEL** command that assigns material models to designated regions.

***GROUP* —** A group in a *FLAC* model refers to a collection of zones identified by a unique name. Groups are used to limit the range of certain *FLAC* commands, such as the **MODEL** command that assigns material models to designated groups. Any command reference to a group name indicates that the command is to be executed on that group of zones.

***STRUCTURAL ELEMENT* —** Structural elements are linear elements used to represent the interaction of structures (such as tunnel liners, rock bolts, cable bolts or support props) with a soil or rock mass. Some restricted material nonlinearity is possible with structural elements. Geometric nonlinearity occurs in large-strain mode.

***STEP* —** Because *FLAC* is an explicit code, the solution to a problem requires a number of computational steps. During computational stepping, the information associated with the phenomenon under investigation is propagated across the zones in the finite difference grid. A certain number of steps is required to arrive at an equilibrium (or steady-flow) state for a static solution. Typical problems are solved within 2000 to 4000 steps, although large complex problems can require tens of thousands of steps to reach a steady state. When using the dynamic analysis option, **STEP** refers to the actual timestep for the dynamic problem. Other terms for step are timestep and cycle.

***STATIC SOLUTION* —** A static or quasi-static solution is reached in *FLAC* when the rate of change of kinetic energy in a model approaches a negligible value. This is accomplished by damping the equations of motion. At the static solution stage, the model will be either at a state of force equilibrium or at a state of steady-flow of material if a portion (or all) of the model is unstable (i.e., fails) under the applied loading conditions. This is the default calculation in *FLAC*.* Static mechanical solutions can be coupled to transient groundwater flow or heat transfer solutions. (As an option, fully dynamic analysis can also be performed by inhibiting the static solution damping.)

***UNBALANCED FORCE* —** The unbalanced force indicates when a mechanical equilibrium state (or the onset of plastic flow) is reached for a static analysis. A model is in exact equilibrium if the net nodal force vector at each gridpoint is zero. The maximum nodal force vector is monitored in *FLAC* and printed to the screen when the **STEP** or **SOLVE** command is invoked. The maximum nodal force vector is also called the *unbalanced* or *out-of-balance* force. The maximum unbalanced force will never exactly reach zero for a numerical analysis. The model is considered to be in equilibrium when the maximum unbalanced force is small compared to the total applied forces in the problem. If the unbalanced force approaches a constant nonzero value, this probably indicates that failure and plastic flow are occurring within the model.

---

* The mistaken notion exists in some finite element (FE) literature that a dynamic solution method cannot produce a true equilibrium state, compared to an FE solution, which is believed to satisfy perfectly the set of governing equations at equilibrium. In fact, *both* methods only satisfy the equations approximately, but the level of residual errors can be made as small as desired. In *FLAC*, the level of error is objectively quantified as the ratio of unbalanced force at a gridpoint to the mean of the set of absolute forces acting at the gridpoint. This measure of error is very similar to the convergence criteria used in FE solutions. In both cases the solution process is terminated when the error is below a desired value.

DYNAMIC SOLUTION — For a dynamic solution, the full dynamic equations of motion (including inertial terms) are solved; the generation and dissipation of kinetic energy directly affect the solution. Dynamic solutions are required for problems involving high frequency and short duration loads — e.g., seismic or explosive loading. The dynamic calculation is an optional module to *FLAC* (see Section 3 in **Optional Features**).

LARGE-STRAIN/SMALL-STRAIN — By default, *FLAC* operates in small-strain mode: that is, gridpoint coordinates are not changed, even if computed displacements are large (compared to typical zone sizes). In large-strain mode, gridpoint coordinates are updated at each step, according to computed displacements. In large-strain mode, geometric nonlinearity is possible.

2.4 The Finite Difference Grid

The finite difference grid spans the physical domain being analyzed. The smallest possible grid that can be analyzed with *FLAC* consists of only one zone. Most problems, however, are defined by grids that consist of hundreds or thousands of zones.

A grid is defined by specifying the number of zones "i" desired in the horizontal (x) direction, and the number of zones "j" in the vertical (y) direction. The grid is organized in a row-and-column fashion. Any zone in the grid is uniquely identified by a pair of i, j indices. Likewise, each gridpoint is uniquely identified by a pair of i, j indices. The i, j indices of the zones and gridpoints associated with the lower-left section of the grid shown in Figure 2.28 are presented in Figures 2.29(a) and (b). Note that if there are p zones in the x-direction and q zones in the y-direction, then there are $p + 1$ gridpoints in the x-direction and $q + 1$ gridpoints in the y-direction.

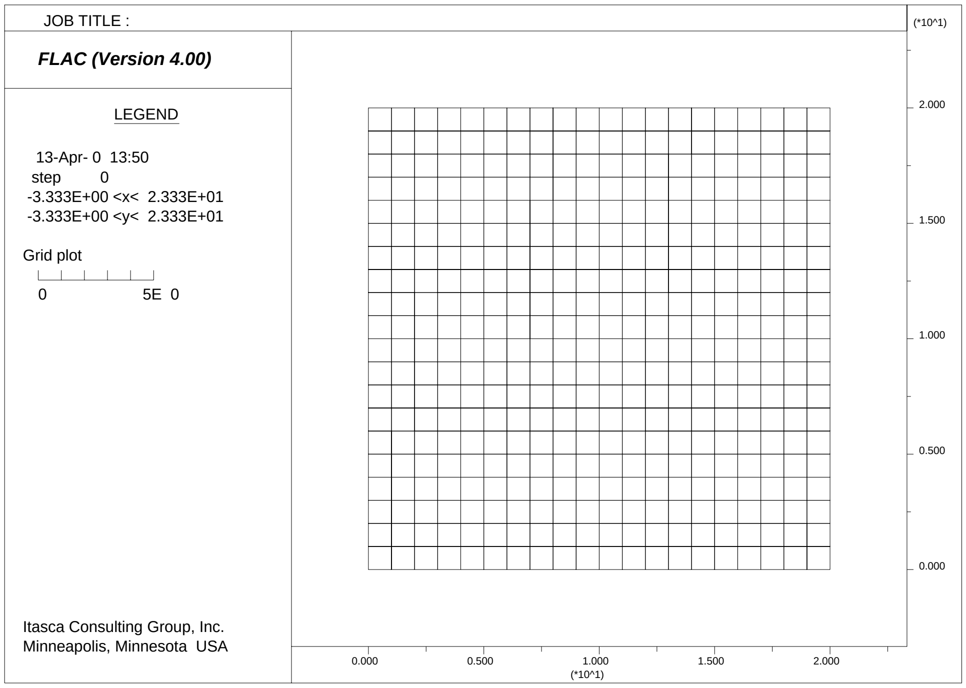
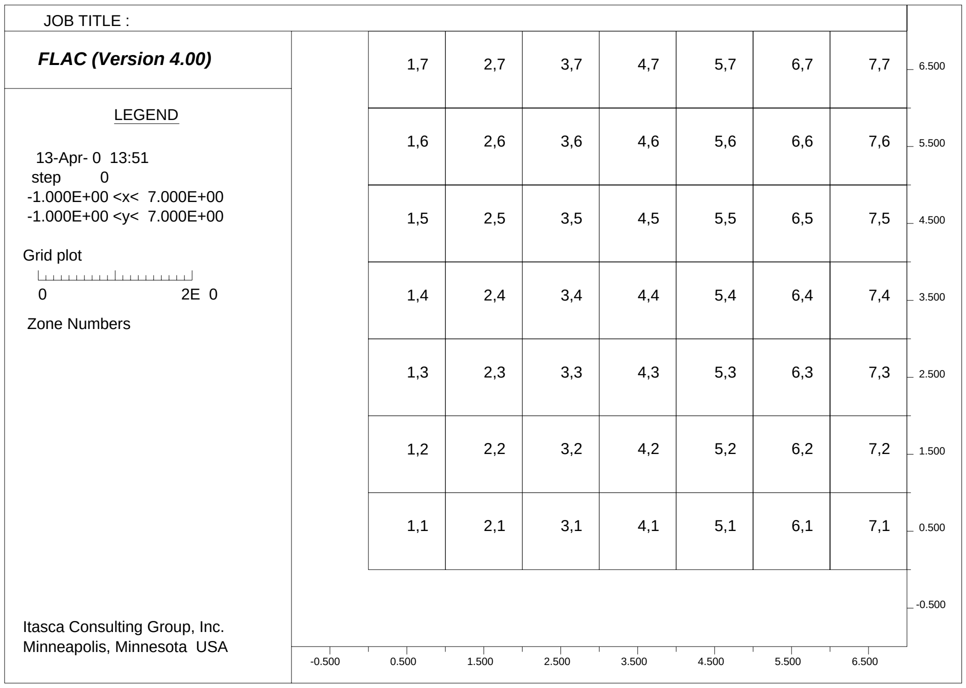
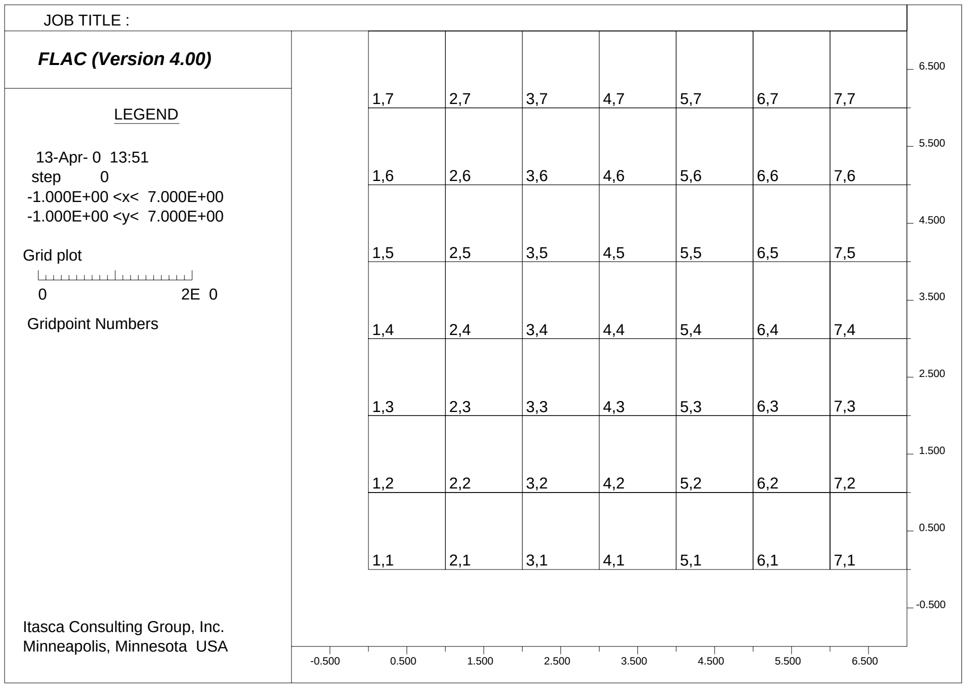


