U.S. DEPARTMENT OF THE INTERIOR
U.S. GEOLOGICALSURVEY

SLOPE--A BASIC program to compute the gravitational stress within a finite slope using a DOS-based PC computer

by

Philip Powers[1] and William Savage[1]

Open-File Report 93-546

[1]Denver, Colorado

CONTENTS

ILLUSTRATIONS

ACKNOWLEDGMENTS

SLOPE was written using Microsoft BASIC Professional Development System version 7.1 (Microsoft Corp., 1991). The mouse, windowing, and menu control routines were developed by Microsoft. Permission to use and freely distribute programs incorporating these routines is included as comments within the subroutines. The program listing included in the appendix of this report shows only the subroutine that contains the analytical solution for gravity-induced stresses in finite slopes.

DISKETTE CONTENTS

The source code, executable code, and sample output are included on the program disk. The files on the disk are:

SLOPE.BAS (main code module),
SLOPE.EXE (executable code), and
README (Title page of this report)
TEST.RUN (sample data set)

INTRODUCTION

SLOPE computes values for gravity-induced stresses in finite elastic slopes from an exact elastic solution. Results compare favorably with published photoelastic and finite-element results and satisfy the conditions that shear normal stresses vanish on the ground surface except at corners where stress concentrations exist. The solution predicts that horizontal stresses are compressive along the tops of slopes (tensile in the case of a 90-degree slope) and tensile away from the bottom of the slope--effects caused by upward movement in front of the slope in response to gravity loading of the slope.

SLOPE is written in the BASIC programming language (Microsoft Professional Development System) incorporating data entry and information windows, mouse selection, and pull-down menus. SLOPE.EXE will run on most PC-DOS (I.B.M.) or MS-DOS (Microsoft Corp.) based systems running DOS 3.0 or higher and equipped with the following hardware: CGA graphics display or better, a mouse (optional), and 640 kilobytes of memory.

OVERVIEW OF THE THEORY BEHIND SLOPE

SLOPE is based on an exact solution for gravity-induced stresses beneath a finite elastic slope of the type shown in figure 1. This solution is based on conformal mapping and a single analytic stress function. The conformal transformation of a finite slope of height b and slope angle β in x,y - coordinates into a half-plane in rectangular u,v - coordinates is given by the Schwarz-Christoffel transformation (Churchill, 1948),

$$z=f(w)$$
$$=\frac{bp}{\pi}\frac{m_p}{m_p^p-1}+\frac{b}{\pi}\ln\left[\frac{m_p+1}{m_p-1}\right]-\frac{b}{\pi}\sum_{k=1}^{p/2-1}P_k\cos(2k\pi/p)-\frac{2b}{\pi}\sum_{k=1}^{p/2-1}Q_k\sin(2k\pi/p) \quad ,$$

where

$$m_p=\left[\frac{w+1}{w-1}\right]^{1/p} \quad ,$$

$z = x + iy,\ w = u + iv,\ p = \pi/\beta,\ p = 2,4,6,8,\ldots,$

$$P_k=\ln[m_p^2-2m_p\cos(2k\pi/p)+1] \quad ,$$

and

$$Q_k=\tan^{-1}\left[\frac{\sin(2k\pi/p)}{m_p-\cos(2k\pi/p)}\right] \quad .$$

Note that this conformal transformation applies for even values of p, for example, for slope angles of 30 or 15 degrees. For odd values of p, for a slope angle of, say, 60 degrees, a different conformal transformation must be obtained.

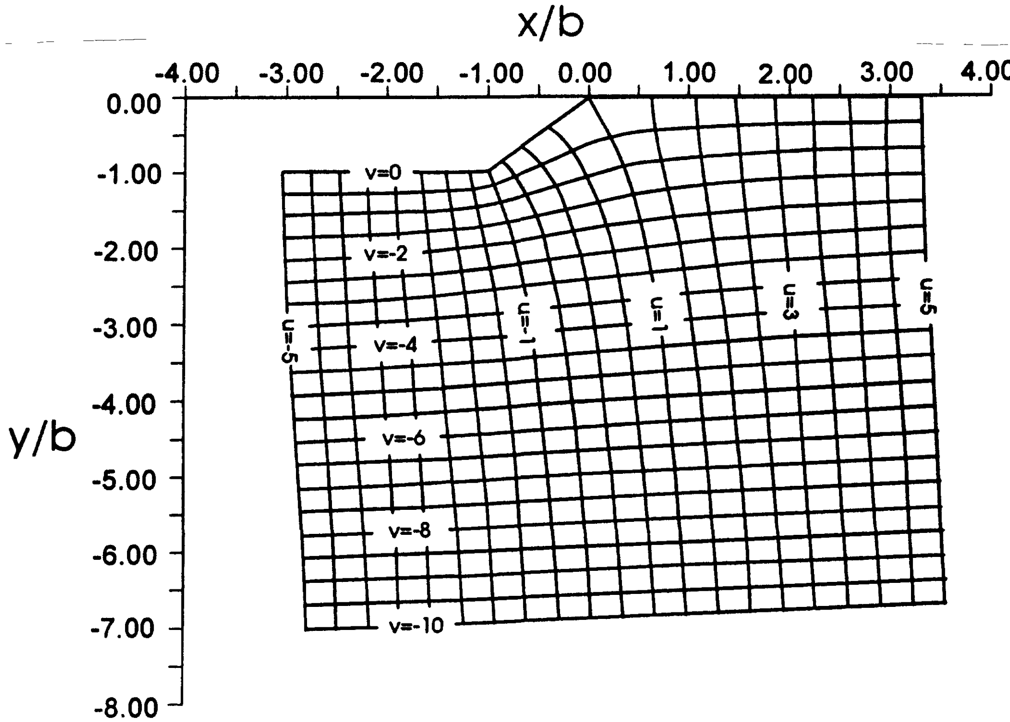

Figure 1--Finite 45° slope of height *b*. The dimensionless x,y-coordinate system and the conformally mapped u,v-coordinate system are shown.

The conformal transformation can be written in terms of real and imaginary parts (indicated by Re and Im) by using a polar coordinate system. The result is

$$x = Re\, f(w)$$

$$= \frac{bp}{2\pi} R_1 \left[\frac{R_2}{R_1}\right]^{1/p} \cos\left[\frac{(p-1)\theta_1+\theta_2}{p}\right] + \frac{b}{2\pi}\ln\left[\frac{(1+Re\, m_p)^2+(Im\, m_p)^2}{(1-Re\, m_p)^2+(Im\, m_p)^2}\right]$$

$$-\frac{b}{2\pi}\sum_{k=1}^{p/2-1} Re\, P_k \cos(2k\pi/p) + \frac{b}{\pi}\sum_{k=1}^{p/2-1} Re\, Q_k \sin(2k\pi/p)$$

$$y = Im\, f(w)$$

$$= \frac{bp}{2\pi} R_1 \left[\frac{R_2}{R_1}\right]^{1/p} \sin\left[\frac{(p-1)\theta_1+\theta_2}{p}\right] + \frac{b}{\pi}\left[\tan^{-1}\left[\frac{Im\, m_p}{1+Re\, m_p}\right] + \tan^{-1}\left[\frac{Im\, m_p}{1-Re\, m_p}\right]\right]$$

$$-\frac{b}{\pi}\sum_{k=1}^{p/2-1} Im\, P_k \cos(2k\pi/p) + \frac{b}{2\pi}\sum_{k=1}^{p/2-1} Im\, Q_k \sin(2k\pi/p)$$

