MAR 1 5 1983

(Return to WM, 623-SS

MEMORANDUM FOR: Malcolm R. Knapp
High-Level Waste Licensing
Management Branch
Division of Waste Management

FROM: Matthew J. Gordon
High-Level Waste Licensing
Management Branch
Division of Waste Management

SUBJECT: NON-GENERALIZED PRE-PROCESSING PROGRAM FOR SWIFT

The enclosed is intended to document a non-generalized program that I have written to facilitate SWIFT input. This program should prove useful to WMHL SWIFT users, and can be extended in the future to provide a more general pre-processor.

Original Signed By:

Matthew J. Gordon
High-Level Waste Licensing
Management Branch
Division of Waste Management

Enclosure:
As stated

Distribution:
WM file: ~~3108.5~~ 109.9
WMHL r/f
NMSS r/f
REBrowning
MJBell
MJGordon & r/f
HJMiller
JOBunting
PSJustus
JTGreeves
PDR

8304140259 830315
PDR WASTE
WM-1 PDR

OFFICE	WMHL MJG						
SURNAME	MJGordon:lmc						
DATE	3/15/83						

NRC FORM 318 (10-80) NRCM 0240 OFFICIAL RECORD COPY USGPO: 1981—335-960

Introduction

One of the major criticisms of SWIFT is that it is not a user-friendly code. This criticism stems partly from the need to specify boundary conditions in terms of pressures, which are not easily calculated from known heads, particularly when density variations with depth are being considered. Also, following a specified format in boundary condition input, as required by SWIFT, often requires tedious and repetitive typing. The purpose of this memo is to document a pre-processing program I have written to facilitate two-dimensional modeling of BWIP. The program has been generalized only so far as to fit the needs of our present deterministic sensitivity analysis modeling efforts. The program is, however, general enough to be used for many SWIFT applications. The advantages of the pre-processor include:

1. Input data may be in terms of heads rather than pressures.

2. Input data may be calculated, if desired, from known gradients rather than from known pressures or heads.

3. Pressures are calculated considering fluid density variations arising from compressibility due to fluid pressure and thermal expansion due to a thermal gradient.

4. All input data is in free format.

5. The output can be easily converted to a form which can be merged directly into the SWIFT input deck.

Limitations of the pre-processor include:

1. It has not been generalized to consider density changes due to dissolved salt.

2. It has not been generalized to adjust for elevated or sloping grid blocks.

3. It is set up only for x-z 2-dimensional grids (vertical cross-sections).

4. It is set up only for SI units (meters, seconds, kilograms, pascals, etc.).

The expansion of this pre-processor (PRESS) to eliminate the above limitations will be a worthwhile project in the future. Also, it would be useful to alter the pre-processor to allow much of the PRESS input deck to be taken directly from the SWIFT input deck, rather than from a separate PRESS deck. However, the current version of PRESS serves the Performance Assessment Section's needs for the foreseeable (2-month) future.

The remainder of this memo describes the pre-processor and how it is used.

<u>Theory</u>

Density in SWIFT is considered as a function of temperature, salinity and pressure:

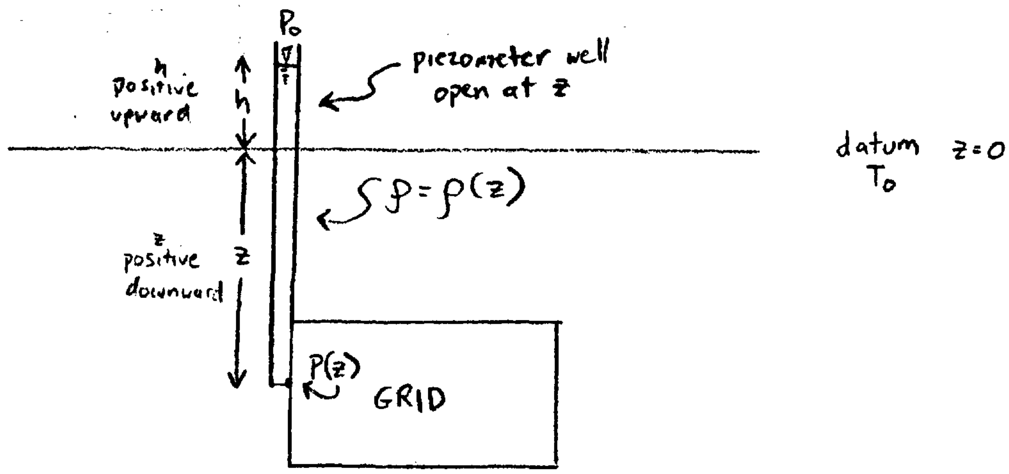
$$\rho(T,P,C) = \rho_o \left(1 + Cw\,(P-P_o) - C_t\,(T-T_o) + C_s\,(C - C_o)\right) \qquad (1)$$