Figure 2.28 Finite difference grid with 400 zones



(a) zone numbers



(b) gridpoint numbers

Figure 2.29 Identification of zone and gridpoint (i, j) indices

In normal operation, the finite difference mesh origin is the lower left-hand corner of the grid. By default, the x-coordinates of the gridpoints are 0, 1, . . . , p, and the y-coordinates are 0, 1, . . . , q. The coordinates are indicated by the scales shown on the plots in Figures 2.28 and 2.29.

Grid generation with *FLAC* involves the shaping of the row-and-column grid to fit the shape of the physical domain. Grid generation is described in Section 3.2.

The finite difference grid also identifies the storage location of all state variables in the model. The procedure followed in *FLAC* is that all vector quantities (e.g., forces, velocities, displacements, flow rates) are stored at gridpoint locations, while all scalar and tensor quantities (e.g., stresses, pressure, material properties) are stored at zone centroid locations. There are three exceptions: saturation and temperature are considered gridpoint variables; and pore pressure is stored at both gridpoint and zone centroid locations.

2.5 Command Syntax

All input commands* to *FLAC* are word-oriented and consist of a primary command word followed by one or more keywords and values, as required. Some commands accept switches — that is, keywords that modify the action of the command. Each command has the following format:

COMMAND **key**word ***value*** . . . <**key**word ***value*** . . . > . . .

Here, optional parameters are denoted by < >, while the ellipses (. . .) indicate that an arbitrary number of such parameters may be given. The commands are typed literally on the command line. You will note that only the first few letters are in bold type. The program requires these letters, at a minimum, to be typed to recognize the command; command input is not case-sensitive. The entire word for commands and keywords may be entered if the user so desires.

Many of the keywords are followed by a series of values which provide the numeric input required by the keyword. The decimal point may be omitted from a real value, but may not appear in an integer value.

Commands, keywords and numeric values may be separated by any number of spaces or by any of the following delimiters:

( ) , =

A semicolon (;) may be used to precede comments; anything that follows a semicolon in an input line is ignored. It is useful, and strongly recommended, to include comments in data files. Not only is the input documented in this way, the comments are echoed to the output as well, providing the opportunity for quality assurance in your analysis.

A single input line, including comments, may contain up to 80 characters.

If more than 80 characters are required to describe a particular command sequence, then an ampersand (&) can be given at the end of an input line to denote that the next line will be a continuation of that line. The maximum length of a single command, including continuations, is 2000 characters. A maximum of 400 input parameters are allowed in one command. A total of 1024 characters per command sequence are allowed.

* The commands and their meanings are presented in Section 1.3 in the **Command Reference**; a summary is given in Section 1 in the **Command and *FISH* Reference Summary**.

2.6 Mechanics of Using *FLAC*

This section provides an introduction to the basic commands a new user needs to perform simple *FLAC* calculations in command-driven mode. If you have not done so already, run the tutorial problem in Section 2.2.1.2 for an example of a command-driven analysis with *FLAC*.

All the commands in *FLAC* can be accessed from the graphical interface. We recommend that you use the *GIIC* (see the introduction in Section 2.2.2) for ease of operation while learning the mechanics of using *FLAC*. You can follow the examples in this section either by entering the word commands at the `flac:` prompt in the text mode, or by point-and-click operation in the graphical mode. In the latter case, the commands will be created by the *GIIC* for you to check as you follow the example.

All the example data files for this section are listed in the "\FLAC\Tutorial\Beginner" directory. These can be read into *FLAC* either by using the **CALL** command in the text mode, or via the CALL button in the RUN modeling tool of the *GIIC*.

In order to set up a model to run a simulation with *FLAC*, three fundamental components of a problem must be specified:

(1) a finite difference grid;

(2) constitutive behavior and material properties; and

(3) boundary and initial conditions.

The grid defines the geometry of the problem. The constitutive behavior and associated material properties dictate the type of response the model will display upon disturbance (e.g., deformation response due to excavation). Boundary and initial conditions define the in-situ state (i.e., the condition before a change or disturbance in the problem state is introduced).

After these conditions are defined in *FLAC*, the initial equilibrium state is calculated for the model. An alteration is then made (e.g., excavate material or change boundary conditions), and the resulting response of the model is calculated. The actual solution of the problem is different for an explicit finite difference program like *FLAC* than it is for conventional implicit-solution programs. (See the background discussion in Section 1 in **Theory and Background**.) *FLAC* uses an explicit time-marching method to solve the algebraic equations. The solution is reached after a series of computational steps. In *FLAC*, the number of steps required to reach a solution can be controlled automatically by the code *or* manually by the user. However, the user ultimately must determine if the number of steps is sufficient to reach the solved state. The way this is done will be covered later in Section 2.6.4.

The general solution procedure, illustrated in Figure 2.30, is convenient because it represents the sequence of processes that occurs in the physical environment. The basic *FLAC* commands needed to perform simple analyses with this solution procedure are described below.

