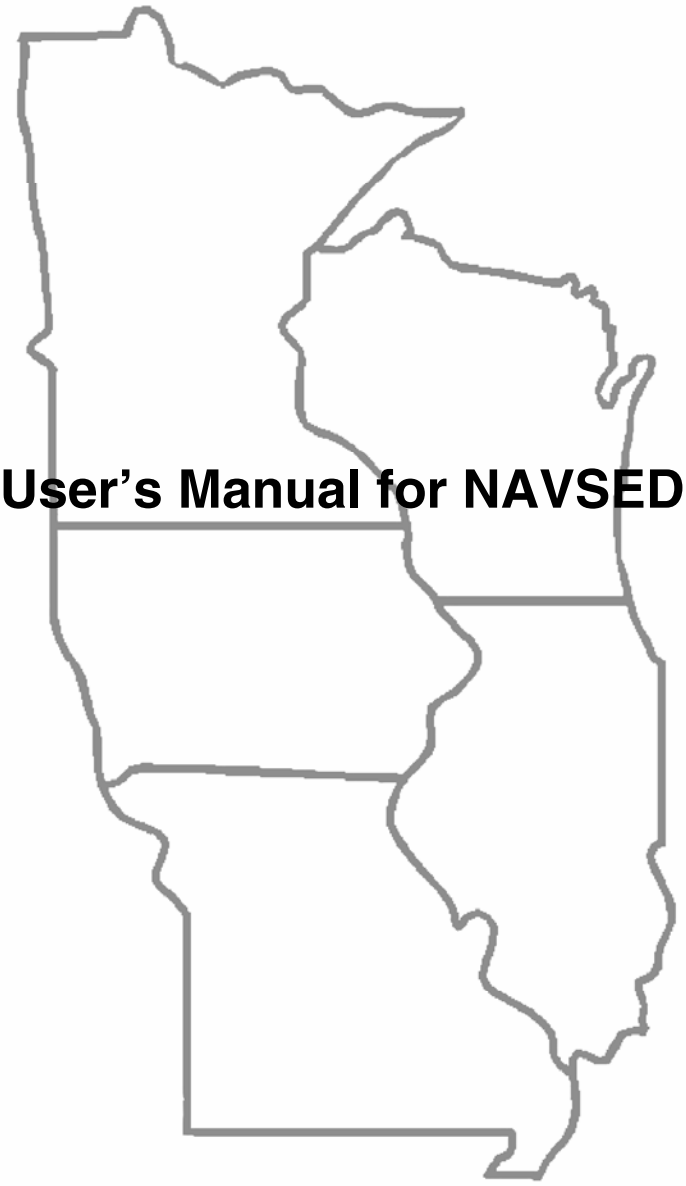
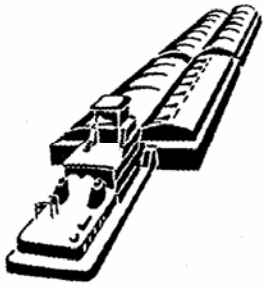
ENV Report 46

Interim Report For The Upper Mississippi River — Illinois Waterway System Navigation Study



User's Manual for NAVSED



December 2004

US Army Corps of Engineers®

Rock Island District
St. Louis District
St. Paul District

Upper Mississippi River - Illinois Waterway
System Navigation Study

ENV Report 46
December 2004

User's Manual for NAVSED

Clay LaHatte and Stephen T. Maynord

Coastal and Hydraulics Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Interim report

Prepared for U.S. Army Engineer District, Rock Island
Rock Island, IL 61204-2004
U.S. Army Engineer District, St. Louis
St. Louis, MO 63103-2833
U.S. Army Engineer District, St. Paul
St. Paul, MN 55101-1638

ABSTRACT: The NAVSED program generates sediment concentration time histories due to passage of shallow draft navigation. This user's manual describes how to run the program interactively or how to set up batch files to run the program. The various input and output file formats are described and example files are presented.

Contents

Preface

The work reported herein was conducted as part of the Upper Mississippi River - Illinois Waterway (UMR-IWW) System Navigation Study. The information generated for this interim effort will be considered as part of the plan formulation process for the System Navigation Study.

The UMR-IWW System Navigation Study is being conducted by the U.S. Army Engineer Districts of Rock Island, St. Louis, and St. Paul under the authority of Section 216 of the Flood Control Act of 1970. Commercial navigation traffic is increasing, and in consideration of existing system lock constraints, will result in traffic delays that will continue to grow in the future. The system navigation study scope is to examine the feasibility of navigation improvements to the Upper Mississippi River and Illinois Waterway to reduce delays to commercial navigation traffic. The study will determine the location and appropriate sequencing of potential navigation improvements on the system, prioritizing the improvements for the 50-year planning horizon from 2000 through 2050. The final product of the System Navigation Study is a Feasibility Report which is the decision document for processing to Congress.

This study was conducted in the Coastal and Hydraulics Laboratory (CHL), U.S. Army Engineer Research and Development Center (ERDC), Vicksburg, MS. The work was conducted under the direction of Mr. Thomas A. Richardson, Director, CHL. This report was written by Mr. W. Clay Lahatte and Dr. Stephen T. Maynord, CHL, ERDC.

At the time of publication of this report, COL James R. Rowan was Commander and Executive Director of ERDC, and Dr. James R. Houston was Director.

1 Overview

General

The NAVSED program generates sediment concentration time histories due to passage of shallow draft navigation on the Upper Mississippi Illinois Waterway. The concentration given is a combined concentration of both fine sediments and sands. The concentration time histories represent the combined influences of prop/bow shear under the barge-tow, shear due to the return currents, and re-suspension due to the short period waves propagating from the barge-tow to the shore.

Input and output are in metric units.

Note: Key elements of the NAVSED program are described in Copeland et al. (2001), Parchure, McAnally, and Teeter (2001), and Maynord et al. (2002).

Pools and Trend Pools

In the development of NAVSED, dimensionless distributions of sediment concentration were developed based on numerical modeling described in Copeland et al. (2001). The numerical modeling was conducted in Trend Pools 8, 26 and 31 (LaGrange). The dimensionless sediment distributions used in NAVSED are found in the file UMMOD.DAT. For pools other than 8, 26, and 31, the sediment distributions are as shown in Table 1.

Table 1
Trend Pool Associations for Upper Mississippi River Study

For Pools	Use Trend Pool
00-17	08
18-27	26
On Illinois Waterway	31 (LaGrange Reach)

Data Files

The NAVSED program requires data input via files, and creates an output file containing results.

Input files

The NAVSED program requires data input via files, and creates an output file containing results.

- `Pxx_yz.OUT` (NAVEFF output).
  There are nine NAVEFF files for combination of three stages and three sailing lines, as described in Maynord et al. (2002). The NAVEFF output provides the bed shear stress, from the velocity changes caused by the tow and the maximum wave height, that is used in the sediment re-suspension calculations.
- `Pxx_PARM.DAT` (tow characteristics)
  See Maynord et al. (2002).
- `Pxx_ELEV.DAT`
  See Maynord et al. (2002).
- `UMMOD.DAT` (common Trend Pool information)
  Contains "Design Curves" which give the lateral and longitudinal distribution of sediment (see Copeland et al. (2001)).
- `Pxx_BWLENGTH.DAT`
  Used when running fixed depth. The appropriate edition of the file must be in the backwater/secondary channel subdirectories (`BW` and `SC`).

where `xx` = pool number.