where

$$R_1=\sqrt{(u-1)^2+v^2},\ R_2=\sqrt{(u+1)^2+v^2},$$

$$\theta_1=\tan^{-1}(v/(u-1)),\ \theta_2=\tan^{-1}(v/(u+1))$$

$$Re\ m_p=\left[\frac{R_2}{R_1}\right]^{1/p}\cos\left[\frac{\theta_2-\theta_1}{p}\right]$$

$$Im\ m_p=\left[\frac{R_2}{R_1}\right]^{1/p}\sin\left[\frac{\theta_2-\theta_1}{p}\right]$$

$$Re\ P_k=\ln[[1+(Re\ m_p)^2-(Im\ m_p)^2-2Re\ m_p\cos(2k\pi/p)]^2$$

$$+[2Im\ m_p\ Re\ m_p-2Im\ m_p\cos(2k\pi/p)]^2]$$

$$Re\ Q_k=\tan^{-1}\left[\frac{Im\ m_p-\sin(2k\pi/p)}{Re\ m_p-\cos(2k\pi/p)}\right]-\tan^{-1}\left[\frac{Im\ m_p+\sin(2k\pi/p)}{Re\ m_p-\cos(2k\pi/p)}\right]$$

$$Im\ P_k=\tan^{-1}\left[\frac{2Im\ m_p\ Re\ m_p-2Im\ m_p\cos(2k\pi/p)}{1+(Re\ m_p)^2-(Im\ m_p)^2-2Im\ m_p\cos(2k\pi/p)}\right]$$

and

$$Im\ Q_k=\ln\left[\frac{(Re\ m_p-\cos(2k\pi/p))^2+(Im\ m_p+\sin(2k\pi/p))^2}{(Re\ m_p-\cos(2k\pi/p))^2+(Im\ m_p-\sin(2k\pi/p))^2}\right].$$

This transformation is shown in figure 1 for a 45-degree slope.

The single stress function $\Phi(w)$ is given by

$$\Phi(w)=\frac{\rho gb}{4\pi}\left[\frac{1-2\mu}{1-\mu}\right][1-e^{-2i\pi/p}]i\ln[w+1]$$

$$-\frac{i\rho gbp}{8\pi}\left[\frac{1+(1-2\mu)e^{-2i\pi/p}}{1-\mu}\right]w+\frac{i\rho g}{4}\left[\frac{1+(1-2\mu)e^{-2i\pi/p}}{1-\mu}\right]f(w)$$

This stress function leads to gravity-induced stresses in finite slopes given by

$$\sigma_x = -Re\left[\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w)\right] + 2Re\,\Phi(w)$$
$$+Re\,[\bar{\Phi}(w)+\Phi(w)]\cos\left[\frac{2(\theta_2-\theta_1)}{p}\right] + Im\,[\bar{\Phi}(w)+\Phi(w)]\sin\left[\frac{2(\theta_2-\theta_1)}{p}\right] + \frac{\mu}{1-\mu}\rho g y$$

$$\sigma_y = Re\left[\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w)\right] + 2Re\,\Phi(w)$$
$$-Re\,[\bar{\Phi}(w)+\Phi(w)]\cos\left[\frac{2(\theta_2-\theta_1)}{p}\right] - Im\,[\bar{\Phi}(w)+\Phi(w)]\sin\left[\frac{2(\theta_2-\theta_1)}{p}\right] + \rho g y$$

and

$$\sigma_{xy} = Im\left[\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w)\right]$$
$$+Re\,[\bar{\Phi}(w)+\Phi(w)]\sin\left[\frac{2(\theta_2-\theta_1)}{p}\right] - Im\,[\bar{\Phi}(w)+\Phi(w)]\cos\left[\frac{2(\theta_2-\theta_1)}{p}\right]$$

Here

$$Re\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w) = 0$$

when $v = 0$ and

$$Re\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w)$$
$$= \frac{\rho g Im f(w)}{1-\mu}\left\{\left[\frac{1-2\mu}{pR_2}\right]\left[\frac{R_1}{R_2}\right]^{1/p}[\cos[\theta_2+(\theta_2-\theta_1)/p]-\cos[\theta_2+(\theta_2-\theta_1+2\pi)/p]]\right.$$
$$+\frac{1}{pR_2^2}\left[\frac{R_1}{R_2}\right]^{1/p}[\cos[2\theta_2+(\theta_2-\theta_1)/p]-[1-2\mu]\cos[2\theta_2+(\theta_2-\theta_1+\pi)/p]]$$
$$\left.+\left[\frac{R_1}{R_2}\right]^{1/p}[1-\cos[(\theta_2-\theta_1)/p]-[1-2\mu][\cos[(\theta_2-\theta_1+2\pi)/p]-\cos[2\pi/p]]]\right\}$$

when $v < 0$, and

$$Im\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w) = 0$$

when $v = 0$, and

$$Im\left[\frac{\overline{f(w)}-\bar{f}(w)}{f'(w)}\right]\Phi'(w)$$

$$=-\frac{\rho g Im f(w)}{1-\mu}\left\{\left[\frac{1-2\mu}{pR_2}\right]\left[\frac{R_1}{R_2}\right]^{1/p}[\sin[\theta_2+(\theta_2-\theta_1)/p]-\sin[\theta_2+(\theta_2-\theta_1+2\pi)/p]]\right.$$

$$+\frac{1}{pR_2^2}\left[\frac{R_1}{R_2}\right]^{1/p}[\sin[2\theta_2+(\theta_2-\theta_1)/p]-[1-2\mu]\sin[\theta_2+(\theta_2-\theta_1+\pi)/p]]$$

$$\left.-\left[\frac{R_1}{R_2}\right]^{1/p}[\sin[(\theta_2-\theta_1)/p]+[1-2\mu][\sin[(\theta_2-\theta_1+2\pi)/p]-\sin[2\pi/p]]]\right\}$$

when $v < 0$.
Also,

$$Re\ \Phi(w)=-\frac{\rho gb}{4\pi}\left[\frac{1-2\mu}{1-\mu}\right][[1-\cos(2\pi/p)]\theta_2+\ln R_2\sin(2\pi/p)]$$

$$+\frac{p\rho gb}{8\pi[1-\mu]}[v-[1-2\mu][u\sin(2\pi/p)-v\cos(2\pi/p)]]$$

$$+\frac{\rho g}{4}\left[\frac{1-2\mu}{1-\mu}\right][Re\ f(w)\sin(2\pi/p)-Im\ f(w)\cos(2\pi/p)]-\frac{\rho g\ Im\ f(w)}{4[1-\mu]}$$

$$Re\ [\bar{\Phi}(w)+\Phi(w)]$$

$$=-\frac{\rho gb}{2\pi}\left[\frac{1-2\mu}{1-\mu}\right]\ln R_2\sin(2\pi/p)$$

$$-\frac{P\rho gb}{4\pi}\left[\frac{1-2\mu}{1-\mu}\right]u\sin(2\pi/p)+\frac{\rho g}{2}\left[\frac{1-2\mu}{1-\mu}\right]Re\ f(w)\sin(2\pi/p)$$

and

$$Im\ [\bar{\Phi}(w)+\Phi(w)]$$

$$=-\frac{\rho gb}{2\pi}\left[\frac{1-2\mu}{1-\mu}\right]\theta_2\sin(2\pi/p)$$

$$-\frac{P\rho gb}{4\pi}\left[\frac{1-2\mu}{1-\mu}\right]v\sin(2\pi/p)+\frac{\rho g}{2}\left[\frac{1-2\mu}{1-\mu}\right]Im\ f(w)\sin(2\pi/p)$$

These equations for gravity induced stresses beneath finite slopes like that shown in figure 1 satisfy the conditions that shear and normal stresses parallel and perpendicular to the ground surface vanish. Note the sign convention adopted; compressive stresses are negative.