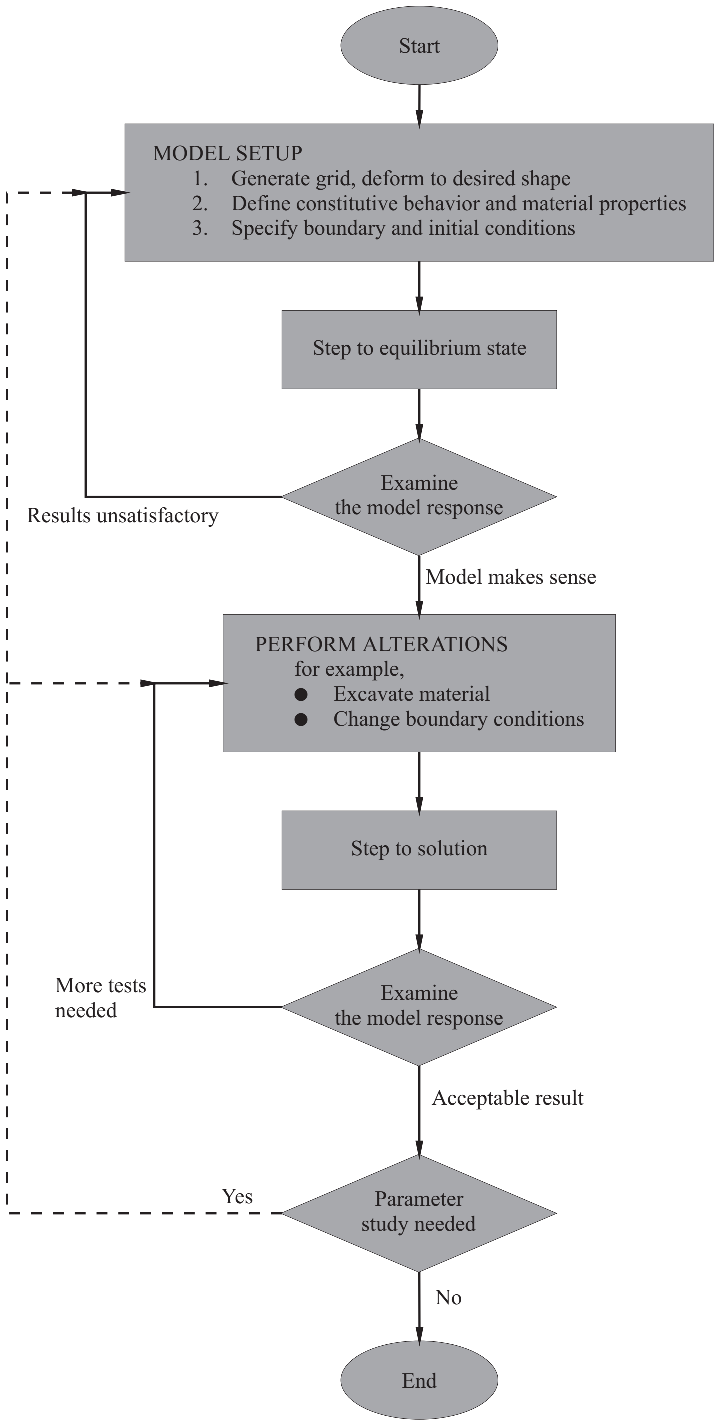

Start
MODEL SETUP
1. Generate grid, deform to desired shape
2. Define constitutive behavior and material properties
3. Specify boundary and initial conditions
Step to equilibrium state
Examine the model response
Results unsatisfactory
Model makes sense
PERFORM ALTERATIONS
for example,
● Excavate material
● Change boundary conditions
Step to solution
Examine the model response
More tests needed
Acceptable result
Parameter study needed
Yes
No
End

Figure 2.30 General solution procedure

2.6.1 Grid Generation

The first input command that must be given to generate a grid is

```
grid  icol jrow
```

where ***icol*** is the number of columns of zones, and ***jrow*** is the number of rows of zones in the mesh. Be careful when selecting the number of zones for a model, because a balance must be struck between the accuracy required and the solution speed. The calculation speed to reach a solution varies directly as a function of the number of elements. As a rule of thumb, models containing up to roughly 5000 elements will typically reach a solution state for a given alteration in approximately 2000 to 4000 steps. On a 300 MHz Pentium II microcomputer, the runtime for a 5000-element model to perform 4000 steps is roughly 3 minutes. Check the speed of calculation on your computer for the specific model to estimate the runtime required. A runtime benchmark test is provided in Section 5.1.

It is best to start with a grid that has few zones (say, 100 to 500) to perform simple test runs and make refinements to the model. Then, increase the number of zones to improve the accuracy.

Two commands are used in *FLAC* to shape the grid:

```
generate
initial
```

The **GENERATE** command creates regions of different shapes within the grid. The **INITIAL** command changes the x- and y-coordinates of selected gridpoints. The complete descriptions for these commands are given in Section 1.3 in the **Command Reference**. The following examples illustrate their use.

Example 1 — In its simplest form, the **GENERATE** command can supply new coordinates to a grid. By entering the commands* in Example 2.2, a square grid of 10 zones by 10 zones (11 gridpoints by 11 gridpoints) will be created, and each zone will be assigned the elastic material model.

Example 2.2 Generating a simple grid

```
grid  10 10
model elastic
```

If the coordinates of the grid are printed at this stage, by issuing the command

```
print x y
```

* If you want to try entering the command examples interactively from the text mode, type **NEW** each time you start a new example. In the *GIIC*, press the FILE/NEW PROJECT item in the main menu. This will initialize *FLAC* without having to exit and reload the program for a new model. In text mode, type **PLOT grid** after entering each example to view the result. The model view will be displayed automatically in the *GIIC*.

you will see that both x and y run from 0.0 to 10.0 (i.e., *FLAC* assigns a square grid with 1 unit spacing between gridpoints). Note that the **MODEL** command *must* be issued before the **PRINT** command, otherwise the grid coordinates will not be displayed. This is also true for the **PLOT** command. There must be a material present for information to be printed or plotted.

If the actual coordinates of the grid are to run from 0.0 to 500.0 in the x-direction and from 0.0 to 1000.0 in the y-direction, the **GENERATE** command is issued as follows:

```
gen  0,0 0,1000 500,1000 500,0  i=1,11 j=1,11
```

Note that the four corner coordinates for the portion of the mesh defined by i = 1,11, j = 1,11 start at the lower left-hand corner of the grid and work around its outer corners in a clockwise fashion. All gridpoints interior to these corner points will have their coordinates reassigned based on the corner point coordinates. Now, print out the coordinates again to see that the coordinates have indeed been changed. Note that just a portion of the grid can be given new coordinates. The portion of the grid is defined by the i, j range (see Example 2.3). The corner coordinates must be specified in a *clockwise* fashion.

Example 2 — The **GENERATE** command can be used to create distortions in the grid. For example, try the commands in Example 2.3.

Example 2.3 Distorting the grid

```
new
grid 20,20
model elas
gen  0,5 0,20 20,20 5,5 i=1,11
gen  same same 20,0 5,0 i=11,21
plot hold grid
```

In this example, only a portion of the grid is distorted with each **GENERATE** command. The first **GENERATE** command creates a distorted quadrilateral from half of the grid, while the second command "wraps" the remainder of the grid around to form a rectangular opening. Successive **GENERATE** commands are additive — i.e., once changed, the coordinates of the grid remain at the new coordinates until changed again by using the **GENERATE** or **INITIAL** commands. In the second **GENERATE** command, the word **same** is used twice, which indicates that coordinates for the first two corner points are not changed. When you type **PLOT grid**, the distorted grid shape should be displayed.

Example 3 — The **GENERATE** command can be used to grade a mesh to represent far boundaries. For example, in many cases, an excavation is to be created at a great depth in a rock mass. Detailed information on the stresses and displacements is to be determined around the excavation, where the disturbance is large, but little detail is necessary at greater distances. In the following example, the lower left-hand portion of the grid is left finely discretized, and the boundaries are graded outward in the *x*- and *y*-directions. Try issuing the commands in Example 2.4.

Example 2.4 Grading the mesh

```
new
grid  20,20
m e
gen  0,0 0,100 100,100 100,0 rat 1.25 1.25
plot hold grid
```

The **GENERATE** command forces the grid lines to expand to 100.0 units at a rate 1.25 times the previous grid spacing in the *x*- and *y*-directions. (Example 2.4 also illustrates that command words can be truncated: **MODEL elas** becomes **M e**.) Note that if the ratio entered on the **GEN** command is between 0 and 1, the grid dimensions will decrease with increasing coordinate value. For example, issue the commands in Example 2.5.

Example 2.5 Applying different gradients to a mesh

```
new
gr  10,10
m e
gen  -100,0 -100,100 0,100 0,0 rat .80,1.25
plot hold grid
```

You will see a grid graded in the negative *x*- and positive *y*-directions.

Example 4 — Excavations often need to be created in the grid. It is very tedious to create complex excavation shapes, especially circular arcs, by simply moving individual gridpoints. Special shape functions are built into the **GENERATE** command (e.g., circles, arcs and lines). An example is given here for the creation of excavation shapes using the **GENERATE** command.

First, a circular excavation is created. Try the commands in Example 2.6.

Example 2.6 Creating a circular hole in a grid

```
new
grid  20,20
m e
gen  circle 10,10 5
```

```
model null region 10,10
plot hold grid
```

This command automatically creates a circular opening within the grid, centered at (x = 10, y = 10) with a radius of 5.0. Note that the remainder of the mesh remains square (i.e., element corners are at 90 degrees). Note, too, that the **MODEL** command *must* be specified first in order for the shape functions (circle, arc, line) to work.

To cause the mesh to conform better to the new opening, type

```
gen adjust
plot grid
```

Successive **GENERATE adjust** commands will smooth the grid to increasingly greater levels.

When creating internal shapes within the grid using the **GENERATE circle**, **GENERATE arc** or **GENERATE line** commands, *FLAC* distinguishes between the various regions of the grid created by marking closed paths. In the previous example, the **GENERATE circle** command creates two regions within the grid created by the boundary of the circle: the region inside the boundary and that outside. If you wish to see where the boundaries of the grid are, type

```
print mark
```

Those gridpoints which have been adjusted by *FLAC* to conform to boundaries are signified by an "M" in the printout.

CAUTION: Two regions can only be formed if they are separated by *closed* contours. In other words, a line segment which begins and ends within the grid and does not form a closed boundary, subsequently will result in only one region.