Output files

Depending upon the option specified at run time, the output file will be either the standard NAVSED output (`.CTH` file), utilizing the original depth from the NAVEFF output, or a fixed-depth NAVSED output (`.CFD` file), utilizing the specified fixed value for depth for all cells. The fixed depth option was used in backwater/secondary channel sedimentation and is documented in Maynord et al. (2002). The `.CFD` files are used as input to the SEDLOAD program package documented in the User's Manual for SEDLOAD, LaHatte and Maynord (2002).

The NAVSED output file will be named as one of the following:

`Pxx_yz.CTH` - based on the NAVEFF depth values

`Pxx_yz.CFD` - based on the specified fixed-depth value

where

xx = Pool number
y = Stage
z = Sailing line

.

2 Installation

Set Up Directories

To successfully run the NAVSED package, a specific directory structure must be used. A general outline showing the directory structure is shown below.

- Project	Root of the project, executables are here.
- Pool01	One of the pool data directories.
- bw	Subdirectory for **backwater** data for the
- sc	Subdirectory for **secondary channel** data
- Pool02	Another pool data directory.
.	
.	… and so on, for all pools in this project.

The subdirectories for backwater and secondary channels are needed only if fixed depth calculations are made for specific backwaters and secondary channels. In this case, data are obtained from the `Pxx_BWLENGTH.DAT` file in each of the `bw` and `sc` subdirectories.

Tip: You can create the data directories as you install the input data files for the pools that you need.

Project directory

The project directory is the root of the project. All data associated with this project will be located in subdirectories of this project directory.

The NAVSED executable program is to be stored in the project directory.

Example:

Project Directory:
`C:\Windows\Desktop\Uppermiss`

Data directory

The Data Directories will hold the pool data. Each pool will have its own data directory.

Example:

One or more Data Directories:
`C:\Windows\Desktop\Uppermiss\pool`**`xx`**

where **`xx`** = pool number

Example:

`C:\Windows\Desktop\Uppermiss\pool13`

Data Files

The NAVSED program requires input via data files located in the Data Subdirectories. You will need all the files described below in order to make a complete automated run.

The data files should be organized as in Table 2.

Table 2
Data File Locations

Data File	Location
NAVEFF output (nine files, `*.out`)	*Project*\poolxx
`Pxx_PARM.DAT` (tow characteristics)	*Project*\poolxx
`Pxx_ELEV.DAT`	*Project*\poolxx
`UMMOD.DAT` (common Trend Pool information) See Copeland et al. (2001)	*Project*\poolxx
`Pxx_BWLENGTH.DAT` (necessary only when running a fixed depth)	*Project*\poolxx\bw *and* *Project*\poolxx\sc

Program

The NAVSED executable program is to be installed in the project directory.

To install the program:

1. If not already existing, create a directory to contain the current project.

Example Project Directory:
`C:\Windows\Desktop\Uppermiss`

2. Copy the NAVSED executable (`.exe`) and batch (`.bat`) files into the project directory.

3 Executing the Program

Types of Runs

The NAVSED program can be run in one of two ways.

To make a run, you should be working in the pool subdirectory in which you are interested. For example,
`C:\Windows\Desktop\Uppermiss\pool13`.

Full run

A full run is an automated means of running the NAVSED program for all nine stage/sailing line scenarios via one command. A full run obtains runtime information from the batch file used to start the process.

Interactive run

At the start of an interactive run, runtime information is supplied via the computer keyboard. Running interactively will process only one of the nine stage/sailing line scenarios per run.

Making a Full Run

The process for running the NAVSED program has been made fairly simple. Once all the input files are in the proper locations, starting a run involves only the following steps:

1. If in Windows, open an MS-DOS command session.
2. Navigate to the directory of the pool in which you are interested. Example:
   `cd C:\Windows\Desktop\Uppermiss\pool13`

3. Start the processes by typing:
   `..\go_navsed xx`
   where *xx* = the pool number
   Example:
   `..\go_navsed 13`

This will execute the batch file `GO_NAVSED.BAT` located in the project directory, i.e., `c:\Windows\Desktop\Uppermiss`, and begin the full NAVSED run.

▸ **Note:** As delivered, the value for depth is fixed at 0.25 meters. Therefore the NAVSED output files will be of type `.CFD` rather than type `.CTH`. If you need to change the depth value used for full automated runs, edit the `GO_NAVSED.BAT` file located in the project directory. To set up to run without a fixed depth (to use the actual depth values from the NAVEFF output files), set the value to zero (0).

Making an Interactive Run

The NAVSED program can be run interactively, meaning the program requires data keyed in from the computer keyboard. An interactive run will process only one stage/sailing line scenario per run.

The runtime prompts are as follows:

1. `Enter the Pool Number (2 characters - Ex: 13):`
   This is the pool number you are working with.
2. `Enter the Flow/Channel (2 characters - Ex: LR):`
   The Stage/Sailing Line combination for the current run.
3. `Enter Fixed Depth value (0 (zero) for original depth):`
   If you want to use a fixed depth value, such as 0.25 meters, enter the numerical value here, otherwise, to use the default values that are read from the NAVEFF output file, enter a zero (0).
4. `Enter the path for all the NAVEFF output files. If same as the default, enter a period (.)`
   Used to specify an alternate location for the NAVEFF output files, such as a CD-ROM, or other hard drive location. Generally the NAVEFF output files should be in the pool directory in which you are working.

4 Output

Final Results

The final output of the NAVSED program will be one of two types of files, depending upon the choice made at run time.

`Pxx_yz.CTH` - based on the NAVEFF depth values

`Pxx_yz.CFD` - based on the specified fixed-depth value

where

`xx` = Pool number
`y` = Stage
`z` = Sailing line.

This file is ready for use with the SEDLOAD package.

Format

The output file can be visualized as consisting of three sections.

Time Frame Section

The Time Frame Section contains the Start Time, End Time, and the number of time steps between the two. The start and end time values are in seconds and are relative to passage of the bow of the tow.

Time Step Intervals Section

Times are in seconds. Concentration is calculated at each time shown. Time intervals are small close to the tow, and large for large times before and after passage.

Data Section

The Data Section contains all the resulting data from the run.

The first line here contains the cellid/traffic scenario/Stage-Sailing Line combination.

The next line contains a write flag, the computed ambient sediment concentration, and the depth at the cell. The write flag is 0 if none of the computed concentration changes due to the tow exceed the computed ambient concentration by 5 mg/L. If the write variable is 0, no concentrations are output for the tow. If the write flag is 1, computed concentrations due to the tow exceed the ambient by 5 mg/L, and concentrations are output for all the time intervals.