Figures 2a-f show, respectively, contours of gravity-induced dimensionless normal, $\sigma_x/\rho gb$ and $\sigma_y/\rho gb$, and shear stresses, $\sigma_{xy}/\rho gb$, greatest principal stresses $\sigma_1/\rho gb$, least principal stresses $\sigma_2/\rho gb$, and maximum shear stresses $\tau_{max}/\rho gb$ in the vicinity of a finite slope with a slope angle of 45 degrees when Poisson's ratio is 1/3. Note the concentrations of stress at the toe of the slope and the tensile region near the bottom and to the left of the bottom of the slope caused by upward movement and near-surface horizontal extension. Also note that the effects of the slope on gravity-induced normal and shear stresses dissolve with depth.

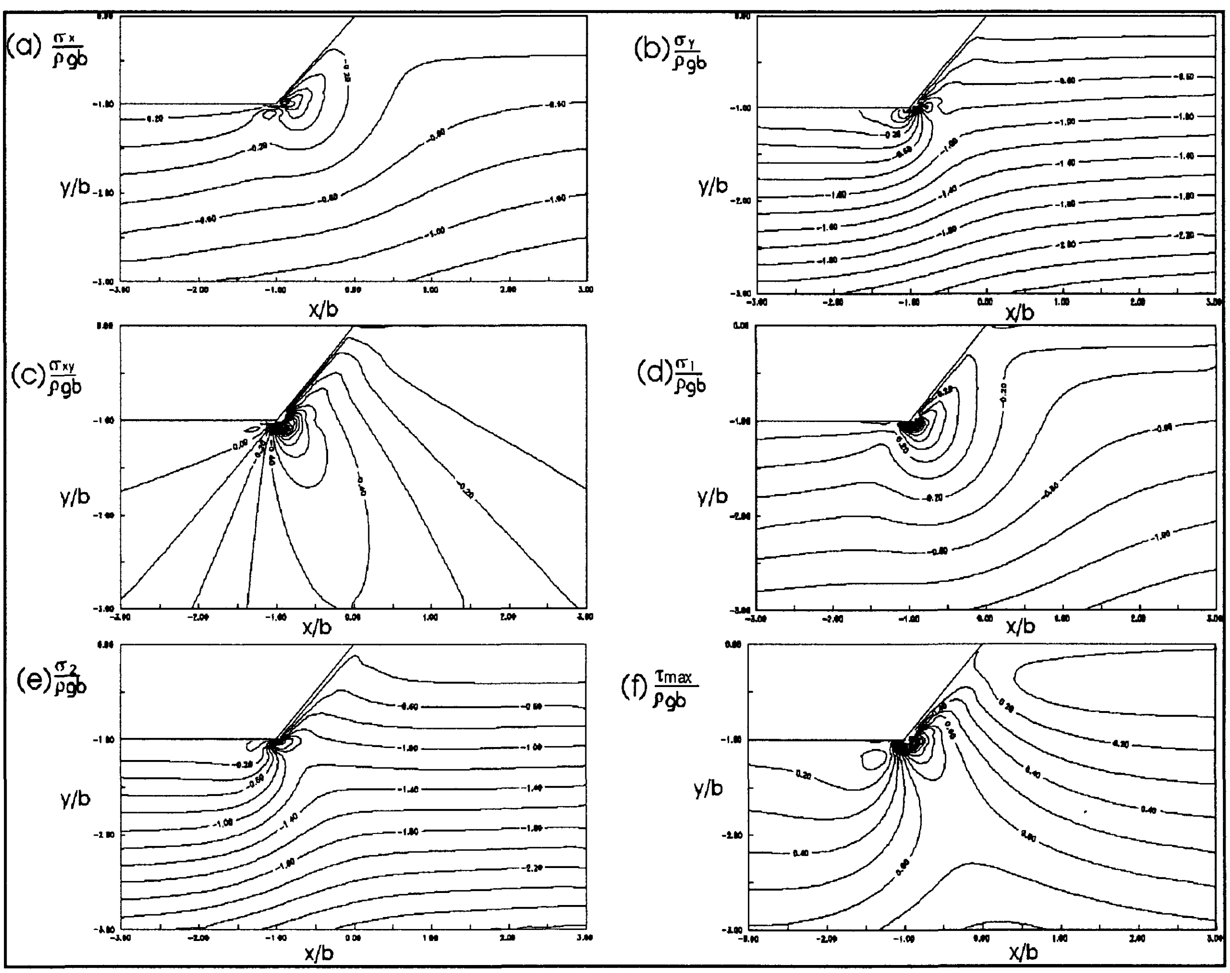

Figure 2--Stress contours for a 45° finite slope of height *b* when Poisson's ratio is 1/3.

USERS GUIDE TO RUNNING THE PROGRAM

Introductory Screen

To run the executable code SLOPE.EXE, copy the program to a subdirectory on the hard drive, change the working directory to that subdirectory, and simply type SLOPE. The following introductory screen appears on the monitor (figure 3):

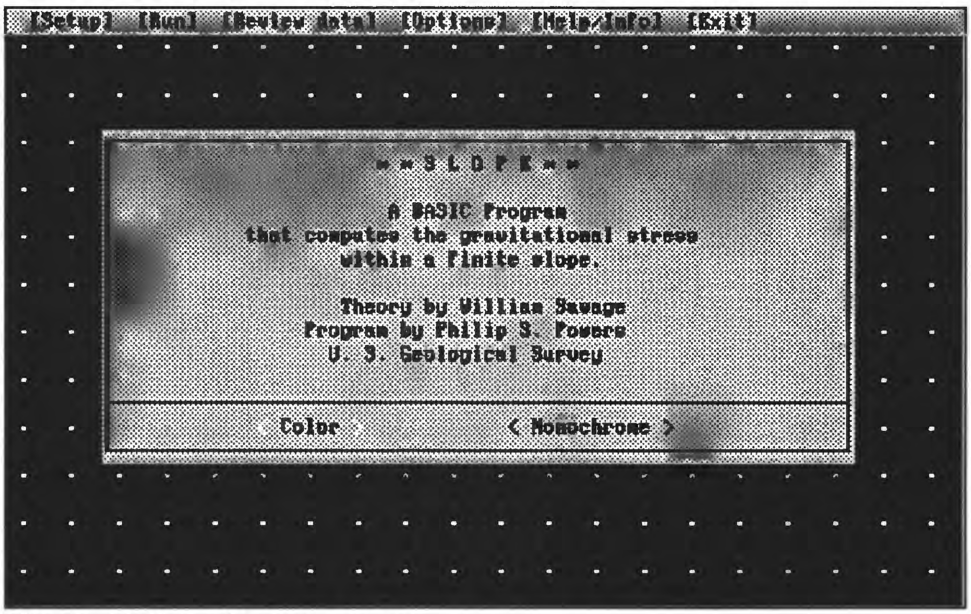

Figure 3--Introductory screen.

Using the left button of the mouse, click on <Color> or <Monochrome> to select the mode for the display. Alternatively, you can move between <Color> and <Monochrome> with the **[TAB]** key. When there are more than two options, **[Shift] [TAB]** moves backward through the options. The brackets for the active selection will be highlighted. To select the highlighted option, press the **[SPACEBAR]**.

After selecting color or monochrome, the introductory screen is cleared, leaving the menu bar at the top of the screen (figure 4). The main menu bar is activated by pressing the **[Alt]** key and using the arrow keys on the keyboard to move between choices or by clicking on the left button of the mouse at a particular choice. The **[Esc]** key cancels a selection and returns the control to the main menu bar.