where ρ_o = reference density at To, Co and Po (kg/m^3)
P_o = reference pressure (Pa)
T_o = reference temperature (C)
C_w = compressibility coefficient of water (Pa^{-1})
C_t = thermal expansion coefficient of water ($°C^{-1}$)
C_o = reference salinity (dimensionless concentration)
P = pressure (Pascals)
T = temperature (°C)
C = salinity (dimensionless concentration)
C_s = density correction coefficient for dissolved salt (dimensionless)

$\rho(T, P, C)$ = density at T, P, and C (Kg/m^3)

We will consider C_s to be zero, since the extent of salt dissolution is not well known for BWIP, and has not been considered in modeling efforts by DOE and NRC up to this date.

Consider the situation shown in Figure 1, where a grid is placed below a datum, and the temperature of water at the datum is known. This datum temperature will be considered to be the reference temperature, T_o.



The reference pressure, P_o, will be considered as atmospheric. At any point z, a measurement of head is made by measuring the height of a water column in a well open at that point. The water column is assumed to have reached thermal equilibrium with its surroundings. The temperature at z due to a geothermal gradient may be calculated with the equation:

$$T(z) = T_o + d(z)K_T \qquad (2)$$

where T_o = reference temperature (at datum) (°C)
$d(z)$ = depth below datum (m)
K_T = geothermal gradient coefficient (°C/m)

The pressure at that point may be calculated:

$$P(z) = P_o + (h(z) + d(z))\ \rho\ (z)g \qquad (3)$$

where h(z) = head at z (meters above datum)
d(z) = depth at z (meters below datum)
ρ(z) = average density of water column

In order to obtain ρ(z) we must solve equation (1). We can approximate the average density of the water column by the density at the midpoint of the column. This approximation is based on our assumption of the linear compressibility and thermal expansion functions in equation (1).

The temperature at the midpoint of the water column is given by:

$$T = T_o + \left[K_t \left\{d(z) - \frac{(h(z) + d(z))}{2}\right\} - T_o\right] \qquad (4)$$

The pressure at the midpoint of the water column can be approximated by:

$$P = P_o + \left[\rho_o g\ \frac{(h(z) + d(z))}{2}\right] \qquad (5)$$

Combination of equations (4) and (5) with (1), with $C_s = 0$, yields ρ(z). Insertion of ρ(z) into equation (3) yields P(z). Th error introduced by using the reference density ρ_o in equation (5) rather than the average density of the half-water column above the midpoint will be negligible for realistic density variations in the field.

Input/Output for PRESS

PRESS requires the following input deck:

FORMAT	CARD(S) NO.	VARIABLE(S)	DESCRIPTION
Integer Free	1	N	Number of layers in grid.
Real Free	2	RHOϕ, Pϕ, Tϕ, C_w, C_T	Fluid density, pressure, and temperature of datum, coefficient of compressibility, coefficient of thermal expansion. (All units in meters, seconds, kilograms, pascals, and/or °C).
Real Free	3	TGRADC	Thermal gradient coefficient (°C/M).
Real Free	4	Dϕ	Depth of below datum to top left corner of grid (M).
Real Free	5	DELZ(I) (I=1,N)	Width of each layer in z-direction (M).
Real Free	6	QUERY	Are heads on left vertical boundary to be 1) input by user, or 2) calculated from given top left head and given vertical gradient? = 1.0 if input by user = 0.0 if calculated
Real Free	7a	HEAD(1), UGRAD	Use 7a only if QUERY = 0.0. Input head on top left vertical boundary and vertical gradient on left and right boundaries (meters).