Example 5 — The **INITIAL** command can be used to move a point or a number of points from the present location to a new one. The following commands in Example 2.7 create a grid and distort it using the **INITIAL** command.

Example 2.7 Moving gridpoints with the* INITIAL *command

```
new
grid  5 5
model  elastic
gen  0,0 0,10 10,10 10,0
ini  x=-2 i=1 j=6
ini  x=12 i=6
plot hold grid
```

The **GENERATE** command assigns coordinates to gridpoints from 0 to 10 in the x- and y-directions. The first **INITIAL** command moves the upper left-hand corner horizontally by -2 units. The second

INITIAL command moves the right-hand boundary gridpoints to the right by 2 units. Note that since the j-range is not given, the entire range is assumed.

The **INITIAL** command can be used to move any gridpoint to any position. Of course, elements cannot overlap. If this happens, a warning message referring to "BAD GEOMETRY" will be given, and *FLAC* will not continue execution until the errors in grid construction are rectified.* A practical limit on the aspect ratio of zones should be kept to about 1:10 or less for reasonable solution accuracy.

During model solution, a quadrilateral may be deformed in any fashion, subject to the following criteria:

(1) the area of the quadrilateral must be positive; and

(2) each member of at least one pair of triangular sub-zones which comprise the quadrilateral must have an area greater than 20% of the total quadrilateral area (see Section 1.3.2 in **Theory and Background**).

These criteria should be applied when creating zones to avoid bad geometry during model solution. If either of these criteria is not met, *FLAC* will give a "BAD GEOMETRY" error message during timestepping. Figure 2.31 illustrates possible zone deformations.

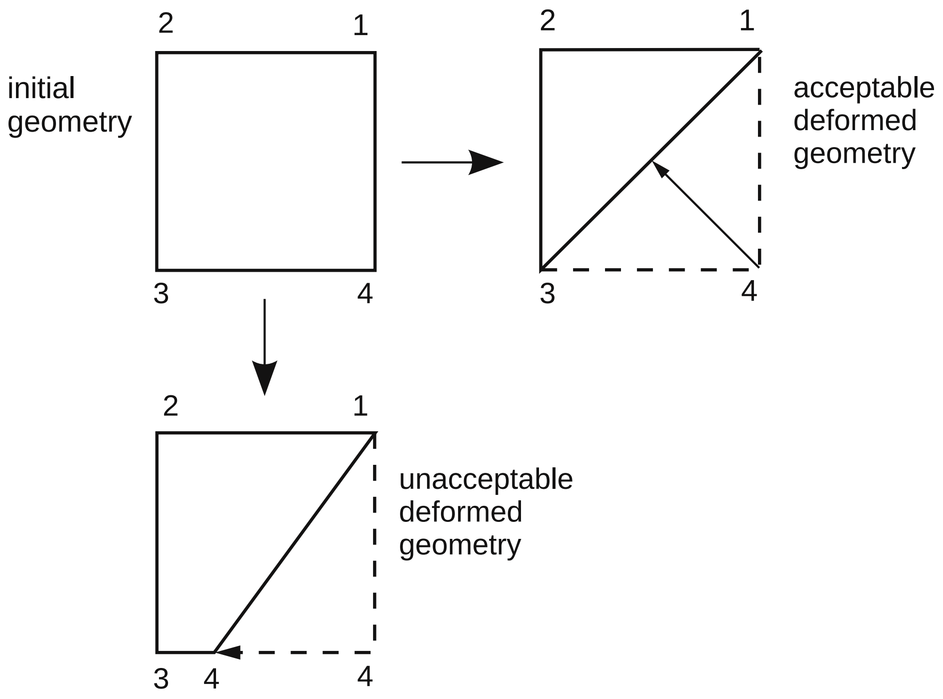


Figure 2.31 Acceptable and unacceptable zone deformations

* Note that in the *GIIC* you can check on whether a bad geometry condition has been created prior to calculation by clicking on the BAD ZONE GEOMETRY item in the DRAW menu of the *Model*-view pane.

WARNING: All grid shaping to create holes or new boundaries (e.g., slope faces) that will be removed, or excavated, at a later state in the solution must be performed *before* the computational stepping begins. The **GENERATE** and **INITIAL** commands should *not* be used to adjust the grid after the **STEP** or **SOLVE** command is issued. (These commands are described below.) This adjustment can introduce an erroneous calculation for gridpoint masses in the model. If it is necessary to move gridpoints after stepping has begun, a velocity can be applied to the gridpoint for a specified number of steps to move the required displacement.

2.6.2 Assigning Material Models

Once the grid generation is complete, one or more material models and associated properties must be assigned to all zones in the model. This is done by using two commands: **MODEL** and **PROPERTY**. *FLAC* has ten (10) built-in material models; these are described in Section 2 in **Theory and Background**. Three models are sufficient for most analyses the new user will make. These are **MODEL null**, **MODEL elastic** and **MODEL mohr**.

MODEL null represents material which is removed or excavated from the model. **MODEL elastic** assigns isotropic elastic material behavior, and **MODEL mohr** assigns Mohr-Coulomb plasticity behavior.

MODEL elastic and **MODEL mohr** require that material properties be assigned via the **PROPERTY** command. For the elastic model, the required properties are:

(1) density;

(2) bulk modulus; and

(3) shear modulus.

NOTE: Bulk modulus, K, and shear modulus, G, are related to Young's modulus, E, and Poisson's ratio, ν, by

$$K = \frac{E}{3(1-2\nu)} \tag{2.1}$$

$$G = \frac{E}{2(1+\nu)} \tag{2.2}$$

or

$$E = \frac{9KG}{3K+G} \tag{2.3}$$

$$\nu = \frac{3K-2G}{2(3K+G)} \tag{2.4}$$

For the Mohr-Coulomb plasticity model, the required properties are:

(1) density;

(2) bulk modulus;

(3) shear modulus;

(4) friction angle;

(5) cohesion;

(6) dilation angle; and

(7) tensile strength.

If any of these properties are not assigned, their values are set to zero by default.

For example, an elastic model may be prescribed for the upper half of a 10 × 10 grid and a Mohr-Coulomb model for the lower half. Example 2.8 shows how this is done.

Example 2.8 Assigning different material models in different areas of a grid

```
new
grid  10,10
model elas j=6,10
prop  den=2000 bulk=1e8 shear=.3e8 j=6,10
model mohr j=1,5
prop  den=2500 bulk=1.5e8 shear=.6e8 j=1,5
prop  fric=30 coh=5e6 ten=8.66e6 j=1,5
plot  hold  model
```

Instead of using i, j indices to specify a range, the word (i.e., keyword) **region** can be used. For example, to excavate the circular tunnel in Example 2.6, the following command can be applied:

```
model  null region 10,10
```

By specifying one zone inside the marked tunnel region (e.g., zone i = 10, j = 10), then all zones within the tunnel are set to null material (i.e., excavated).

The tunnel can be filled at a later stage by typing, for example,

```
model  elas region 10,10
```

Note that the excavation can be replaced by any model, and with properties consistent with the model.

2.6.3 Applying Boundary and Initial Conditions

After the grid is generated, boundary and initial conditions are applied. These conditions can be specified in *FLAC* by means of the commands **APPLY**, **INITIAL**, and **FIX** or **FREE**. Table 2.3 provides a summary of the boundary condition commands and their effects. Table 2.4 provides a similar summary for initial condition commands. Note that by using the boundary condition commands, a condition or constraint will be imposed that will not change (unless specifically changed by the user) while *FLAC* is calculating a solution. By using the initial condition commands, *initial* values are assigned to variables; these *can* change while the computation proceeds.

Table 2.3 Boundary condition command summary

Command		**Effect**
APPLY	**pressure**	mechanical pressure (*not* pore pressure) applied at boundary
	sxx	xx-component of total stress tensor applied at boundary
	sxy	xy-component of total stress tensor applied at boundary
	syy	yy-component of total stress tensor applied at boundary
	xforce	x-component of force applied at boundary gridpoints
	yforce	y-component of force applied at boundary gridpoints
	xvel	x-velocity applied at boundary gridpoints
	yvel	y-velocity applied at boundary gridpoints
FIX	**pp**	pore pressure fixed at boundary gridpoints
	x	x-velocity fixed at boundary gridpoints
	y	y-velocity fixed at boundary gridpoints

NOTE:

1. The **FREE** command is used to release the constraint set by the **FIX** command.
2. In order to assign a fixed-displacement boundary condition, only the **FIX x** and/or **FIX y** commands are needed, provided that the velocity at the selected gridpoint is zero.
3. See Section 1.3 in the **Command Reference** for a complete listing of **APPLY** and **FIX** keywords.