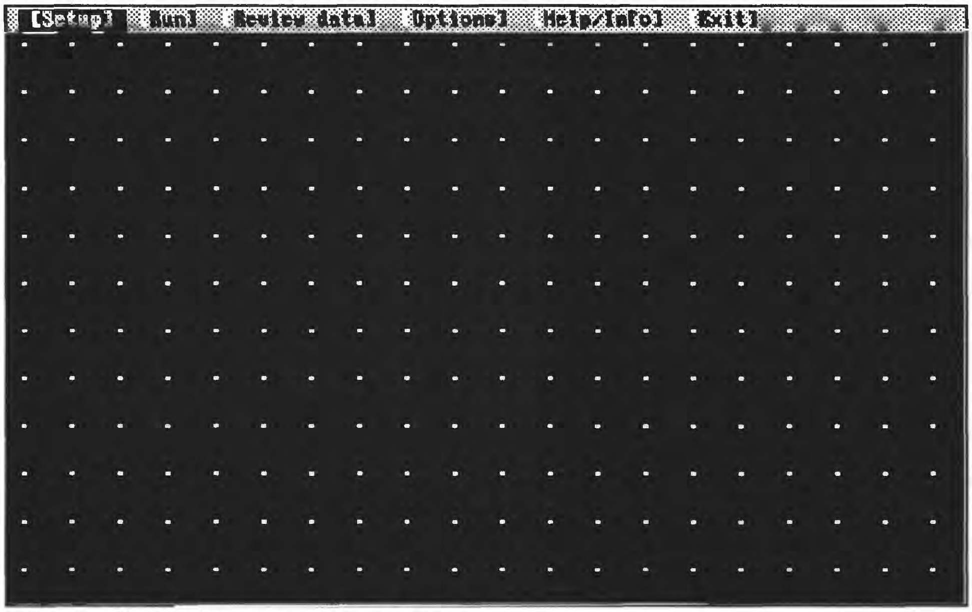

Figure 4--Main menu bar screen.

Setup

To start the program, click on **[Setup]** with the left mouse button, or use the [Alt] key and arrow keys to produce the pull down menu shown in figure 5.

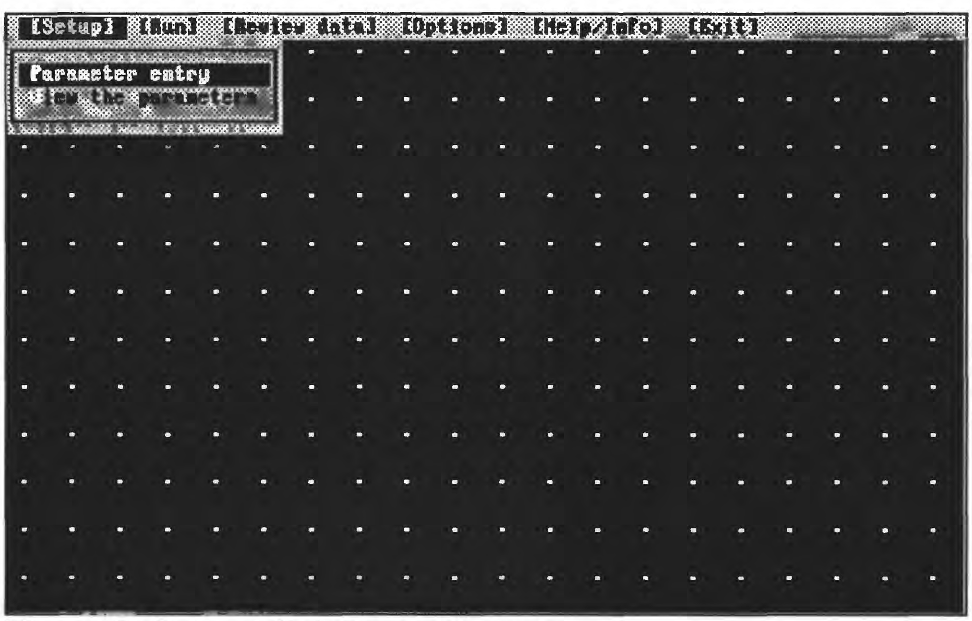

Figure 5--Setup pull down menu screen.

The two items in the pulldown menu are arranged in the order in which they should be selected. When the program is first started the only allowed selection is **Parameter entry**, the other choice is displayed in subdued characters and cannot be selected at this time. **Parameter entry**, can be accessed by pressing the highlighted letter 'P', clicked with the mouse, or moved to with the arrow keys, and selected with the **[RETURN]** key.

Selecting **Parameter entry** produces the screen shown in figure 6.

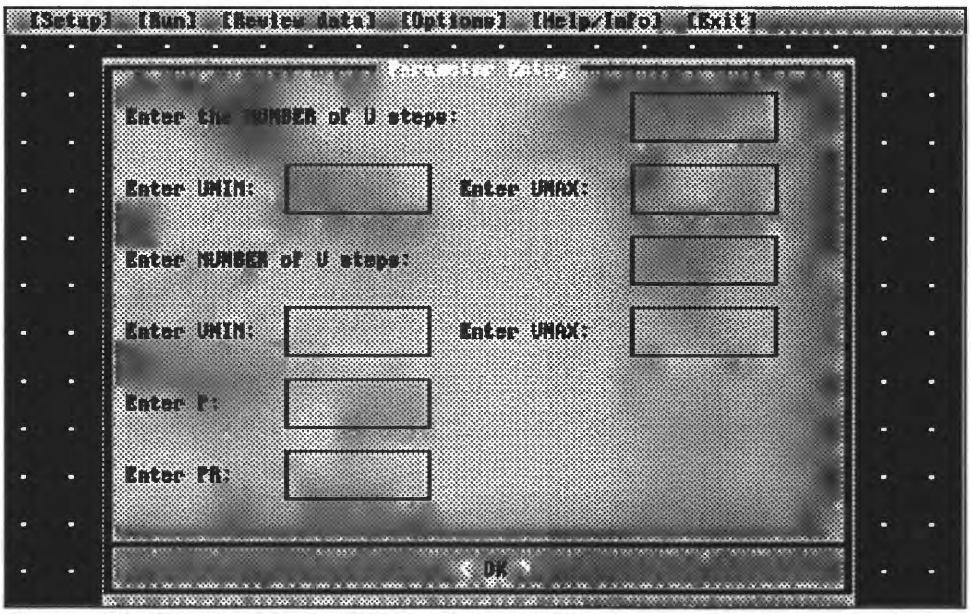

Figure 6--Parameter entry screen.

The first active edit field, indicated by the flashing cursor, is **Enter the NUMBER of U steps**. The cursor is at the first position in the edit field. Enter the desired number of u-steps. Recall that u is the horizontal rectangular coordinate that is mapped along with v, the vertical rectangular coordinate, into finite slopes of the type shown in figure 1. Press the **[RETURN]** or **[TAB]** to proceed to the next field, the maximum and minimum u-values, then proceed to the desired number of v-steps and the maximum and minimum v-values. Note that the minimum v-value will be negative and the maximum v-value will generally be zero, which represents the surface of the lower u,v half-plane. Next enter P, which is given by $P=\pi/b$, where b is the slope angle. Note that P must be even. Finally, enter the value for Poisson's ratio (PR). Continue making entries until all fields contain data. Press the **[TAB]** key until <OK>is highlighted, and press the **[SPACEBAR]** to close the parameter entry menu. The values entered during parameter entry will be checked for valid ranges, and if invalid, you will be returned to the parameter edit field to make corrections. To duplicate the output data set in figure 8, use the following for input:

U steps 3
Umin . . . . -4
Umax 4

V steps 3
Vmin . . . . -4
Vmax 0
P 4
PR 0.3333

Click on **[Setup]** and note that the selection **View the parameters** is no longer displayed in subdued print. If **View the parameters** is selected from the pull-down menu, the screen containing the input parameters given above is displayed (figure 7). After displaying the parameters, press the **[SPACEBAR]** or click on <OK> with the mouse to return to the menu bar.