Real Free	7b	HEAD(I), I=1,N	Use 7b only if QUERY = 1.0. Input heads for each layer on left boundary (M).
Real Free	8	HGRAD	Horizontal gradient to be applied across grid boundaries.
Real Free	9	NX	Number of grid blocks in horizontal direction.
Real Free	10	DELX(I), (I=1, NX)	Width of each block in X direction (M).

In order to run PRESS on MFZ, the following Job stream is necessary:

```
USERNAME,STMFZ,TPϕ,T37.
ACCOUNT,USERNAME,PROBNO,PROBNO.
ATTACH,FILELIB,MR=1.
LIBRARY,FILELIB.
FILE,PRESS,RT=Z,BT=C,FL=90.
ATTACH,PRESS,PRESS,ID=GORDON,SF,MFA,MR=1.
FIN,I=PRESS,B,REW,L=0.
*EOR
(Input Deck)
```

Once the program has executed, the output file should be put in the user's editor. The output includes the following information:

For the left hand boundary only, a table is produced with the following information:

Layer, Pressure, Head, Layer Thickness, Depth Below Datum, and Density.

After this table, the right hand side pressures are printed out. This information should be enough to check the performance of the program. After these two tables, the remainder of the output is formatted to provide all of the R1-28 cards (AIF's) necessary for SWIFT. The cards in this section of the output should be saved as a separate file by finding the starting and ending line numbers of the desired cards:

S, PRFILE, N, 0, (starting line no.), (ending line no.).

The PRFILE must be edited in the following manner:

Type in	Computer will respond
E, PRFILE,S.	..
/ /=//, (1), A.	(#) CHANGE(S).
/+ϕ/=//, A.	(#) CHANGE(S).

PRESS creates an AIF for all blocks in both horizontal and vertical directions. Therefore two AIF's have been calculated for each of the four corner blocks. SWIFT permits only one AIF per grid block. Therefore, the user must delete one of the AIF sets for each of the four corner grid blocks.

Once these setps have been taken, the file should be saved, and the user is ready to merge this file of AIF's into his SWIFT input deck. This can be easily done using MERGE.

To merge 2 files X and Y, X is edited in the following manner:

An ampersand (&) is placed at the point where Y is to appear.

*EOR's and *EOF's must be changed, e.g., to .*EOR and .*EOF, or they will be lost during the merge.

Once these changes have been made, the following commands may be executed interactively:

```
FETCH,MERGE,SAS.
MERGE,X,Y,Z.
```

A new merged file, Z, will have been created. Before using Z, all of the changed *EOR's and *EOF's must be changed back, i.e.,

```
/.*EO/ = /*EO/, A.
```