Table 2.4 Initial condition command summary

Command		Effect
INITIAL	**pp**	initialize pore pressure for a zone*
	sat	initialize saturation at a gridpoint
	sxx	initialize xx-component of total stress for a zone
	sxy	initialize xy-component of total stress for a zone
	syy	initialize yy-component of total stress for a zone
	szz	initialize zz-component of total stress for a zone
	xvel	initialize x-velocity at a gridpoint
	yvel	initialize y-velocity at a gridpoint
	xdis	initialize x-displacement at a gridpoint
	ydis	initialize y-displacement at a gridpoint

* Note that when running a groundwater flow analysis (by specifying **CONFIG gw** — see Section 1 in **Fluid-Mechanical Interaction**), pore pressure is initialized at *gridpoints*. Zone pore pressures are then derived by averaging.

Example 2.9 illustrates the application of boundary and initial conditions.

Example 2.9 Applying boundary and initial conditions

```
new
grid  10 10
mod  el
fix  x i=1
fix  x i=11
fix  y j=1
app  press = 10 j=11
ini  sxx=-10 syy=-10
plot hold bou fix apply stress
```

The grid has the left- and right-hand sides fixed from movement in the x-direction, and the bottom fixed in the y-direction. A pressure is applied to the top boundary, and all zones in the model have an initial stress: $\sigma_{xx} = \sigma_{yy} = -10$. In *FLAC*, compressive stresses have a negative sign, while compressive pressure is positive. All these conditions are displayed with the **PLOT** command. The applied pressure is displayed as force vectors. The stresses are shown as principal stress tensors. The boundary of the grid is also shown.

2.6.4 Stepping to Initial Equilibrium

The *FLAC* model must be at an initial force-equilibrium state before alterations can be performed. The boundary conditions and initial conditions may often be assigned such that the model is exactly at equilibrium initially. However, it may be necessary to calculate the initial equilibrium state under the given boundary and initial conditions, particularly for problems with complex geometries or multiple materials. This is done by using either the **STEP** or **SOLVE** command. With the **STEP** command, the user specifies a number of calculation steps to perform in order to bring the model to equilibrium. The model is in equilibrium when the net nodal force vector at each gridpoint is close to zero (see Section 1.3.3.5 in **Theory and Background**).

Two different values are printed to the screen during model solution:

1. the maximum nodal force vector (called the maximum *out-of-balance* or *unbalanced* force); and
2. the largest ratio of maximum unbalanced force to average applied force amongst all of the gridpoints (called the *equilibrium ratio*).

Using one or both of these numbers as a guide, the user can assess when equilibrium has been reached. It is important to realize that for a numerical analysis, the out-of-balance force will never reach exactly zero. It is sufficient, though, to say that the model is in equilibrium when the maximum unbalanced force is small compared to the applied forces in the problem. Therefore, a value of 1% or 0.1% for the equilibrium ratio may be acceptable as denoting equilibrium, depending on the degree of precision required.

This is an important aspect of numerical problem solving with *FLAC*. *The user must decide when the model has reached equilibrium.* There are several features built into *FLAC* to assist with this decision. The history of the maximum unbalanced force may be recorded with the following command:

```
hist  unbal
```

Additionally, the history of selected variables (e.g., velocity or displacement at a gridpoint) may be recorded. The following commands are examples:

```
hist  xvel i=5 j=5
hist  ydisp i=5 j=11
```

The first history records x-velocity at gridpoint (5,5), while the second records y-displacement at gridpoint (5,11).

After running several hundred (or thousand) calculation steps, a history of these records may be plotted to indicate the equilibrium condition. The data file in Example 2.10 illustrates this process.

Example 2.10 Stepping to initial equilibrium

```
new
grid  10 10
mod  el
prop  d=1800 bulk=1e8 shear =.3e8
fix  x i=1
fix  x i=11
fix  y j=1
app  pres=1e6 j=11
hist  unbal
hist  ydisp i=5 j=11
step 900
```

The initial maximum unbalanced force is 1 MN. After 900 steps, this force has dropped to approximately 100 N. By plotting the two histories, it can be seen that the maximum unbalanced force has approached zero, while the displacement has approached a constant magnitude of approximately 0.07 m.

Type

```
plot hold hist 1
plot hold hist 2
```

to view these plots. The number following **PLOT hist** corresponds to the order in which the histories are entered in the data file. Figures 2.32 and 2.33 show the unbalanced force and displacement history plots.

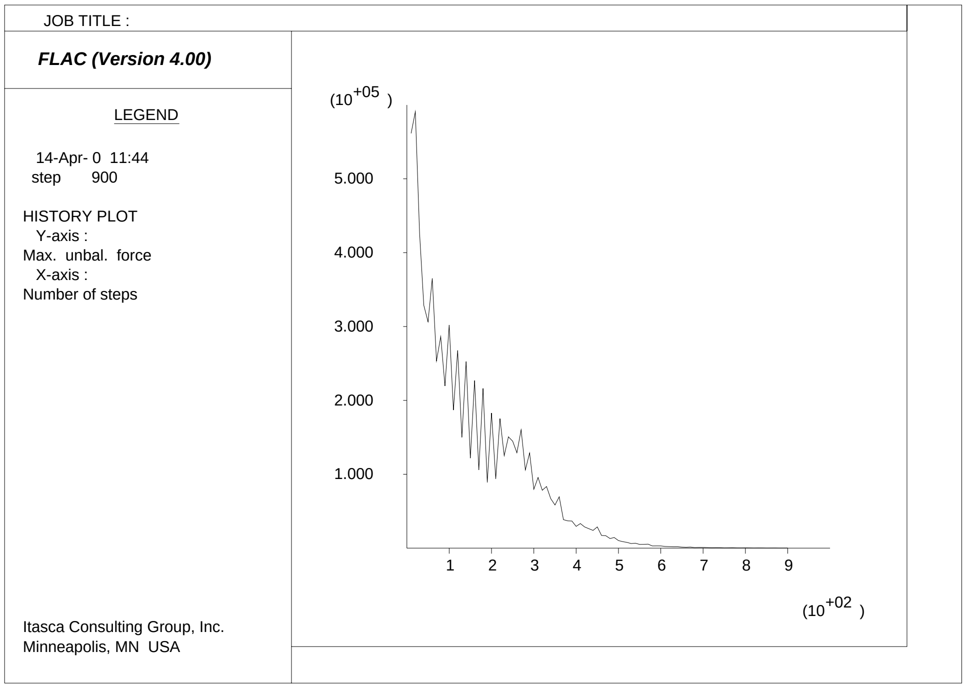


Figure 2.32 Maximum unbalanced force history

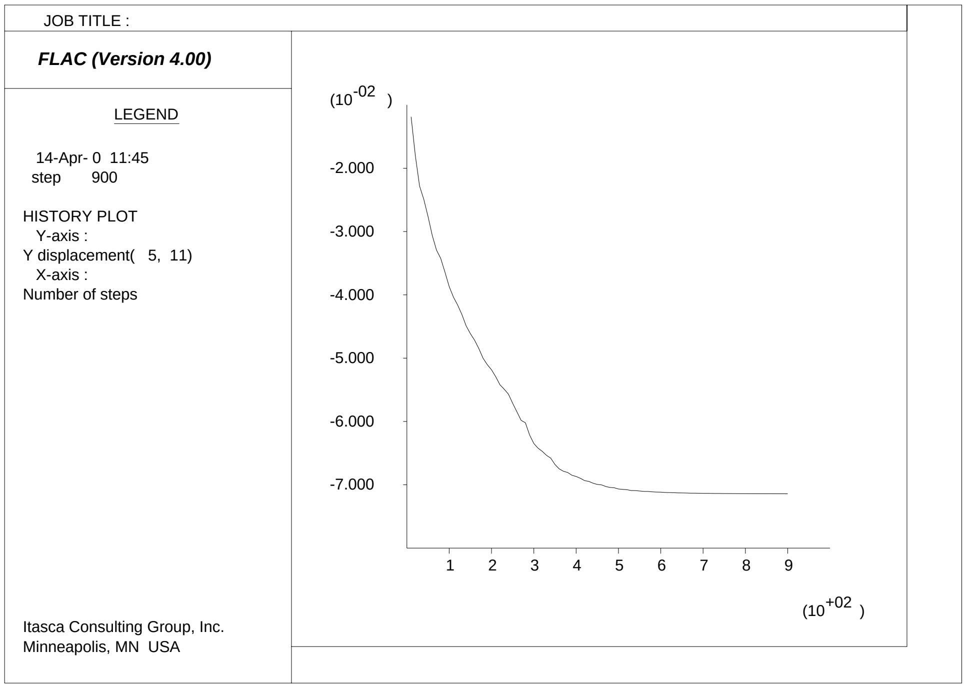


Figure 2.33 y-displacement history of gridpoint 5,11

Normally, displacements are initialized to zero at the initial equilibrium stage. This can be done now by typing

```
ini  xdis=0 ydis=0
```

Type

```
print  xdis ydis
```

to confirm this.