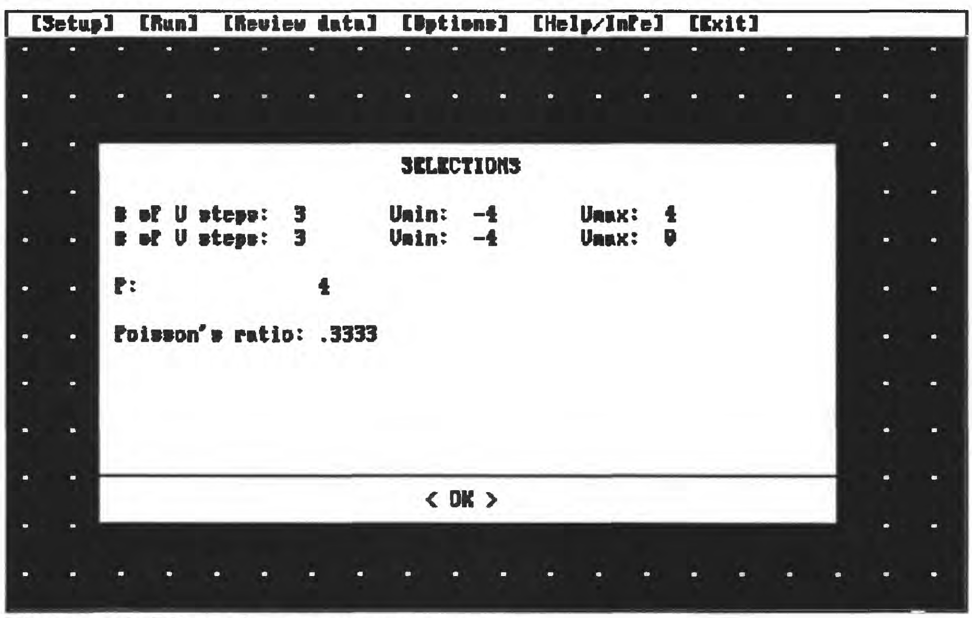

Figure 7--View the parameters screen.

Run

Next, click on **[Run]** to display the run pull-down menu, and again click on **Run** on the pull-down menu. A window appears that prompts you to enter the name of the data output file. Recall that DOS filenames can have no more than eight characters in the first name, followed by a period and up to a three character extension. After entering the output file name, press **[TAB]** to select <OK> and press the **[SPACEBAR]**. If the file name you have chosen already exists in the working directory, you will be asked to overwrite, append, or select a new name. Press **[TAB]** until the desired selection is highlighted, and then press the **[SPACEBAR]/[RETURN]** key. The calculations begin, and **WORKING** is displayed in the middle of the screen. When the calculations are finished, you will hear two beeps, and a window appears displaying **Finished with the calculations.** Press the **[SPACEBAR]** to return to the main menu bar.

Review Data

Click on **[Review data]** from the main menu bar. The screen will be filled with data and a prompt to see the next page of data will appear (figure 8). After all the data have been displayed, the screen will display the main menu bar again. Press **[Esc]** to abort the review data process.

x	y	sigx	sigy	sigxy	sig1	sig2	tmax	alfa
-2.3620	-3.3046	-1.1330	-2.4980	-0.1870	-1.1080	-2.5230	0.707	-7.664
-0.7594	-3.1647	-1.1590	-2.5740	-0.3700	-1.0680	-2.6640	0.798	-13.812
0.9267	-2.9638	-1.2160	-2.6420	-0.3430	-1.1380	-2.7210	0.791	-12.859
2.7112	-2.8084	-1.3100	-2.6160	-0.1970	-1.2800	-2.6450	0.682	-8.398
-2.4109	-2.5174	-0.6830	-1.6330	-0.1170	-0.6680	-1.6470	0.489	-6.914
-0.8628	-2.3638	-0.7170	-1.7230	-0.3890	-0.5850	-1.8560	0.636	-18.850
0.8123	-2.0789	-0.8130	-1.8180	-0.3120	-0.7230	-1.9070	0.592	-15.934
2.6545	-1.8966	-0.9360	-1.7550	-0.1300	-0.9160	-1.7750	0.430	-8.798
-2.4501	-1.7505	-0.2030	-0.7780	-0.0200	-0.2020	-0.7780	0.288	-2.036
-0.9948	-1.6137	-0.2820	-0.8260	-0.4240	-0.0500	-1.0580	0.504	-28.667
0.6474	-1.1469	-0.4770	-0.9710	-0.2180	-0.3950	-1.0540	0.329	-20.696
2.6067	-0.9595	-0.5830	-0.8800	-0.0520	-0.5740	-0.8890	0.157	-9.658
-2.4659	-1.0000	0.3660	0.0000	0.0000	0.3660	0.0000	0.183	0.000
-1.1061	-1.0000	0.5460	0.0000	0.0000	0.5460	0.0000	0.273	0.000
0.4504	0.0000	-0.3340	0.0000	0.0000	0.0000	-0.3340	0.167	90.000
2.5867	0.0000	-0.2360	0.0000	0.0000	0.0000	-0.2360	0.118	90.000

Press any key to proceed.

Figure 8--Sample output data based on test input data set.

Options

The **[Options]** selection on the main menu bar allows the user to change during the use of the program between a color and a monochrome display. The currently active selection is indicated by a small double arrow pointing to either color or monochrome.

Help/Info

The **[Help/Info]** displays a pull-down menu with three options. Each option displays information about the program. Selecting the **[Help/Info]** choice displays the pull-down menu shown in figure 9.

Figure 9--Help/Info pull-down menu screen.

Selecting **About This Program** from the **[Help/Info]** pull-down menu displays the window of information shown in figure 10.

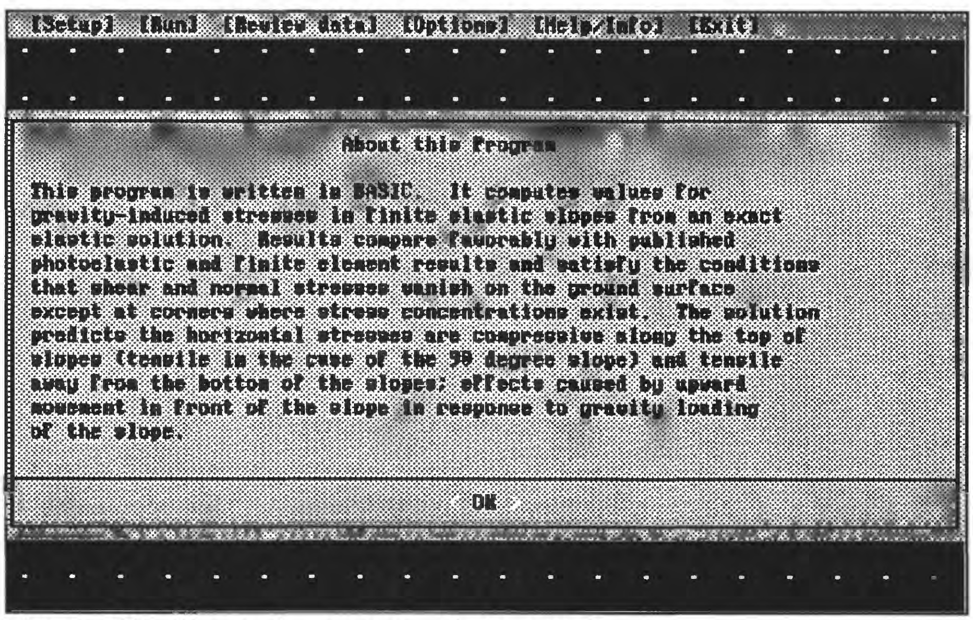

Figure 10--About the program help screen.