Then, all that remains is to save and catalog the new file.

```
      PROGRAM PRESS(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      DIMENSION DEPTH(100),DELZ(100),T(100),HEAD(100),
     *RHOAV(100),P(100),PR(100),PB(100),PT(100),
     *DELX(100),DELCTR(100),DELH(100)
C*******SET GRAVITATIONAL CONSTANT M/SEC2
      PAV=0.0
      G=9.80655
C********READ NUMBER OF LAYERS AND REFERENCE VALUES
      READ(5,*)N
      DO 500 I=1,N
      P(I)=0.0
      DEPTH(I)=0.0
      RHOAV(I)=0.0
      PB(I)=0.0
      PT(I)=0.0
      DELCTR(I)=0.0
      T(I)=0.0
  500 CONTINUE
      SUMX=0.0
      READ(5,*)RHO0,P0,T0,CW,CT
      READ(5,*)TGRADC
      READ(5,*)D0
      READ(5,*)(DELZ(I),I=1,N)
C*********SET TEMPERATURES FOR EACH LAYER
      T(1)=T0+(D0*TGRADC)
      DEPTH(1)=D0
      DO 10 I=2,N
      J=I-1
      DEPTH(I)=DEPTH(J)+DELZ(J)
      T(I)=T0+(DEPTH(I)*TGRADC)
   10 CONTINUE
      READ(5,*)QUERY
      IF(QUERY.EQ.1.0)GO TO 650
      READ(5,*)HEAD(1),VGRAD
      DO 201 I=2,N
      J=I-1
      HEAD(I)=HEAD(J)+(VGRAD*DELZ(J))
  201 CONTINUE
      GO TO 651
  650 READ(5,*)(HEAD(I),I=1,N)
  651 CONTINUE
      DO 20 I=1,N
      PAV=P0+(RHO0*G*(HEAD(I)+DEPTH(I))/2.)
      TAV=T0+(TGRADC*(DEPTH(I)-((HEAD(I)+DEPTH(I))/2.)))
      RHOAV(I)=RHO0*(1.+(CW*(PAV-P0))-(CT*(TAV-T0)))
      P(I)=P0+(RHOAV(I)*G*(HEAD(I)+DEPTH(I)))
   20 CONTINUE
      WRITE(6,30)
   30 FORMAT(1X,'INPUT PRESSURES WITH DENSITY CORRECTIONS')
      WRITE(6,40)
   40 FORMAT(1X,'LAYER NO.     PRESSURE(PA)     HEAD(M)     LAYER THICKNE
     *SS   TEMP      DEPTH')
      DO 50 I=1,N
      WRITE(6,60)I,P(I),HEAD(I),DELZ(I),T(I),DEPTH(I),RHOAV(I)
   60 FORMAT(1X,I7,8X,1PE12.6,6X,0PF6.1,8X,0PF6.1,8X,F5.2,34X,F7.1,
     *5X,F7.1)
   50 CONTINUE
      READ(5,*)HGRAD
      READ(5,*)NX
      READ(5,*)(DELX(I),I=1,NX)
      DO 990 I=1,NX
      SUMX=SUMX+DELX(I)
  990 CONTINUE
      DO 677 I=1,N
      PR(I)=P(I)-(9.80655*RHOAV(I)*HGRAD*SUMX)
  677 CONTINUE
      WRITE(6,80)
   80 FORMAT(1X,'RIGHT SIDE BDRY PRESSURES')
      DO 100 I=1,N
      WRITE(6,70)I,PR(I)
   70 FORMAT(1X,I7,8X,1PE12.6)
  100 CONTINUE
      DO 702 I=1,N
      WRITE(6,62)I,I
   62 FORMAT(1X,'    1    1    1    1',I5,I5,'     0')
      WRITE(6,61)P(I),T(I)
   61 FORMAT(1X,'       1.0',1PE12.6,0PF10.2,'          0.0')
  702 CONTINUE
      DO 703 I=1,N
      WRITE(6,63)NX,NX,I,I
   63 FORMAT(1X,I5,I5,'    1    1',I5,I5,'    0')
      WRITE(6,64)PR(I),T(I)
   64 FORMAT(1X,'       2.0',1PE12.6,0PF10.2,'          0.0')
  703 CONTINUE
      DELCTR(1)=DELX(1)/2.
      DO 443 I=2,NX
      J=I-1
      DELCTR(I)=(DELX(J)+DELX(I))/2.
  443 CONTINUE
      PT(1)=P(1)-(DELCTR(1)*HGRAD*RHOAV(1)*G)
      DO 450 I=2,NX
      J=I-1
      PT(I)=PT(J)-(DELCTR(I)*HGRAD*RHOAV(1)*G)
  450 CONTINUE
      PB(1)=P(N)-(DELCTR(1)*HGRAD*RHOAV(N)*G)
      DO 460 I=2,NX
      J=I-1
      PB(I)=PB(J)-(DELCTR(I)*HGRAD*RHOAV(N)*G)
  460 CONTINUE
      DO 499 I=1,NX
      WRITE(6,470)I,I
  470 FORMAT(1X,I5,I5,'    1    1    1    1    0')
      WRITE(6,481)PT(I),T(1)
  481 FORMAT(1X,'       5.0',1PE12.6,0PF10.2,'          0.0')
      WRITE(6,482)I,I,N,N
  482 FORMAT(1X,I5,I5,'    1    1',I5,I5,'    0')
      WRITE(6,483)PB(I),T(N)
  483 FORMAT(1X,'       6.0',1PE12.6,0PF10.2,'          0.0')
  499 CONTINUE
      STOP
      END
```