An example of the output follows:

Sample of Output File

```
 START TIME  END TIME  NO. TIME STEPS
  -1800.00   8000.00     125
 TIME VALUES (SECONDS)
  -1800.00  -1600.00  -1400.00  -1200.00  -1000.00   -800.00   -600.00   -500.00   -400.00   -300.00
   -275.00   -250.00   -225.00   -200.00   -175.00   -150.00   -125.00   -100.00    -90.00    -80.00
    -70.00    -60.00    -50.00    -40.00    -30.00    -20.00    -10.00      0.00     10.00     20.00
     30.00     40.00     50.00     60.00     70.00     80.00     90.00    100.00    110.00    120.00
    130.00    140.00    150.00    160.00    170.00    180.00    190.00    200.00    210.00    220.00
    230.00    240.00    250.00    260.00    270.00    280.00    290.00    300.00    310.00    320.00
    330.00    340.00    350.00    360.00    370.00    380.00    390.00    400.00    410.00    420.00
    430.00    440.00    450.00    460.00    470.00    480.00    490.00    500.00    525.00    550.00
    575.00    600.00    625.00    650.00    675.00    700.00    725.00    750.00    775.00    800.00
    850.00    900.00    950.00   1000.00   1100.00   1200.00   1300.00   1400.00   1500.00   1600.00
   1700.00   1800.00   1900.00   2000.00   2200.00   2400.00   2600.00   2800.00   3000.00   3200.00
   3400.00   3600.00   3800.00   4000.00   4200.00   4400.00   4600.00   4800.00   5000.00   5500.00
   6000.00   6500.00   7000.00   7500.00   8000.00
15L5290    USSEK HL
 1      1.99      0.25
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      2.00      2.00      2.00
      2.00      2.00      4.23     10.14     15.98     20.81     24.56     24.88     23.01     20.30
     17.59     15.23     13.37     11.95     10.89     10.11      9.48      8.97      8.55      8.20
      7.90      7.63      7.39      7.18      6.99      6.81      6.65      6.51      6.38      6.25
      6.14      6.03      5.93      5.84      5.76      5.68      5.60      5.53      5.46      5.39
      5.33      5.27      5.22      5.16      5.11      5.07      5.02      4.98      4.87      4.78
      4.70      4.62      4.55      4.49      4.43      4.37      4.32      4.27      4.22      4.18
      4.10      4.03      3.97      3.91      3.81      3.72      3.65      3.58      3.52      3.47
      3.43      3.38      3.34      3.31      3.24      3.19      3.13      3.08      3.04      2.99
      2.95      2.91      2.87      2.83      2.79      2.76      2.72      2.69      2.66      2.59
      2.39      2.14      2.05      2.01      2.00
15L5290    USSEO HL
 0      9.25      0.25
15L5290    USSMK HL
 1      1.99      0.25
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99      1.99
      1.99      1.99      1.99      1.99      1.99      1.99      1.99      2.00      2.00      2.00
      2.00      2.00      4.23     10.14     15.98     20.81     24.56     24.88     23.01     20.30
     17.59     15.23     13.37     11.95     10.89     10.11      9.48      8.97      8.55      8.20
      7.90      7.63      7.39      7.18      6.99      6.81      6.65      6.51      6.38      6.25
      6.14      6.03      5.93      5.84      5.76      5.68      5.60      5.53      5.46      5.39
      5.33      5.27      5.22      5.16      5.11      5.07      5.02      4.98      4.87      4.78
      4.70      4.62      4.55      4.49      4.43      4.37      4.32      4.27      4.22      4.18
      4.10      4.03      3.97      3.91      3.81      3.72      3.65      3.58      3.52      3.47
      3.43      3.38      3.34      3.31      3.24      3.19      3.13      3.08      3.04      2.99
      2.95      2.91      2.87      2.83      2.79      2.76      2.72      2.69      2.66      2.59
      2.39      2.14      2.05      2.01      2.00
.
```

References

Copeland, R. R., Abraham, D. D., Nail, G. H., Seal, R., and Brown, G. L. (2001). "Entrainment and transport of sediments by towboats in the Upper Mississippi River and Illinois Waterway, numerical model study," ENV Report 37, U.S. Army Engineer Research and Development Center, Vicksburg, MS.

LaHatte, C. W., and Maynord, S. T. (2004). "Users manual for SEDLOAD: Sediment loading of backwaters and secondary channels," ENV Report 47, U.S. Army Engineer Research and Development Center, Vicksburg, MS.

Maynord, S. T., Knight, S. K., Bourne, S., Graves, M. R., and Landwehr, K. J. (2004). "Upper Mississippi River – Illinois Waterway system models report — Physical effects models," ENV Report 42, U.S. Army Engineer Research and Development Center, Vicksburg, MS.

Parchure, T. M., McAnally, W. H., Jr., and Teeter, A. M. (2001). "Wave-induced sediment resuspension near the shorelines of the upper Mississippi River," ENV Report 20, U. S. Army Engineer Research and Development Center, Vicksburg, MS.

Appendix A
Batch Files

Go_NAVSED.BAT

```
@echo off

REM **** DEPTH VALUE
REM - Set depth=0 to use the actual depth values from the NAVEFF output files.
set depth=0.25

call ..\navsedx.bat %1 HL %depth%
call ..\navsedx.bat %1 HM %depth%
call ..\navsedx.bat %1 HR %depth%
call ..\navsedx.bat %1 ML %depth%
call ..\navsedx.bat %1 MM %depth%
call ..\navsedx.bat %1 MR %depth%
call ..\navsedx.bat %1 LL %depth%
call ..\navsedx.bat %1 LM %depth%
call ..\navsedx.bat %1 LR %depth%
```

NAVSEDX.BAT

```
echo.
echo.

echo Running NAVSED...
echo Pool%1, %2, Depth is %3

echo %1 > navsedx.inp
echo %2 >> navsedx.inp
echo %3 >> navsedx.inp
echo . >> navsedx.inp

echo.

..\navsed < navsedx.inp

Del navsedx.inp
```