Selecting **Program usage** from the **[Help/Info]** pull-down menu displays the window of information shown in figure 11.

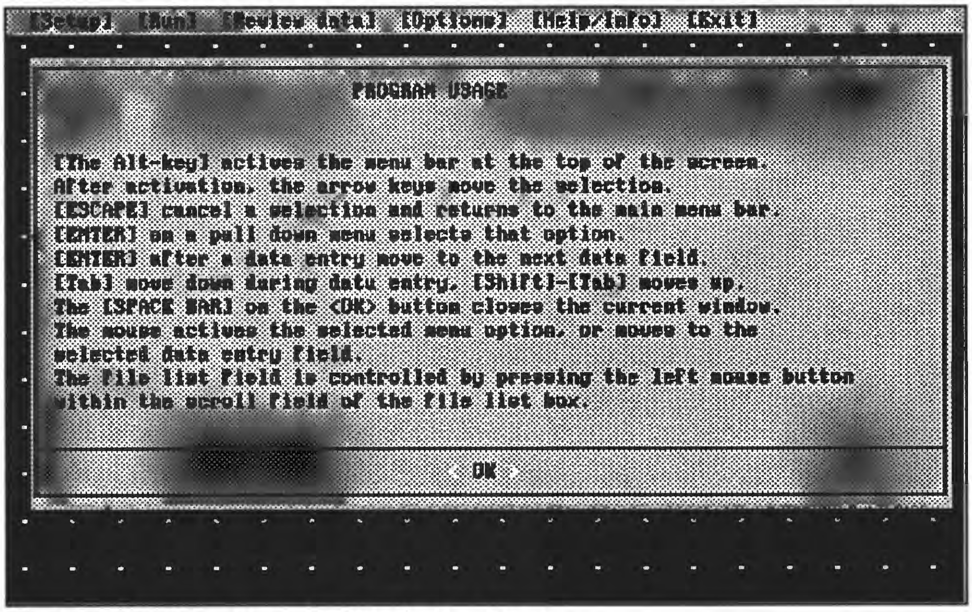

Figure 11--Program usage help screen.

Selecting **Variable definitions** from the **[Help/Info]** pull-down menu displays the window of information shown in figure 12.

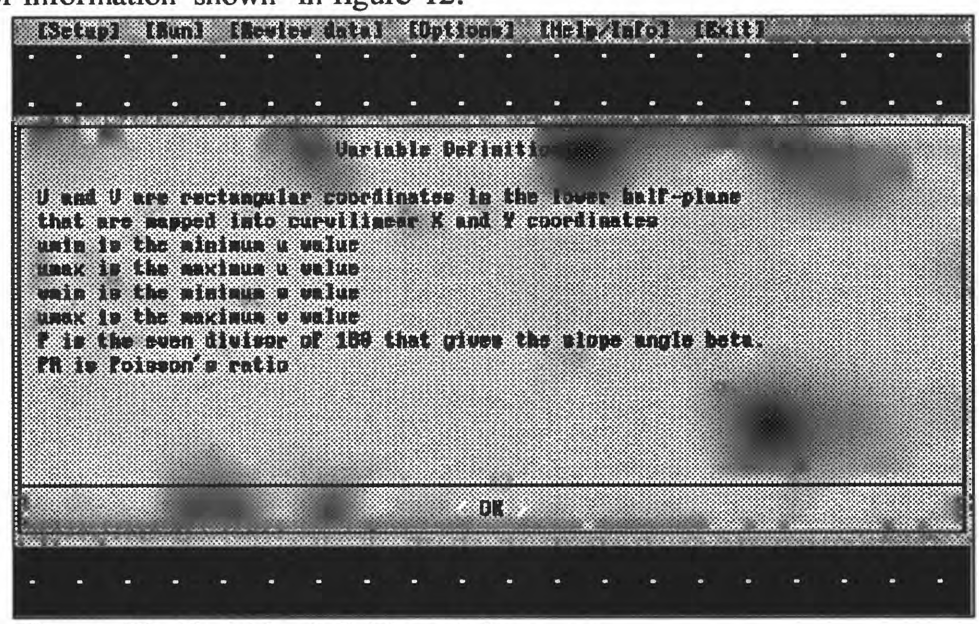

Figure 12--Variable definitions help screen.

Exit

Selecting **[Exit]** will produce a pull-down menu with two options, yes or no. Select Yes to end the program or No to return to the program.

REFERENCES

Churchill, R.V., 1948, Complex variables: NY, McGraw-Hill 216p

Microsoft Corporation, 1987, Microsoft BASIC Programmer's Guide Version 7.0 For IBM Personal Computers and Compatibles, United States and Canada.

Microsoft Corporation, 1991, Microsoft MS-DOS Users Guide and reference--for the MS-DOS Operating System version 5.0, United States.