The **SOLVE** command can be used in place of **STEP** if the user wishes *FLAC* to stop automatically when the maximum unbalanced force or equilibrium ratio falls below a specified limit. Replace **STEP 900** with **SOLVE** and repeat the above problem. This time, *FLAC* should stop the calculation at step 664. If the plots are made again, essentially the same results as given in Figures 2.32 and 2.33 will be seen.

The **SOLVE** command is controlled by a limiting equilibrium ratio (10^{-3}), a limiting unbalanced force (100 force units), a limiting number of timesteps (100,000 steps), and a limiting computer runtime (1440 minutes), where the default values are given in parentheses. The calculation will stop when any one of these limits is reached. In the above example, the equilibrium ratio of 10^{-3} is reached first. In order for the unbalanced force to control stepping, the command **SET sratio=0** should be given before the **SOLVE** command. Now, the calculation will stop at a force limit of 100. Each of the solving limits can be changed with the **SET** command. For example, **SET force=50** will change the unbalanced force limit to 50. (Alternatively, the **SOLVE force=50** command can be given.) The limit will remain in effect until changed again, or until a **NEW** command is issued, which will reset the limits to their default values. When using the **SOLVE** command, it is important to check that the calculation does not stop prematurely (e.g., if the calculation is expected to take more than 100,000 steps to reach equilibrium, then the **SET step** command should be used to increase the step limit).

For the above example, an initial equilibrium stage can be achieved without stepping by simply inserting an **INITIAL** command:

```
ini  sxx=-1e6 syy=-1e6 szz=-1e6
```

Now, the unbalanced force is exactly zero. Type **SOLVE** to confirm this. Note that, in this case, the initial displacements in the model are automatically zero. Note also that, in this example, any initial value for **sxx** or **szz** will give an initial equilibrium.

If the initial stage is subjected to gravitational loading, this may be added via

```
set  gravity=9.81
```

where a gravitational acceleration of 9.81 m/sec^2 is applied in the negative y-direction. If the above problem is continued with gravity loading, a maximum unbalanced force of approximately 18,000 N develops, and 720 steps are required (using **SOLVE**) to bring the model back to equilibrium. There is a stress gradient now in the model which can be viewed by typing

```
print  syy
```

The values for σ_{yy} range from 1 MPa in the top zones of the model, to 1.166 MPa in the bottom zones. There is also a gradient in the x-direction z-direction; type

```
print  sxx szz
```

In an analysis, it is very important that the model be at equilibrium before alterations are made. Several histories should be recorded throughout a model to ensure that a large force imbalance does not exist. It does not affect the analysis adversely if more steps than needed are taken to reach equilibrium, but it will affect the analysis if an insufficient number of steps are taken.

A *FLAC* calculation can be interrupted at any time during stepping by pressing the <Esc> key. It often is convenient to use the **STEP** command with a high step number and periodically interrupt the stepping, check the histories, and resume stepping (with **STEP continue**) until the equilibrium condition is reached.

2.6.5 Performing Alterations

FLAC allows model conditions to be changed at any point in the solution process. These changes may be of the following form:

- excavation of material;
- addition or deletion of gridpoint loads or pressures;
- change of material model or properties for any zone; and
- fix or free velocities for any gridpoint.

Excavation is performed with the **MODEL null** command. Gridpoint loads can be applied at any gridpoint with the **APPLY xforce** and **APPLY yforce** commands. Pressure or stress alterations can be made at model boundaries with the **APPLY** command, as discussed previously. Material models and properties are changed with the **MODEL** and **PROPERTY** commands. Gridpoint velocities are fixed or freed via the **FIX**/**FREE** commands. It should be evident that several commands can be repeated to perform various model alterations.

Try the data file in Example 2.11.

Example 2.11 Excavating a tunnel and monitoring the response

```
grid  10,10
model elastic
gen   circle 5,5 2
plot  hold  grid
gen   adjust
plot  hold  grid
prop  s=.3e8 b=1e8 d=1600
set   grav=9.81
fix   x i=1
fix   x i=11
fix   y j=1
solve
pr    mark
ini sxx  0.0 syy 0.0 szz 0.0 region 5,5
prop s   .3e5 b 1e5 d 1.6    region 5,5
;mod   null region 5,5
plot  hold  grid
pause
solve
plot  hold  str bou
```

This problem illustrates the alteration on stress distribution due to excavation of a circular tunnel in an elastic material. Because the grid cannot be altered after stepping begins, it must be deformed to fit the boundaries of the tunnel before the initial stresses are equilibrated. After excavation (i.e., **MODEL null**), an unbalanced force results, and the model is stepped to equilibrium again. The plot of principal stress tensors shows the stress distribution resulting from the excavation.

If model zones contain a plasticity material model (e.g., **MODEL mohr**), it is possible that an alteration may be such that force equilibrium cannot be achieved. In other words, the unbalanced forces in part or all of the model cannot approach zero — in which case, the maximum unbalanced force will approach a constant nonzero value, indicating that steady-state flow of material is occurring (i.e., a portion, or all, of the model is failing).

Example 2.12 illustrates model failure.

Example 2.12 Excavate and fill in stages

```
grid  10,10
m  e
prop  s=5.7e9 b=11.1e9 d=2000
fix  x i=1
fix  y j=1
fix  x i=11
apply syy -20e6 j=11
ini  sxx -30e6 syy -20e6 szz -20e6
his  unbal
his  xdis i=4 j=5
solve
mod  null i 4,7 j 3,6
plot  hold  grid
solve
plo  hold  his 1
plo  hold  his 2
plo  hold  grid str
mod  mohr i 4,7 j 3,6
```

```
prop  s=.3e8 b=1e8 fric=30 i=4,7 j=3,6
mod  null i=1,3 j=3,6
mod  null i=8,10 j=3,6
ini  xd=0 yd=0
his  reset
his  unbal
his  xdis i=4 j=5
step 1000
plot  hold  his -2
plot  hold  xdis fill zero bou
```

This is a simple analysis of cut-and-fill mining, where excavations are created and backfilled sequentially. The boundaries are too close for an accurate solution, but the simulation illustrates *FLAC*’s ability to change model conditions and calculate the results — in this case, the backfill fails upon excavation of the adjacent cuts. The region of failure is indicated by the x-displacement contour plot. The history plot shows that the gridpoint (4,5) in the backfill zone is at a constantly increasing steady-state displacement.

2.6.6 Saving/Restoring Problem State

Two other commands, **SAVE** and **RESTORE**, are helpful when performing analyses in stages. At the end of one stage (e.g., initial equilibrium), the model state can be saved by typing

```
save  file.sav
```

where file is a user-specified filename. The extension “.SAV” identifies this file as a saved file (see Section 2.10). This file can be restored at a later time by typing

```
rest  file.sav
```

and the model state at the point at which the model was saved will be restored. It is not necessary to build the model from scratch every time a change is made; merely save the model before the change and restore it whenever a new change is to be analyzed. For example, in the previous example, the state should be saved after the initial equilibrium stage. Then, the effect of different backfill properties can be evaluated by restoring this file, changing the properties, and calculating the result. For example, insert the following

```
save  fill1.sav
```

after the **MODEL mohr** command. Then create a data file of the form shown in Example 2.13 to study the influence of the backfill.

Example 2.13 A parametric study

```
rest  fill1.sav
prop  _ _ _ _ _ (first set of fill properties)
mod null i=1,3  j=3,6
mod null i=8,10 j=3,6
step  1000
save  fill2.sav
rest  fill1.sav
prop  _ _ _ _ _ (second set of fill properties)
mod null i=1,3  j=3,6
mod null i=8,10 j=3,6
step  1000
save  fill3.sav
rest  fill1.sav
prop  _ _ _ _ _ (third set of fill properties)
mod null i=1,3  j=3,6
mod null i=8,10 j=3,6
step 1000
save  fill4.sav
.
.
.
```

This file should be created with a text editor and called into *FLAC*. After the run is completed, the saved files can be restored and evaluated separately to study the effect of the backfill properties.

2.6.7 Summary of Commands for Simple Analyses

The major command words described in Section 2.6 are summarized in Table 2.5. These are all that are needed to begin performing simple analyses with *FLAC*. Start by running simple tests with these commands (e.g., uniaxial and confined compression tests or simple excavation stability analyses). It may be helpful to review the detailed description of these commands in Section 1.3 in the **Command Reference**. Then try adding more complexity to the model. Before running very detailed simulations though, we recommend that you read Section 3, which provides guidance on problem solving in general.

Table 2.5 Basic commands for simple analyses

Function	**Command**
Grid Generation	**GRID** **GEN** **INITIAL**
Boundary/Initial Conditions	**APPLY** **FIX** **INITIAL**
Material Model & Properties	**MODEL** **PROPERTY**
Initial Equilibrium (with gravity)	**STEP** **SOLVE** **SET gravity**
Perform Alterations	**MODEL** **PROPERTY** **APPLY** **FIX** **FREE**
Save/Restore Problem State	**SAVE** **RESTORE**

2.7 Sign Conventions

The following sign conventions are used in *FLAC* and must be kept in mind when entering input or evaluating results.