Appendix B
Source Code

NAVSED.FOR

```
*     PROGRAM NAVSED
*     VERSION 2.0 LAST MODIFIED 2/19/02 BY CL

      DIMENSION NST(4),NSL(4)
      DIMENSION ST(50,4),SL(10,4)
      DIMENSION TPSHM(10,4),CONM(50,10,4)
      DIMENSION XDAT(2000),YDAT(2000),CASD(50)
      DIMENSION TOUT(125), COUT(1000), CFAMB(10,4)
      DIMENSION TIMEW(2000),CONCW(2000)
      CHARACTER*1 CDUM1,SPEED,SIZE,STAGE
      CHARACTER*2 STGSL,STGSLB
      CHARACTER*5 TRAFFIC,TRAFFICB
      CHARACTER*10 CDUM10,CELLID,CELLID1,CELLIDC,CELLIDV(5000)
      CHARACTER*7 DFNAME,DSNAME,DSNSBS,DSNSRC,DSNFBS,DSNFRC
      CHARACTER*255 PARMDAT,ELEVDAT,NAVFOUT,OUTFILE,BWDAT
      INTEGER COHSED,SEDTYP,COHSEDV,SEDTYPV
        integer bw_cohsed(1000),bw_sedtyp(1000), bw_cohsedx, bw_sedtypx

        character*2 pool, flowchan
        character*10 bw_cellid(1000), bw_cellidx
        real         bw_rm(1000), last_rmile
        character*6  c_rm, cc_rm
        dimension    bw_depth(1000)
        real            bw_depthx
        character PATH*255, naveffpath*255

      DIMENSION GSV(5000),VSV(5000),COHSEDV(5000),SEDTYPV(5000)
C
      DATA TOUT/-1800.0, -1600.0, -1400.0, -1200.0, -1000.0,
     +           -800.0,  -600.0,  -500.0,  -400.0,  -300.0,
     +           -275.0,  -250.0,  -225.0,  -200.0,  -175.0,
     +           -150.0,  -125.0,  -100.0,   -90.0,   -80.0,
     +            -70.0,   -60.0,   -50.0,   -40.0,   -30.0,
     +            -20.0,   -10.0,     0.0,    10.0,    20.0,
     +             30.0,    40.0,    50.0,    60.0,    70.0,
     +             80.0,    90.0,   100.0,   110.0,   120.0,
     +            130.0,   140.0,   150.0,   160.0,   170.0,
     +            180.0,   190.0,   200.0,   210.0,   220.0,
     +            230.0,   240.0,   250.0,   260.0,   270.0,
     +            280.0,   290.0,   300.0,   310.0,   320.0,
     +            330.0,   340.0,   350.0,   360.0,   370.0,
     +            380.0,   390.0,   400.0,   410.0,   420.0,
     +            430.0,   440.0,   450.0,   460.0,   470.0,
     +            480.0,   490.0,   500.0,   525.0,   550.0,
     +            575.0,   600.0,   625.0,   650.0,   675.0,
     +            700.0,   725.0,   750.0,   775.0,   800.0,
     +            850.0,   900.0,   950.0,  1000.0,  1100.0,
     +           1200.0,  1300.0,  1400.0,  1500.0,  1600.0,
     +           1700.0,  1800.0,  1900.0,  2000.0,  2200.0,
     +           2400.0,  2600.0,  2800.0,  3000.0,  3200.0,
     +           3400.0,  3600.0,  3800.0,  4000.0,  4200.0,
     +           4400.0,  4600.0,  4800.0,  5000.0,  5500.0,
     +           6000.0,  6500.0,  7000.0,  7500.0,  8000.0/
C
      TIMES = -1800.0
      TIMEE = 8000.0
      NPTS = 125
      IDSOLD = 0
      IRELEV = 0
      RMILE1 = 0.0
C

      FIXED_DEPTH=0.0
```

```
      WRITE(*,*) '***************** Program NAVSED ****************'
      WRITE(*,*)
      WRITE(*,*) 'This program reads NAVEFF input and output files'
      WRITE(*,*) 'and generates sediment concentration time histories'
      WRITE(*,*) 'due to a boat passage for each cell in the NAVEFF'
      WRITE(*,*) 'output.  The concentration given is a combined'
      WRITE(*,*) 'concentration of both fines and sands.  The'
      WRITE(*,*) 'concentration time histories represent the'
      WRITE(*,*) 'combined influences of prop/bow shear under the'
      WRITE(*,*) 'barge-tow, shear due to the return currents, and'
      WRITE(*,*) 'shear due to the short period waves propagating'
      WRITE(*,*) 'from the barge-tow to the shore'
      WRITE(*,*) '************************************************'


      write(*,*) 'Enter the Pool Number (2 characters - Ex: 13):'
      read (*,*) pool
      write(*,*) 'Enter the Flow/Channel (2 characters - Ex: LR):'
      read (*,*) flowchan
      depthx=0.0
      write(*,*)'Enter Fixed Depth value (0 (zero) for original depth):'
      read (*,*) depthx

C
       pool=trim(adjustr(pool))
       if(pool(1:1) .eq. '') then
         pool(1:1) = '0'
       endif
       open (300,file='ns.tmp',status='unknown')
       write(300,301) pool
 301  format(a2)
      close(300)

c  used when running in debugger...
*      PATH = 'c:\windows\desktop\uppermiss\pool'//trim(pool)//'\'
c  NORMAL use
      PATH=''

      write(*,*) 'Enter the path for all the NAVEFF output files.'
       write(*,*) 'If same as the default, enter a period (.)'
       read(*,38) naveffpath
  38  format(a200)

      PARMDAT=trim(PATH)//'P'//pool//'_PARM.DAT'
      ELEVDAT=trim(PATH)//'P'//pool//'_ELEV.DAT'
      if(naveffpath .eq. '.')then
          NAVFOUT=trim(PATH)//'P'//pool//'_'//flowchan//'.OUT'
      else
          NAVFOUT=
     &    trim(naveffpath)//'P'//pool//'_'//flowchan//'.OUT'
        end if
      if (depthx .ne. 0.0) then
          OUTFILE=trim(PATH)//'P'//pool//'_'//flowchan//'.CFD'
      else
        OUTFILE=trim(PATH)//'P'//pool//'_'//flowchan//'.CTH'
        end if

      i_bwcnt = 0

      if(depthx .GT. 0.0) then
          ! Read Backwater cellids
          BWDAT='bw\P'//pool//'_'//'BWLENGTH.DAT'
          BWDAT=trim(PATH)//BWDAT
          OPEN(UNIT=110,FILE=BWDAT,STATUS='OLD',action='read',err=113)
          do 118
            read(110,*,end=111) bw_cellidx,bw_depthx,
     &                          bw_cohsedx,bw_sedtypx
            ! Check for duplicate cellid and skip if found.
               do k=1,i_bwcnt
              if (bw_cellidx .eq. bw_cellid(k)) then
```

```
              goto 118  ! Read next cellid
            end if
          end do
          i_bwcnt = i_bwcnt + 1
          bw_cellid(i_bwcnt) = bw_cellidx
              bw_depth(i_bwcnt) = bw_depthx
          bw_cohsed(i_bwcnt) = bw_cohsedx
              bw_sedtyp(i_bwcnt) = bw_sedtypx
118      continue
111      close(110)