APPENDIX--BASIC program listing

Subroutine Rnprog

```
SHARED nus, umin, umax, nvs, vmin, vmax, p, mu, c, phi1, filename1$

'ifno number of u steps entered, exit sub
IF nus = 0 THEN
      junk = Alert(4, " Run [Setup] first.",8, 5, 10, 55, "","","")
      EXIT DEF
END IF

CALL SaveToFile
IF filename1$ = ""THEN EXIT DEF

IF Header% = 1 THEN
      PRINT #2, "    x         y      sigx     sigy     sigxy    sig1     sig2     tmax     alfa"
END IF

LOCATE 10, 30
PRINT "WORKING"
mu1 = (1 - 2 * mu) / (1 - mu)
mu2 = 1 / (1 - mu)
mu3 = mu / (1 - mu)
kmax = p / 2 - 1
pi = 3.14159265358979#
uinc = (umax - umin) / nus
vinc = (vmax - vmin) / nvs
numax% = nus
nvmax% = nvs
nv = 0
v = vmin
cp1 = COS(2 * pi / p)
sp1 = SIN(2 * pi / p)

DO
      u = umin
      nu = 0

      DO
      r1 = SQR((u - 1) ^ 2 + v ^ 2)
      r2 = SQR((u + 1) ^ 2 + v ^ 2)
      IF r2 < 9.999999999999D-12 THEN r2 = 0
```

```
IF r1 < 9.999999999999D-12 THEN r1 = 0
theta1 = Atan2(v, u - 1)
theta2 = Atan2(v, u + 1)
beta = (theta2 - theta1) / p
IF v ^ 2 < 9.999999999999D-12 THEN v = 0

IF v = 0 AND u < -1 THEN
       theta1 = -pi
       theta2 = -pi
       beta = 0
END IF

IF v = 0 AND u ^ 2 < 1 THEN
       theta1 = -pi
       theta2 = 0
       beta = pi / p
END IF

IF v = 0 AND u > 1 THEN
       theta1 = 0
       theta2 = 0
       beta = 0
END IF

IF r1 = 0 OR r2 = 0 THEN
       IF r1 = 0 THEN GOSUB choice1:
       IF r2 = 0 THEN GOSUB choice3:
ELSE
       GOSUB choice2:
END IF

tmax = SQR(((sigx - sigy) / 2) ^ 2 + sigxy ^ 2)
sigsum = sigx + sigy
sigd = sigy - sigx
sigt = sigsum / 2 - (sigd / 2) * COS(2 * beta) + sigxy * SIN(2 * beta)
sign = sigsum / 2 - (sigd / 2) * COS(2 * beta) + sigxy * SIN(2 * beta)
signt = (sigd / 2) * SIN(2 * beta) + sigxy * COS(2 * beta)
sig1 = sigsum / 2 + tmax
sig2 = sigsum / 2 - tmax
alfa = (90! / pi) * Atan2(2 * sigxy, (sigx - sigy))
PRINT #2, USING "####.#####      ####.#####      ####.###     ####.###
####.###    ####.###    ####.###    ####.###    ####.###";    x; y; sigx; sigy; sigxy;
sig1; sig2; tmax; alfa
nu = nu + 1
```

```
        u = u + uinc
        LOOP WHILE nu <= numax%
        nv = nv + 1
        v = v + vinc
LOOP WHILE nv <= nvmax%
CLOSE #2
GOTO around:

choice1:
        x = 0
        y = 0
        sigx = -mu1 * SIN(2 * pi / p) * (p + 2 * LOG(2)) / (2 * pi)
        sigy = 0
        sigxy = 0
        beta = 0
RETURN

choice2:

r21 = (r2 / r1) ^ (1 / p)
theta21 = (theta2 - theta1) / p
theta123 = (theta2 + (p - 1) * theta1) / p
cs21 = COS(theta21)
sn21 = SIN(theta21)
mpr = r21 * cs21
mpi = r21 * sn21
rfd = (p * mpr) / (2 * pi)
ifd = (p * mpi) / (2 * pi)
rt1 = .5 * p * r1 * r21 * COS(theta123)
a1 = SQR(((1 + mpr) ^ 2 + mpi ^ 2) / ((1 - mpr) ^ 2 + mpi ^ 2))
racth = LOG(a1)
it1 = .5 * p * r1 * r21 * SIN(theta123)
iacth = Atan2((2 * mpi), (mpi ^ 2 + mpr ^ 2 - 1))
rs1 = 0
rs2 = 0
is1 = 0
is2 = 0
k = 1

DO
        IF kmax = 0 THEN
          EXIT DO
        END IF
```

```
        cp = COS(2 * k * pi / p)
        sp = SIN(2 * k * pi / p)
        b1 = (1 + mpr ^ 2 - mpi ^ 2 - 2 * mpr * cp)
        b2 = 2 * (mpi * mpr - mpi * cp)
        c1 = mpi - sp
        c2 = mpi + sp
        c3 = mpr - cp
        c4 = (c3 ^ 2 + c2 ^ 2) / (c3 ^ 2 + c1 ^ 2)
        rs1 = rs1 + .5 * LOG(b1 ^ 2 + b2 ^ 2) * cp
        rs21 = Atan2(c1, c3)
        rs22 = Atan2(c2, c3)
        rs2 = rs2 + (rs21 - rs22) * sp
        is1 = is1 + Atan2(b2, b1) * cp
        is2 = is2 + .5 * LOG(c4) * sp
        k = k + 1
LOOP WHILE k <= kmax

x = (rt1 + racth - rs1 + rs2) / pi
y = (it1 - iacth - is1 + is2) / pi
cost = COS(2 * (theta2 - theta1) / p)
sint = SIN(2 * (theta2 - theta1) / p)
lr2 = LOG(r2)
rph1 = -mu1 * ((1 - cp1) * theta2 + lr2 * sp1) / (4 * pi)
rph2 = (p * mu2 * v) / (8 * pi) - p * mu1 * (u * sp1 - v * cp1) / (8 * pi) + mu1 * (x *
sp1 - y * cp1) / 4 - (y * mu2) / 4
sigsum = 4 * (rph1 + rph2) + mu2 * y
sumrphi = -mu1 * lr2 * sp1 / (2 * pi) - p * mu1 * u * sp1 / (4 * pi) + mu1 * x * sp1 /
2
difiphi = -mu1 * theta2 * sp1 / (2 * pi) - p * mu1 * v * sp1 / (4 * pi) + mu1 * y * sp1
/ 2
t1 = theta2 + theta21
t2 = theta2 + theta21 + 2 * pi / p
t3 = 2 * theta2 + theta21
t4 = 2 * theta2 + theta21 + pi / p
rd1 = 2 * mu1 * (COS(t1) - COS(t2)) / (p * r21 * r2)
rd3 = mu2 * (1 - COS(theta21)) / r21
rd4 = mu1 * (COS(2 * pi / p) - COS(theta21 + 2 * pi / p)) / r21
rdw = rd1 + rd3 + rd4
id1 = 2 * mu1 * (SIN(t2) - SIN(t1)) / (p * r2 * r21)
id3 = (mu2 * SIN(theta21) + mu1 * (SIN(theta21 + 2 * pi / p) - SIN(2 * pi / p))) / r21
idw = id1 + id3

IF v = 0 THEN
        dw = 0
```

```
ELSE
	dw = y

END IF

sigd1  = dw * rdw
sigd2  = -2 * sumrphi  * cost
sigd3  = -2 * difiphi * sint
sigdif = sigd1  + sigd2  + sigd3  + mu1  * y
sigx  = .5 * (sigsum  - sigdif)
sigy  = .5 * (sigsum  + sigdif)
sigxy1  = -difiphi * cost
sigxy2  = sumrphi  * sint
sigxy3  = .5 * dw * idw
sigxy = sigxy1  + sigxy2  + sigxy3
RETURN

choice3:
k = 1
rs2  = 9.999999999999D-12
x = 0
DO
	IF kmax  = 0 THEN
		EXIT DO
	END IF

	xs  = -4 * k * SIN(2 * k * pi / p) / p
	x = x + xs
	k = k + 1
LOOP WHILE k <= kmax

	y = -1
	sigtau  = mu1  * ((pi * x + p / 2 - LOG(rs2)) * SIN(2 * pi / p) + pi * COS(2  * pi /
p)) / (2 * pi)
	sigx  = sigtau  * SIN(2 * pi / p)
	sigy  = sigtau  * (1 - COS(2  * pi / p))
	sigxy  = sigtau  * (1 + COS(2  * pi / p))
	beta  = 0
RETURN

around:
junk = Alert(1, "   Finished  with the calculations",  15, 20, 17, 55, "OK","","")

IF Displaytype  THEN
```

```
   ColorDisplay
ELSE
   MonoDisplay
END IF

MenuSetState  1, 1, 1
MenuSetState  1, 2, 1

END SUB