DIRECT STRESS — Positive stresses indicate tension; negative stresses indicate compression.

SHEAR STRESS — With reference to Figure 2.34, a positive shear stress points in the positive direction of the coordinate axis of the second subscript if it acts on a surface with an outward normal in the positive direction. Conversely, if the outward normal of the surface is in the negative direction, then the positive shear stress points in the negative direction of the coordinate axis of the second subscript. The shear stresses shown in Figure 2.34 are all positive.

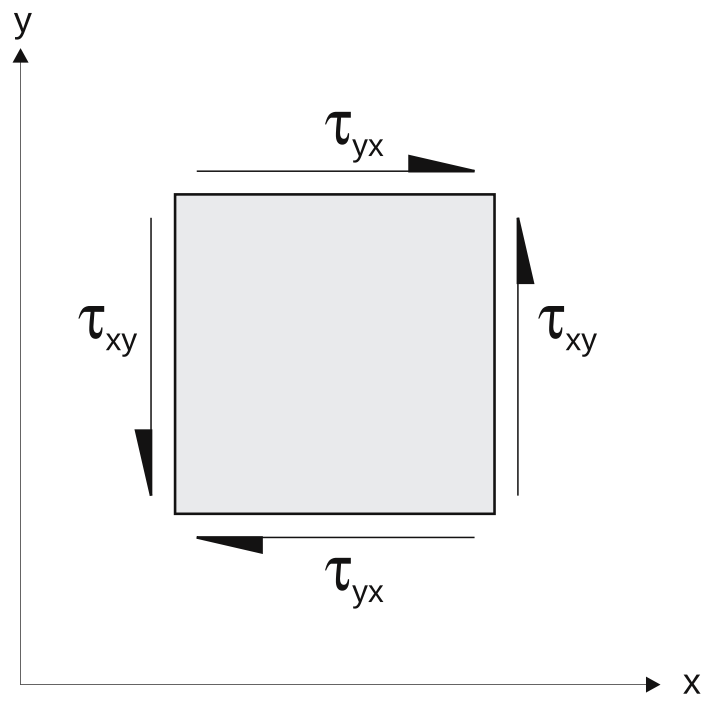


Figure 2.34 Sign convention for positive shear stress components

DIRECT STRAIN — Positive strain indicates extension; negative strain indicates compression.

SHEAR STRAIN — Shear strain follows the convention of shear stress (see above). The distortion associated with positive and negative shear strain is illustrated in Figure 2.35.

PRESSURE — A positive pressure will act normal to, and in a direction toward, the surface of a body (i.e., push). A negative pressure will act normal to, and in a direction away from, the surface of a body (i.e., pull). Figure 2.36 illustrates this convention.

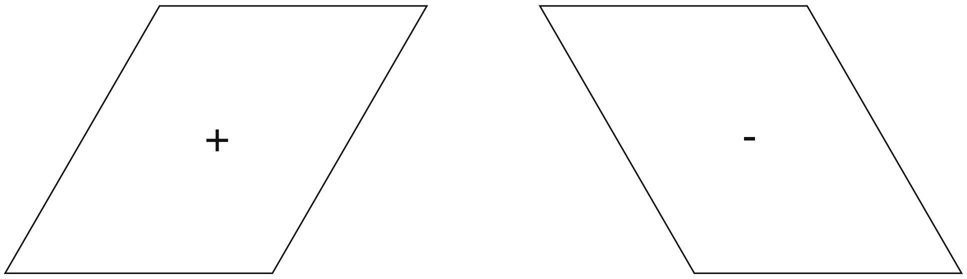


Figure 2.35 Distortion associated with positive and negative shear strain

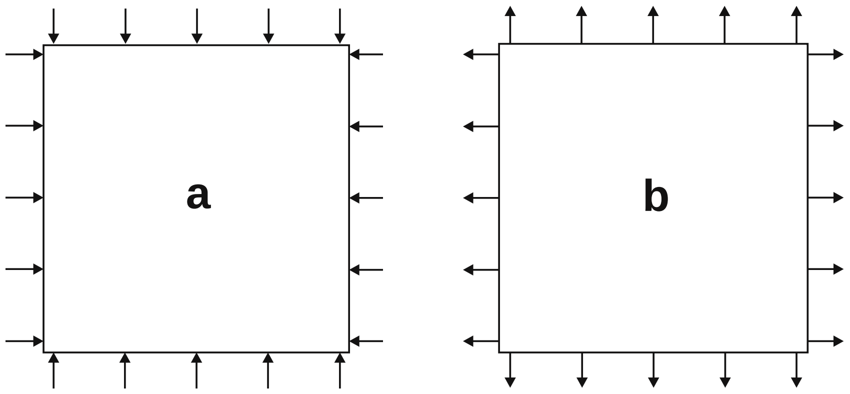


Figure 2.36 Mechanical pressure: (a) positive; (b) negative

PORE PRESSURE — Fluid pore pressure is positive in compression. Negative pore pressure indicates fluid tension.

GRAVITY — Positive gravity will pull the mass of a body downward (in the negative y-direction). Negative gravity will pull the mass of a body upward.

GFLOW — This is a *FISH* parameter (see Section 2 in the ***FISH* volume**) which denotes the net fluid flow associated with a gridpoint. A positive **`gflow`** corresponds to flow into a gridpoint. Conversely, a negative **`gflow`** corresponds to flow out of a gridpoint.

TFLOW — This is also a *FISH* parameter which denotes net heat flux associated with a gridpoint. The convention for heat flux at a gridpoint is the same as for fluid flow.

The x- and y-components of vector quantities such as forces, displacements, velocities, and flow vectors are positive when pointing in the directions of the positive x- and y-coordinate space.

INTERFACES — Positive shear stresses are induced at interface nodes for the following direction of relative movement:

$$\rightleftarrows$$

Shear displacements in the sense depicted above are plotted as filled areas or curves to the *right* of the interface, when looking along the Aside of the interface, in the direction in which it was specified.

Normal stress is negative if the interface node is in compression.

Compressional displacements are plotted as filled areas or curves to the *left* of the interface, when looking along the Aside of the interface, in the direction in which it was specified.

STRUCTURAL ELEMENTS — Axial forces in structural elements are positive in compression. Shear forces in structural elements follow the opposite sign convention as that given for zone shear stress, illustrated in Figure 2.34. Moments at the end of beam and pile elements are positive in the counterclockwise direction.

Translational displacements at nodes are positive in the direction of the positive coordinate axes, and angular displacements are positive in the counterclockwise direction.

The shear force and shear displacement at a cable/grout interface-spring node, or a pile shear coupling-spring node, are positive if the node displacement is in the direction of the specification of the cable or pile (i.e., **begin –> end**).

The normal force and normal displacement at a pile normal coupling-spring node are positive if the coupling spring is in compression.

2.8 Systems of Units

FLAC accepts any consistent set of engineering units. Examples of consistent sets of units for basic parameters are shown in Tables 2.6, 2.7 and 2.8. The user should apply great care when converting from one system of units to another. An excellent reference on the subject of units and conversion between the Imperial and SI systems can be found in the *Journal of Petroleum Technology* (December 1977). No conversions are performed in *FLAC* except for friction and dilation angles, which are entered in degrees.

Table 2.6 Systems of units — mechanical parameters

	SI				Imperial	
Length	m	m	m	cm	ft	in
Density	kg / m^3	10^3 kg / m^3	10^6 kg / m^3	10^6 g / cm^3	slugs / ft^3	snails / in^3
Force	N	kN	MN	Mdynes	lb$_f$	lb$_f$
Stress	Pa	kPa	MPa	bar	lb$_f$ / ft^2	psi
Gravity	m / sec^2	m / sec^2	m / sec^2	cm / s^2	ft / sec^2	in / sec^2
Stiffness*	Pa / m	kPa / m	MPa / m	bar / cm	lb$_f$ / ft^3	lb / in^3

* Stiffness refers to normal and shear stiffnesses at interfaces.

where	1 bar	=	10^6 dynes / cm^2 = 10^5 N / m^2 = 10^5 Pa;
	1 atm	=	1.013 bars = 14.7 psi = 2116 lb$_f$ / ft^2 = 1.01325×10^5 Pa;
	1 slug	=	1 lb$_f$ - s^2 / ft = 14.59 kg;
	1 snail	=	1 lb$_f$ -s^2 / in; and
	1 gravity	=	9.81 m / s^2 = 981 cm / s^2 = 32.17 ft / s^2.