         ! Read Secondary Channel cellids
         BWDAT='sc\P'//pool//'_'//'BWLENGTH.DAT'
         BWDAT=trim(PATH)//BWDAT
         OPEN(UNIT=110,FILE=BWDAT,STATUS='OLD',action='read',err=114)
         do 119
           read(110,*,end=112) bw_cellidx,bw_depthx,
    &                          bw_cohsedx,bw_sedtypx
           ! Check for duplicate cellid and skip if found.
              do k=1,i_bwcnt
             if (bw_cellidx .eq. bw_cellid(k)) then
               goto 119  ! Read next cellid
             end if
           end do
           i_bwcnt = i_bwcnt + 1
           bw_cellid(i_bwcnt) = bw_cellidx
              bw_depth(i_bwcnt) = bw_depthx
           bw_cohsed(i_bwcnt) = bw_cohsedx
              bw_sedtyp(i_bwcnt) = bw_sedtypx
119      continue
112      close(110)

        goto 115        ! Jump over error messages.
113      write(*,*) 'ERROR! Backwater cell list file was not found',
    &             ' in BW subdirectory.'
         write(*,*) 'Expected ',trim(BWDAT)
         stop
114      write(*,*) 'ERROR! Backwater cell list file was not found',
    &             ' in SC subdirectory.'
         write(*,*) 'Expected ',trim(BWDAT)

115      continue
         write(*,*)
         write(*,*) 'Using a fixed depth value of ', depthx
         write(*,*) i_bwcnt, ' unique cellids to be used...'

       else
         write(*,*)
         write(*,*) 'Using original depth value.'
       end if

      write(*,*)


       open (300,file='ns.tmp',status='unknown')
       read(300,302,err=306) IDS
302   format(i2)

       goto 307

306   IDS=3
      write(*,*) 'Using trend pool 31 (LG)'
      goto 305

307   if((IDS .ge. 28) .and. (IDS .le. 89)) then
          IDS=3
          write(*,*) 'Using trend pool 31 (LG)'
          goto 305
       end if
      if((IDS .le.17).or.(IDS .eq. 95))then
          IDS=2
```

```
          write(*,*) 'Using trend pool 13'
          goto 305
        end if
       if(((IDS .ge. 18) .and. (IDS .le.27)).or.(IDS.eq.96))then
          IDS=1
          write(*,*) 'Using trend pool 26'
          goto 305
        end if

 305  close(300, status='delete')

*      WRITE(*,*) 'POOL SPECIFICATION'
*      WRITE(*,*)
*      WRITE(*,*) 'Select one of the 3 trend pools.'
*      WRITE(*,*) 'For pools 0-17 use 2 (pool08).'
*      WRITE(*,*) 'For pools 18-27 use 1 (pool26).'
*      WRITE(*,*) 'For pools on Illinois use 3 (LaGrange).'
*      WRITE(*,*) '  to select POOL26, input a 1'
*      WRITE(*,*) '  to select POOL8, input a 2'
*      WRITE(*,*) '  to select LAGRANGE, input a 3'
*      READ(*,*) IDS
       IDS = IDS*10
C
      OPEN(UNIT=5,FILE=OUTFILE,FORM='FORMATTED',STATUS='UNKNOWN')
      OPEN(UNIT=10,FILE='..\UMMOD.DAT',STATUS='OLD')
      OPEN(UNIT=50,FILE=PARMDAT,FORM='FORMATTED',STATUS='OLD')
      OPEN(UNIT=60,FILE=ELEVDAT,FORM='FORMATTED',STATUS='OLD')
      OPEN(UNIT=70,FILE=NAVFOUT,FORM='FORMATTED',STATUS='OLD')
C
      if(depthx .GT. 0.0) then
*        OPEN(UNIT=110,FILE=BWDAT,STATUS='OLD',action='read',err=112)
         OPEN(UNIT=210,FILE='x.tmp',STATUS='unknown')
*         do i=1,1000
          do i=1,i_bwcnt
*           read(110,*,end=111) bw_cellid(i),bw_depth(i),
*     &                         bw_cohsed(i),bw_sedtyp(i)
              i_pos=scan(bw_cellid(i),'LR')
            c_rm=trim(bw_cellid(i)(i_pos+1:i_pos+6))
            c_rm=adjustr(c_rm)
            cc_rm=c_rm(6:6)
            c_rm=trim(c_rm(2:5))//'.'//cc_rm
            c_rm=adjustl(c_rm)
            rewind(210)
            write(210,211) c_rm
 211       format(a)
           rewind(210)
             read(210,*) bw_rm(i)

          end do
* 111     close(110)
          close(210)
*         i_bwcnt=i-1
*         goto 113
* 112      write(*,*) 'ERROR! Backwater cell list file was not found.'
*          write(*,*) 'Expected ',trim(BWDAT)
*          stop
        end if
C
      READ(50,55) CDUM10
      READ(50,56) CDUM1,VBRGS
      READ(50,56) CDUM1,VBRGM
      READ(50,56) CDUM1,VBRGF
      READ(50,*)
      READ(50,55) CDUM10
      READ(50,57) CDUM1,BWIDTHL,BLENGTHL
      READ(50,57) CDUM1,BWIDTHS,BLENGTHS
      READ(50,57) CDUM1,BWIDTHM,BLENGTHM
      READ(50,57) CDUM1,BWIDTHB,BLENGTHB
C
  55  FORMAT(A)
  56  FORMAT(A,1X,F4.2)
```

```
  57  FORMAT(A,1X,F5.2,1X,F6.2)
C
      NLINES = 1
      NCOUNT = 0
      IONCE = 0
      IRECORD = 0
        RMILEOLD = 0.0
        i_writeout=0

      i_good_rmile=0
      last_rmile = 0.0

      DO II = 1, 90000000
        INODAT = 0
        NCOUNT = NCOUNT + 1
        IF (NCOUNT .EQ. NLINES) IHAVE = 0
 241    READ(70,*,END=5000) RMILE,CDUM1,SPEED,SIZE,CDUM1,CDUM1,
     +                      STAGE,CDUM1,TRAFFIC,STGSL,CELLID,DEPTH,
     +                      DUM,DUM,DTSL,HMAX,DUM,TPSHR,DUM,DUM,TAMB
        if(depthx .le. 0.0) goto 455
c       Only care about certain river miles, from cell depth file.
          if(last_rmile .eq. RMILE) then
            if(i_good_rmile .ne. 0) goto 455
            goto 241
          end if
          last_rmile=RMILE
        i_good_rmile = 0
          do i=1,i_bwcnt
            if(RMILE .eq. bw_rm(i)) then
              i_good_rmile = -1
              goto 455
            end if
        end do
        goto 241
c

*          if(depthx .GT. 0.0) then
*            depth_orig=DEPTH
*            do i=1,i_bwcnt
*               if(trim(CELLID).eq.trim(bw_cellid(i)))then
*                 i_writeout=-1
*                        if(DEPTH .gt. depthx) then
*                      DEPTH=depthx
*                   goto 455
*                 end if
*               end if
*             end do
*          else
*                i_writeout=-1
*              end if


 455      IF(RMILEOLD.NE.RMILE)THEN
           WRITE(*,*) RMILE
          END IF
          RMILEOLD=RMILE

C 75     FORMAT(F5.1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A1,1X,A5,