SUB RnProg  STATIC

SHARED nus, umin, umax, nvs, vmin, vmax, p, mu, c, phi1, filename1$

'ifno number  of u steps  entered,  exit sub
IF nus  = 0 THEN
        junk = Alert(4, " Run [Setup]  first.",8, 5, 10, 55, "","","")
        EXIT SUB
END IF

CALL SaveToFile
IF filename1$  = ""THEN EXIT SUB

IF Header%  = 1 THEN
        PRINT #2, "    x          y      sigx     sigy     sigxy    sig1     sig2     tmax     alfa"
END IF

LOCATE 10, 30
PRINT "WORKING"
mu1  = (1 - 2 * mu) / (1 - mu)
mu2  = 1 / (1 - mu)
mu3  = mu / (1 - mu)
kmax  = p / 2 - 1
pi = 3.14159265358979#
uinc  = (umax  - umin) / nus
vinc = (vmax  - vmin) / nvs
numax%  = nus
nvmax%  = nvs
nv  = 0
v = vmin
cp1  = COS(2  * pi /  p)
sp1  = SIN(2 * pi /  p)

DO
```

```
u = umin
nu = 0

DO
r1 = SQR((u - 1) ^ 2 + v ^ 2)
r2 = SQR((u + 1) ^ 2 + v ^ 2)
IF r2 < 9.999999999999D-12  THEN r2 = 0
IF r1 < 9.999999999999D-12  THEN r1 = 0
theta1 = Atan2(v, u - 1)
theta2 = Atan2(v, u + 1)
beta = (theta2 - theta1) / p
IF v ^ 2 < 9.999999999999D-12  THEN v = 0

IF v = 0 AND u < -1 THEN
      theta1 = -pi
      theta2 = -pi
      beta = 0
END IF

IF v = 0 AND u ^ 2 < 1 THEN
      theta1 = -pi
      theta2 = 0
      beta = pi / p
END IF

IF v = 0 AND u > 1 THEN
      theta1 = 0
      theta2 = 0
      beta = 0
END IF

IF r1 = 0 OR r2 = 0 THEN
      IF r1 = 0 THEN GOSUB choice1:
      IF r2 = 0 THEN GOSUB choice3:
ELSE
      GOSUB choice2:
END IF

tmax = SQR(((sigx - sigy) / 2) ^ 2 + sigxy ^ 2)
sigsum = sigx + sigy
sigd = sigy - sigx
sigt = sigsum / 2 - (sigd / 2) * COS(2 * beta) + sigxy * SIN(2 * beta)
sign = sigsum / 2 - (sigd / 2) * COS(2 * beta) + sigxy * SIN(2 * beta)
signt = (sigd / 2) * SIN(2 * beta) + sigxy * COS(2 * beta)
```

```
        sig1 = sigsum / 2 + tmax
        sig2 = sigsum / 2 - tmax
        alfa = (90! / pi) * Atan2(2 * sigxy, (sigx - sigy))
        PRINT #2, USING "####.#####      ####.#####      ####.###      ####.###
####.###      ####.###      ####.###      ####.###      ####.###";      x; y; sigx; sigy; sigxy;
sig1; sig2; tmax; alfa
        nu = nu + 1
        u = u + uinc
        LOOP WHILE nu <= numax%
        nv = nv + 1
        v = v + vinc
LOOP WHILE nv <= nvmax%
CLOSE #2
GOTO around:

choice1:
        x = 0
        y = 0
        sigx = -mu1 * SIN(2 * pi / p) * (p + 2 * LOG(2)) / (2 * pi)
        sigy = 0
        sigxy = 0
        beta = 0
RETURN

choice2:

r21 = (r2 / r1) ^ (1 / p)
theta21 = (theta2 - theta1) / p
theta123 = (theta2 + (p - 1) * theta1) / p
cs21 = COS(theta21)
sn21 = SIN(theta21)
mpr = r21 * cs21
mpi = r21 * sn21
rfd = (p * mpr) / (2 * pi)
ifd = (p * mpi) / (2 * pi)
rt1 = .5 * p * r1 * r21 * COS(theta123)
a1 = SQR(((1 + mpr) ^ 2 + mpi ^ 2) / ((1 - mpr) ^ 2 + mpi ^ 2))
racth = LOG(a1)
it1 = .5 * p * r1 * r21 * SIN(theta123)
iacth = Atan2((2 * mpi), (mpi ^ 2 + mpr ^ 2 - 1))
rs1 = 0
rs2 = 0
is1 = 0
is2 = 0
```

```
k = 1

DO
      IF kmax = 0 THEN
         EXIT DO
      END IF

      cp = COS(2 * k * pi / p)
      sp = SIN(2 * k * pi / p)
      b1 = (1 + mpr ^ 2 - mpi ^ 2 - 2 * mpr * cp)
      b2 = 2 * (mpi * mpr - mpi * cp)
      c1 = mpi - sp
      c2 = mpi + sp
      c3 = mpr - cp
      c4 = (c3 ^ 2 + c2 ^ 2) / (c3 ^ 2 + c1 ^ 2)
      rs1 = rs1 + .5 * LOG(b1 ^ 2 + b2 ^ 2) * cp
      rs21 = Atan2(c1, c3)
      rs22 = Atan2(c2, c3)
      rs2 = rs2 + (rs21 - rs22) * sp
      is1 = is1 + Atan2(b2, b1) * cp
      is2 = is2 + .5 * LOG(c4) * sp
      k = k + 1
LOOP WHILE k <= kmax

x = (rt1 + racth - rs1 + rs2) / pi
y = (it1 - iacth - is1 + is2) / pi
cost = COS(2 * (theta2 - theta1) / p)
sint = SIN(2 * (theta2 - theta1) / p)
lr2 = LOG(r2)
rph1 = -mu1 * ((1 - cp1) * theta2 + lr2 * sp1) / (4 * pi)
rph2 = (p * mu2 * v) / (8 * pi) - p * mu1 * (u * sp1 - v * cp1) / (8 * pi) + mu1 * (x *
sp1 - y * cp1) / 4 - (y * mu2) / 4
sigsum = 4 * (rph1 + rph2) + mu2 * y
sumrphi = -mu1 * lr2 * sp1 / (2 * pi) - p * mu1 * u * sp1 / (4 * pi) + mu1 * x * sp1 /
2
difiphi = -mu1 * theta2 * sp1 / (2 * pi) - p * mu1 * v * sp1 / (4 * pi) + mu1 * y * sp1
/ 2
t1 = theta2 + theta21
t2 = theta2 + theta21 + 2 * pi / p
t3 = 2 * theta2 + theta21
t4 = 2 * theta2 + theta21 + pi / p
rd1 = 2 * mu1 * (COS(t1) - COS(t2)) / (p * r21 * r2)
rd3 = mu2 * (1 - COS(theta21)) / r21
rd4 = mu1 * (COS(2 * pi / p) - COS(theta21 + 2 * pi / p)) / r21
```

```
rdw = rd1 + rd3 + rd4
id1 = 2 * mu1 * (SIN(t2) - SIN(t1)) / (p * r2 * r21)
id3 = (mu2 * SIN(theta21) + mu1 * (SIN(theta21 + 2 * pi / p) - SIN(2 * pi / p))) / r21
idw = id1 + id3

IF v = 0 THEN
      dw = 0
ELSE
      dw = y

END IF

sigd1 = dw * rdw
sigd2 = -2 * sumrphi * cost
sigd3 = -2 * difiphi * sint
sigdif = sigd1 + sigd2 + sigd3 + mu1 * y
sigx = .5 * (sigsum - sigdif)
sigy = .5 * (sigsum + sigdif)
sigxy1 = -difiphi * cost
sigxy2 = sumrphi * sint
sigxy3 = .5 * dw * idw
sigxy = sigxy1 + sigxy2 + sigxy3
RETURN

choice3:
k = 1
rs2 = 9.999999999999D-12
x = 0
DO
      IF kmax = 0 THEN
            EXIT DO
      END IF

      xs = -4 * k * SIN(2 * k * pi / p) / p
      x = x + xs
      k = k + 1
LOOP WHILE k <= kmax

      y = -1
      sigtau = mu1 * ((pi * x + p / 2 - LOG(rs2)) * SIN(2 * pi / p) + pi * COS(2 * pi /
p)) / (2 * pi)
      sigx = sigtau * SIN(2 * pi / p)
      sigy = sigtau * (1 - COS(2 * pi / p))
      sigxy = sigtau * (1 + COS(2 * pi / p))
```

```
        beta  = 0
RETURN

around:
junk = Alert(1, "   Finished  with the  calculations",  15, 20, 17, 55, "OK","","")

IF Displaytype  THEN
    ColorDisplay
ELSE
   MonoDisplay
END IF

MenuSetState   1, 1, 1
MenuSetState   1, 2, 1

END SUB
```