Table 2.7 Systems of units — groundwater flow parameters

	SI		Imperial	
Water Bulk Modulus	Pa	bar	lbf/ft^2	psi
Water Density	kg / m^3	10^6 g / cm^3	slugs / ft^3	snails / in^3
Permeability	m^{3}sec / kg	10^{-6} cm sec / g	ft^3 sec / slug	in^3 sec / snail
Intrinsic Permeability	m^2	cm^2	ft^2	in^2
Hydraulic Conductivity	m / sec	cm / sec	ft / sec	in / sec

NOTE: *FLAC* permeability (in SI units) $\equiv$ intrinsic permeability (in cm^2) $\times 9.9 \times 10^{-2}$
$\equiv$ hydraulic conductivity (in cm / sec) $\times 1.02 \times 10^{-6}$

FLAC permeability is the *mobility coefficient* (coefficient of pore pressure term in Darcy’s law).

Systems of units for parameters associated with structural elements and heat transfer are given in Section 1 in **Structural Elements** and Section 1 in **Optional Features**, respectively.

Table 2.8 Systems of units — structural elements

Property	Unit	SI				Imperial	
area	length^2	m^2	m^2	m^2	cm^2	ft^2	in^2
axial or shear stiffness	force/disp	N/m	kN/m	MN/m	Mdynes/cm	lb$_f$/ft	lb$_f$/in
bond stiffness	force/length/disp	N/m/m	kN/m/m	MN/m/m	Mdynes/cm/cm	lb$_f$/ft/ft	lb$_f$/in/in
bond strength	force/length	N/m	kN/m	MN/m	Mdynes/cm	lb$_f$/ft	lb$_f$/in
exposed perimeter	length	m	m	m	cm	ft	in
moment of inertia	length^4	m^4	m^4	m^4	cm^4	ft^4	in^4
plastic moment	force-length	N-m	kN-m	MN-m	Mdynes-cm	ft-lb$_f$	in-lb$_f$
yield strength	force	N	kN	MN	Mdynes	lb$_f$	lb$_f$
Young's modulus	stress	Pa	kPa	MPa	bar	lb$_f$/ft^2	psi

where 1 bar = 10^6 dynes / cm^2 = 10^5 N / m^2 = 10^5 Pa,

Systems of units for parameters associated with heat transfer are given in Section 1 in **Optional Features**.

2.9 Precision Limits

When selecting a system of units, care should be taken to avoid calculations that approach the precision limits of the computer hardware. For 80386/387-based computers, the range is approximately 10^{-35} to 10^{35} in single-precision. If numbers exceed these limits, it is likely that the program will crash or, at least, produce artifacts in the model that may be difficult to identify or detect.

There are two versions of *FLAC*: a single-precision version and a double-precision version. In the single-precision version ("FLACW_SP.EXE"), calculations are primarily based upon single-precision variables. Errors may be introduced for some variables (e.g., pore pressure) in cases in which the accumulated value of the variable after many thousands of timesteps is much larger than the incremental change in the variable (i.e., an accumulated value that is roughly six orders of magnitude larger than the incremental value). In such a case, precision limitations will prevent further change to the value of the variable. It is recommended that the double-precision version ("FLACW_DP.EXE") be used for calculations involving variable changes of these magnitudes, or for grids containing many zones with coordinates that are large compared to typical zone dimensions.

2.10 Files

There are nine types of files that are either used or created by *FLAC*. The files are distinguished by their extensions and are described below.

INITIALIZATION FILE

"FLAC.INI" — This is a formatted ASCII file, created by the user, that *FLAC* will automatically access upon start-up or when a **NEW** command is issued. *FLAC* searches for the file "FLAC.INI" in the directory in which the code is executed and, if not found, in the directory pointed to by the ITASCA environment variable. The file may contain any valid *FLAC* command(s) (see Section 1 in the **Command Reference**). Although this file does not need to exist (i.e., no errors will result if it is absent), it is normally used to change default options in *FLAC* to those preferred by the individual user each time a new analysis is run (see Section 2.1.7).

DATA FILES

The user has a choice of running *FLAC* interactively (i.e., entering *FLAC* commands while in the *FLAC* environment) or via a data file (also called a "batch file"). The data file is a formatted ASCII file created by the user which contains the set of *FLAC* commands that represents the problem being analyzed. In general, creating data files is the most efficient way to use *FLAC*. To use data files with *FLAC*, see the **CALL** command in Section 1 in the **Command Reference**. Data files can have any filename and any extension. It is recommended that a common extension (e.g., ".DAT" for *FLAC* input commands, and ".FIS" for *FISH* function statements) be used to distinguish these files from other types of files.

SAVE FILES

"FLAC.SAV" — This file is created by *FLAC* at the user's request when issuing the command **SAVE**. The default file name is "FLAC.SAV," which will appear in the default directory when quitting *FLAC*. The user may specify a different filename by issuing the command **SAVE** filename, where filename is a user-specified filename. "FLAC.SAV" is a binary file containing the values of all state variables and user-defined conditions. The primary reason for creating save files is to allow one to investigate the effect of parameter variations without having to rerun a problem completely. A save file can be restored and the analysis continued at a subsequent time (see the **RESTORE** command in Section 1 in the **Command Reference**). If the save file is created in the *GIIC*, the file will also include information that describes the state of the *GIIC* at the stage the file is saved. Normally, it is good practice to create several save files during a *FLAC* run.

GIIC PROJECT SAVE FILES

This file is created at the user's request when the FILE / SAVE PROJECT menu item is pressed in the *GIIC*. The file name has the extension ".PRJ," which should not be changed. The file is an ASCII file containing variables that describe the state of the *GIIC* at the stage that the project is saved, and includes a link to the individual *FLAC* save files (".SAV") associated with the project.

GIIC MATERIALS FILES

This file is created by *FLAC* at the user's request as a library of commonly used material properties. The file is created from the *Materials List* dialog in the MATERIALS pane of the *GIIC*. This file is automatically given the extension ".GMT," and is an ASCII file containing the values of material properties that the user wishes to save for application in different projects. The file can be updated and modified from the *Materials List* dialog. A default materials file is provided that is automatically loaded in the MATERIALS pane.

LOG FILES

"FLAC.LOG" — This file is created by *FLAC* at the user's request when issuing the command **SET log on**. It is a formatted ASCII file. The default name of the file is "FLAC.LOG," which will appear in the default directory after quitting *FLAC*. The user may specify a different filename by issuing the command **SET log** filename, where filename is a user-supplied filename. The command may be issued interactively or be part of a data file. Subsequent to the **SET log on** command, all text appearing on the screen will be copied to the log file. The log file is useful in providing a record of the *FLAC* work session; it also provides a document for quality-assurance purposes.

HISTORY FILES

"FLAC.HIS" — This file is created by *FLAC* at the user's request when issuing the command **HISTORY write** ***n***, where ***n*** is a history number (see the **HISTORY** command, Section 1 in the **Command Reference**). It is a formatted ASCII file. The default name of the file is "FLAC.HIS," which will appear in the default directory after quitting *FLAC*. The user may specify a different filename by issuing the command **SET hisfile** filename. The user-supplied filename takes the place of "FLAC.HIS." The command may be issued interactively or be part of a data file. A record of the history values is written to the file, which can be examined using any text editor that can access formatted ASCII files. Alternatively, the file may be processed by a commercial graph-plotting or spreadsheet package.

PLOT FILES

Plot files are created at the user's request by issuing the command **COPY** filename in the command mode, after first creating the plot. By default, a Windows Enhanced Metafile will be created with the user-specified filename when **COPY** filename is issued. The plotter type can be changed with the **SET plot** command.

PCX output can also be created by either setting this output mode on with the **SET pcx on** command before creating the plot, or by pressing the <F2> key while in the graphics-screen mode. When PCX mode is turned on, or the <F2> key is pressed in the graphics-screen mode, a PCX screen dump will be written to a file named "FLAC.PCX." Only one screen image can be written to a file. The user may specify a different title name with the command **SET pcxfile** filename where the user-specified filename takes the place of "FLAC.PCX." PCX files consist of bitmaps of screen images; they are accepted by many image display and manipulation programs.

MOVIE FILES

“FLAC.DCX” — This file is created by *FLAC* at the user’s request when issuing the command **MOVIE on**. Its purpose is to capture graphics images for playback as a movie on the computer monitor at a later time. The default file name is “FLAC.DCX,” which will appear in the default directory when quitting *FLAC*. The user may specify a different filename by issuing the command **MOVIE file** filename, where filename takes the place of “FLAC.DCX.” A DCX file format is used for the movie file. DCX files are a collection of PCX files and include an index to the PCX files. A DCX file can contain up to 1024 PCX images. See the **MOVIE** command in Section 1 in the **Command Reference**.

2.11 References

Journal of Petroleum Technology. “The SI Metric System of Units and SPE’s Tentative Metric Standard,” 1575-1616 (December, 1977).