C     +         1X,A2,1X,A10,1X,F6.2,1X,F7.3,1X,F7.3,1X,F7.1,1X,F7.3,
C     +         1X,F9.4,1X,F9.4,1X,F9.4,1X,F12.7,1X,F12.7)

        if(HMAX .lt. 0.0) then
         if(HMAX .ne. -9.0)then
          HMAX=abs(HMAX)
         end if
        end if

        DTSL = ABS(DTSL)
        RMILEB = RMILE
        TRAFFICB = TRAFFIC
        STGSLB = STGSL
```

```
        IF (IHAVE .EQ. 0) THEN
          BACKSPACE(70)
          NLINES = 0
          DTSLMIN = 2000.0
          DO JJ = 1, 100000
            NLINES=NLINES+1
            READ(70,*,END=110) RMILE,CDUM1,SPEED,SIZE,CDUM1,CDUM1,
     +                 STAGE,CDUM1,TRAFFIC,STGSL,CELLID,DEPTH,DUM,
     +                 DUM,DTSL,HMAX,DUM,TPSHR,DUM,DUM,TAMB
 456    if(HMAX .lt. 0.0) then
          if(HMAX .ne. -9.0)then
           HMAX=abs(HMAX)
          end if
        end if
           DTSL = ABS(DTSL)
            RMILED=ABS(RMILE-RMILEB)
            IF (RMILED .GT. 0.01 .OR. TRAFFIC .NE. TRAFFICB
     +         .OR. STGSL .NE. STGSLB) GO TO 110
            IF (DTSL.LT.DTSLMIN) THEN
              DTSLMIN=DTSL
              TPSHB=TPSHR
              DASL=DEPTH
              CELLIDC=CELLID
            END IF
          END DO
  110     CONTINUE
          IHAVE = 1
          DO JJ = 1, NLINES
            BACKSPACE(70)
          END DO
          READ(70,*,END=5000) RMILE,CDUM1,SPEED,SIZE,CDUM1,CDUM1,
     +                 STAGE,CDUM1,TRAFFIC,STGSL,CELLID,DEPTH,DUM,
     +                 DUM,DTSL,HMAX,DUM,TPSHR,DUM,DUM,TAMB
 457      if(HMAX .lt. 0.0) then
         if(HMAX .ne. -9.0)then
          HMAX=abs(HMAX)
         end if
        end if
          DTSL = ABS(DTSL)
          NCOUNT = 1
        END IF
        IF (ABS(RMILE-RMILE1) .GT. 0.01) IRELEV = 0
        IF (HMAX .LT. -1.0)  HMAX  = 0.0
        IF (TPSHR .GT. 9000.0 .OR. TAMB .GT. 9000.0) INODAT = 1
        IF (INODAT .EQ. 0) THEN
C
        IF (IRELEV .EQ. 0) THEN
          REWIND(60)
          DO JJ = 1, 100000
            READ(60,65) RTEST,CELLID1,DUM,DUM,DUM,DUM,DUM,DUM,
     +                  GS,VS,COHSED,SEDTYP
            RMILED = ABS(RMILE-RTEST)
            IF (RMILED .LT. 0.01) GO TO 120
          END DO
   65     FORMAT(F5.1,A10,F12.1,F12.1,F8.1,F8.3,
     +           F8.3,F8.3,F8.3,F8.3,I6,I6)
  120     BACKSPACE(60)
          RMILE1 = RTEST
          IVLEN = 0
*           DO KJ = 1,5000
*            GSV(KJ) = 0.0
*                VSV(KJ) = 0.0
*                COHSEDV(KJ) = 0.0
*                SEDTYPV(KJ) = 0.0
*           END DO
          DO JJ = 1, 100000
            READ(60,65,END=130) RTEST,CELLIDV(JJ),DUM,DUM,DUM,DUM,
     +      DUM,DUM,GSV(JJ),VSV(JJ),COHSEDV(JJ),SEDTYPV(JJ)
            RMILED = ABS(RMILE-RTEST)
            IF (RMILED .GT. 0.01) GO TO 130
            IVLEN = IVLEN + 1
```

```
          END DO
 130      CONTINUE
          IRELEV = 1
        END IF
        DO JJ = 1, IVLEN
          IF (CELLIDV(JJ) .EQ. CELLID) THEN
            GS = GSV(JJ)
            VS = VSV(JJ)
            COHSED = COHSEDV(JJ)
            SEDTYP = SEDTYPV(JJ)
              do i_bw=1,i_bwcnt
                if(trim(CELLID).eq.trim(bw_cellid(i_bw)))then
                  COHSED = bw_cohsed(i_bw)
                 SEDTYP = bw_sedtyp(i_bw)
                  goto 1234
                       end if
              end do
1234      END IF
          IF (CELLIDV(JJ) .EQ. CELLIDC) THEN
            GSC = GSV(JJ)
            VSC = VSV(JJ)
          END IF
        END DO
C
        GS=GS/1000.
        VS=VS/100.
        GSC=GSC/1000.
        VSC=VSC/100.
C
        IF (SPEED .EQ. 'S') THEN
          VBRG=VBRGS
        ELSE IF (SPEED .EQ. 'M') THEN
          VBRG=VBRGM
        ELSE IF (SPEED .EQ. 'F') THEN
          VBRG=VBRGF
        END IF
        IF (SIZE .EQ. 'L') THEN
          BLENGTH=BLENGTHL
          BWIDTH=BWIDTHL
        ELSE IF (SIZE .EQ. 'S') THEN
          BLENGTH=BLENGTHS
          BWIDTH=BWIDTHS
        ELSE IF (SIZE .EQ. 'M') THEN
          BLENGTH=BLENGTHM
          BWIDTH=BWIDTHM
        ELSE IF (SIZE .EQ. 'B') THEN
          BLENGTH=BLENGTHB
          BWIDTH=BWIDTHB
        END IF
        IF (STAGE .EQ. 'H') THEN
          IDSF = 1
        ELSE IF (STAGE .EQ. 'M') THEN
          IDSF = 2
        ELSE IF (STAGE .EQ. 'L') THEN
          IDSF = 3
        END IF
      IDS = IDS + IDSF
C Now supply values to some variables based on the pool-flow selected
      IF (IDS .EQ. 11) THEN
        DSNSBS = 'p26hq  '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 140.0
        NPASS = 1
      ELSE IF (IDS .EQ. 12) THEN
        DSNSBS = 'p26mq  '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 140.0
        NPASS = 1
      ELSE IF (IDS .EQ. 13) THEN
```

```
        DSNSBS = 'p26lq  '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 140.0
        NPASS = 1
      ELSE IF (IDS .EQ. 21) THEN
        DSNSBS = 'p8hqbs '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 167.0
        NPASS = 1
      ELSE IF (IDS .EQ. 22) THEN
        DSNSBS = 'p8mqbs '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 140.0
        NPASS = 1
      ELSE IF (IDS .EQ. 23) THEN
        DSNSBS = 'p8lqbs '
        DSNSRC = 'p8lqrc '
        GSM = 0.0005
        VSM = 0.06
        TSHIFT = 103.0
        NPASS = 2
      ELSE IF (IDS .EQ. 31) THEN
        DSNSBS = 'lghqsbs'
        DSNSRC = 'lghqsrc'
        DSNFBS = 'lqhqfbs'
        DSNFRC = 'lghqfrc'
        GSM = 0.0001
        VSM = 0.00658
        TSHIFT = 213.0
        NPASS = 4
      ELSE IF (IDS .EQ. 32) THEN
        DSNSBS = 'lgmqsbs'
        DSNSRC = 'lglqsrc'
        DSNFBS = 'lgmqfbs'
        DSNFRC = 'lgmqfrc'
        GSM = 0.0001
        VSM = 0.00658
        TSHIFT = 219.0
        NPASS = 4
      ELSE IF (IDS .EQ. 33) THEN
        DSNSBS = 'lglqsbs'
        DSNSRC = 'lglqsrc'
        DSNFBS = 'lglqfbs'
        DSNFRC = 'lglqfrc'
        GSM = 0.0001
        VSM = 0.00658
        TSHIFT = 175.0
        NPASS = 4
      END IF
      INOREAD = 0
      IF (IDS .EQ. IDSOLD) INOREAD = 1
      IDSOLD = IDS
      IDS = IDS - IDSF
C let GS = GSM and VS=VSM if no data
      IF (GS .GT. 9.0) GS=GSM
      IF (GSC .GT. 9.0) GSC=GSM
      IF (VS .GT. 90.0) VS=VSM
      IF (VSC .GT. 90.0) VSC=VSM
c Done to set grain size=grainsize used in trend pool (pool 31 (LG))
       if(IDS.ge.31 .and. IDS.le.33)then
         GS=GSM
         GSC=GSM
         VS=VSM
         VSC=VSM
       end if
c
C Now read data from the data set
C NOTE: go ahead and shift the ST values so that
```

```
C       time zero corresponds to the time of bow passage
      IF (INOREAD .EQ. 0) THEN
        DO L = 1,NPASS
          IF (L .EQ. 1) DSNAME = DSNSBS
          IF (L .EQ. 2) DSNAME = DSNSRC
          IF (L .EQ. 3) DSNAME = DSNFBS
          IF (L .EQ. 4) DSNAME = DSNFRC
           DO K = 1, 19
            READ(10,5) DFNAME
 5          FORMAT(A)
            READ(10,*) NST(L),NSL(L)
            READ(10,*) (SL(I,L),I=1,NSL(L))
            READ(10,*) (TPSHM(I,L),I=1,NSL(L))
            READ(10,*) (CFAMB(I,L),I=1,NSL(L))
            DO I=1,NST(L)
              READ(10,*) ST(I,L),(CONM(I,J,L),J=1,NSL(L))
              ST(I,L) = ST(I,L) + TSHIFT
c
              if(L.eq.3)then
c zero out boat shear fines on LaGrange
                 do LL=1,NSL(L)
                   CONM(I,LL,L) = CONM(I,LL,L)*0.67
                        end do
                      end if
c reduce return current fines on LG by 50%
              if(L.eq.4)then
                 do LL=1,NSL(L)
                   CONM(I,LL,L) = CONM(I,LL,L)*0.67
                        end do
                      end if
c
                 END DO
            IF (DFNAME .EQ. DSNAME) GO TO 10
           END DO
 10       CONTINUE
          REWIND (10)
        END DO
      END IF
C     initialize the output concentration
      DO I = 1, 1000
        COUT(I) = 0.0
      END DO
C     now loop on all scenarios for the given pool
      DO L = 1, NPASS
C now do linear interpolation to find the peak shear at the given lateral distance
C for return currents
        IF (L .EQ. 2 .OR. L .EQ. 4) THEN
          DO I = 1, NSL(L)
            XDAT(I) = SL(I,L)
            YDAT(I) = TPSHM(I,L)
          END DO
          CALL VALUE(NSL(L),DTSL,XDAT,YDAT,0.0,1.0,1.0,TPSHM1)
        ELSE
          TPSHM1 = TPSHM(1,L)
        END IF
C now do linear interpolation to find design curve at the
C requested distance from the sailing line
        DO I = 1, NST(L)
          DO J = 1, NSL(L)
            XDAT(J) = SL(J,L)
            YDAT(J) = CONM(I,J,L)
          END DO
          CALL VALUE(NSL(L),DTSL,XDAT,YDAT,0.0,1.0,1.0,CASD(I))
        END DO
        IF (L .EQ. 1) THEN
C now find ratio for sands
          RATIO =  CSAMB(TPSHB,GSC,VSC)/CSAMB(TPSHM1,GSM,VSM)
          RPD = 1.0
        ELSE IF (L .EQ. 2) THEN
          RATIO =  CSAMB(TPSHR,GSC,VSC)/CSAMB(TPSHM1,GSM,VSM)
          DTSLC = (BWIDTH+20.0)/2.
```

```
          IF (DTSL .LT. DTSLC) RATIO = 0.0
          RPD = 1.0
        ELSE IF (L .EQ. 3) THEN
C now find ratio for clays
          TC = 0.3
          IF (IDSF .EQ. 1) TC = 1.2
          RATIO = (TPSHB - TC)/(TPSHM1 - TC)
          IF (RATIO .LT. 0.0) RATIO = 0.0
          RPD = AMAX1((1. - TAMB/TC),0.0)
        ELSE
          TC = 0.3
          IF (IDSF .EQ. 1) TC = 1.2
          RATIO = (TPSHR - TC)/(TPSHM1 - TC)
          IF (RATIO .LT. 0.0) RATIO = 0.0
          IF (DTSL .LT. 25.0) RATIO = 0.0
          RPD = AMAX1((1. - TAMB/TC),0.0)
        END IF
C now multiply CASD by the ratio
        IF (L .EQ. 1) THEN
          CAMB = CSAMB(TAMB,GSC,VSC)
        ELSE IF (L .EQ. 3) THEN
          DO J = 1, NSL(L)
            XDAT(J) = SL(J,L)
            YDAT(J) = CFAMB(J,L)
          END DO
          CALL VALUE(NSL(L),DTSL,XDAT,YDAT,0.0,1.0,1.0,CFINES)
          CAMB = CFINES
        ELSE
          CAMB = 0.0
        END IF
        DO I = 1, NST(L)
          CASD(I)=CASD(I)*RATIO
        END DO
C now interpolate onto curve
        DO I = 1, NST(L)
          XDAT(I) = ST(I,L)
          YDAT(I) = CASD(I)
        END DO
        DO I = 1, NPTS
          VS1=VSC
          IF (L .GT. 2) VS1 = 0.0025
          CALL VALUE(NST(L),TOUT(I),XDAT,YDAT,VS1,DEPTH,RPD,COUT1)
          COUT(I) = COUT(I) + COUT1 + CAMB
        END DO
      END DO
C now do contribution from waves
          if(depthx .GT. 0.0) then
            depth_orig=DEPTH
            do i=1,i_bwcnt
                if(trim(CELLID).eq.trim(bw_cellid(i)))then
                  i_writeout=-1
                  if(DEPTH .gt. depthx) then
                       DEPTH=depthx
                    goto 444
                  end if
                end if
              end do
          else
                 i_writeout=-1
          end if
*
444   tvrmax = 1.0
      IF (DEPTH .LE. 1.5 .AND. COHSED .LT. 3) THEN
        DO I = 1, 2000
          TIMEW(I) = 0.0
          CONCW(I) = 0.0
        END DO
        CALL WAVECONC(SEDTYP,DEPTH,HMAX,TVRMAX,BLENGTH,VBRG,
     +                NUMPNTS,TIMEW,CONCW)
        TWA = DTSL/(SQRT(9.806*(DEPTH+DASL)*0.5))
        DO I = 1, NUMPNTS
```

```
          XDAT(I) = TIMEW(I) + TWA
          YDAT(I) = CONCW(I)
            if(sedtyp.eq.1)then
              if(YDAT(I).GT.3000.0) YDAT(I) = 3000.0
            else if(sedtyp.eq.2) THEN
              IF(YDAT(I).GT.1500.0) YDAT(I) = 1500.0
            END IF
        END DO
        COUT1 = 0.0
        DO I = 1, NPTS
          CALL VALUE(NUMPNTS,TOUT(I),XDAT,YDAT,VS,DEPTH,1.0,COUT1)
          IF(DEPTHX .GT. 0.0) THEN
            if(depth_orig .gt. depthx) then
              COUT1=COUT1*(depthx**2/depth_orig**2)
            end if
          END IF
          COUT(I) = COUT(I) + COUT1
        END DO
      END IF
C now write out results
      NLNS=NPTS/10
      NRM=MOD(NPTS,10)
      IF (IONCE .EQ. 0) THEN
        IONCE = 1
        WRITE(5,*) 'START TIME  END TIME  NO. TIME STEPS'
        WRITE(5,13) TIMES,TIMEE,NPTS
        WRITE(5,*) 'TIME VALUES (SECONDS)'
        K=1
        DO I = 1, NLNS
          WRITE(5,15) (TOUT(J),J=K,K+9)
          K=K+10
        END DO
        IF (NRM .GT. 0) THEN
          NRMM1=NRM-1
          WRITE(5,15) (TOUT(J),J=K,K+NRMM1)
        END IF
      END IF
      if(i_writeout .ne. 0) then
        i_writeout=0
         WRITE(5,14) CELLID,TRAFFIC,STGSL
        IWRITE = 0
        DO I = 1, NPTS
          IF ((COUT(I)-COUT(1)) .GE. 5.0) IWRITE = 1
        END DO
        WRITE(5,16) IWRITE, COUT(1),DEPTH
        IF (IWRITE .EQ. 1) THEN
          K=1
          DO I = 1, NLNS
            WRITE(5,15) (COUT(J),J=K,K+9)
            K=K+10
          END DO
          IF (NRM .GT. 0) THEN
            NRMM1=NRM-1
            WRITE(5,15) (COUT(J),J=K,K+NRMM1)
          END IF
        END IF
       end if
C the following end if is for INODAT loop
      END IF
 13   FORMAT(2F10.2,I8)
 14   FORMAT(A,1X,A,1X,A)
 15   FORMAT(10F10.2)
 16   FORMAT(I2,2F10.2)
      IF (INODAT .EQ. 0) IRECORD = IRECORD + 1
*      IF (INODAT .EQ. 0) WRITE(*,*) 'WROTE RECORD ', IRECORD
*     &                             ,' FOR LINE ', II
*      IF (INODAT .EQ. 1) WRITE(*,*) 'NO DATA; NO RECORD WRITTEN'
*     &                              ,' FOR LINE ', II
      END DO
C
 5000 CONTINUE
```

```
      WRITE(*,*)
      WRITE(*,*) 'Results file contains a concentration time history'
      WRITE(*,*) 'for each entry in the .OUT file.  The entries are'
      WRITE(*,*) 'identified by CELL_ID, TRAFFIC, and STG_SL values.'
      WRITE(*,*) 'The time values for the time history are given'
      WRITE(*,*) 'at the beginning of the results file'
      WRITE(*,*)
      STOP
      END

      SUBROUTINE VALUE(NWP,DLOC,XDAT,YDAT,VS,DEPTH,RPD,VOUT)
C
      INTEGER NWP
      REAL DLOC,XDAT,YDAT
      DIMENSION XDAT(2000),YDAT(2000)
      TC=-VS*RPD/DEPTH
      VALUET=YDAT(1)
      IF(XDAT(1) .GT. DLOC)
     +VALUET = YDAT(1)*EXP(TC*(XDAT(1)-DLOC))
      DO I =1, NWP-1
        IF (XDAT(I) .LE. DLOC .AND. XDAT(I+1) .GT. DLOC)
     +  VALUET = (DLOC-XDAT(I))*(YDAT(I+1)-YDAT(I))
     +          /(XDAT(I+1)-XDAT(I)) + YDAT(I)
      END DO
      IF (XDAT(NWP) .LE. DLOC)
     +VALUET = YDAT(NWP)*EXP(TC*(DLOC-XDAT(NWP)))
      VOUT = VALUET
      RETURN
      END



      REAL FUNCTION CSAMB(TAU,GS,VS)
      REAL TAU,GS,VS
C
      A = 1.3E-7
      USTAR = SQRT(TAU/1000.)
      REP = SQRT(9.806*1.65*GS)*GS/1.306E-6
      ZU = (USTAR*REP**0.6)/VS
      UST1 = AMAX1((USTAR/VS),0.5)
      RO = 1 + 31.5*UST1**(-1.46)
      CB = ((1.3E-7)*ZU**5.)/(1. + ((1.3E-7)*ZU**5.)/0.3)
      CSAMB = CB*2.65*1000000./RO
      END
*
          subroutine waveconc(sedtyp,hDEP,hmax,tvrmax,tl,tvelg,
     *                      numpnts, time, conc)
*
*     Common (Global) values
*
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *           MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
      common/flow/uu,QN
      common/tide/qqn,umax,ptide
      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
      common/mr/gg1,gg2,uu2
      common/out/WSC0,WS0,WSA,WSB,WSM,WSN,DSK1,DSK2
      common/cal/suspmas,eromas
      common/bed/ dzb,TAUDEP,BDP1,BDP2,RHBBAR,PHIC,tcp1,tcp2,
     *            ERSMAX,ERP1,ERP2,tauero
        common/vest/ jn,zi,ci
*
      dimension time(2000), conc(2000)
*
      character DESCR*60
        real*8 dt
      real*8 tt,TRATIO
        real*8 vr,VRATIO
        real M0, kk, kkss
      dimension zi(500),ci(500)
```

```
*
       INTEGER WAVEPERIOD
       INTEGER numpnts!number of data points calculated by wavavg.
       INTEGER sedtyp  !1=soft, 2=medium, 3=hard
*      INTEGER tdir    !Tow direction, 0=down (with), 1=up (against the current)

*
*      Determines whether to use a slope for calculating
*       h in range 4, in subroutine wavavg. 1=slope, 0=no slope
       INTEGER r4slopeF
*
*     Total number of ratio array elements,
*      associated with time history of return velocity.
       INTEGER   RATOT
       dimension TRATIO(18)
       dimension VRATIO(18)
*
      CHARACTER FILEWAVE*256, VESTOUT*256, ACKEROUT*256
*
       dtrun = 97.0    !minutes - end time
       dt    = 6.0             !seconds - time step
      dtout = 0.1              !minutes - output interval
*
*     * * * sample values for vestuns, wavavg, acker
*     sedtyp = 3               !1=soft, 2=medium, 3=hard
*     h0 = .5                  !meters - averaged water depth in the cell
*     hmax = .5        !m - (h2) maximum vessel induced wave height
*      tl = 100.               !meters - length of tow barge
*      tvelg = 1.0             !m/sec - tow speed relative to the ground
*      tvrmax = 0.5    !m/sec - maximum magnitude of return velocity
*      tdir = 1                !0=downbound (with), 1=upbound (against the current)
*      tbp                             !sec - time for barge passage (dl/tvelg)
*      tt                              !sec - tow time
*
*        For Ackers subroutine...
*      d50 = .001              !m - median grain diameter
*      va = 1.0                !m/sec - ambient current velocity magnitude
*      vr = 0.2                !m/sec - return velocity magnitude
*
      FILEWAVE='filewave.dat'
      VESTOUT = 'vestsub.out'
       ACKEROUT = 'acker.out'
*
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*
*     THIS ROUTINE LIMITS THE WAVE CONCENTRATION TO ABOUT 1000 MG/L
*     THIS BECAME A PROBLEM WHEN WE HAD SHALLOW DEPTHS IN CELLS SUCH
*     AS A 0.12 M DEPTH WITH A 0.10 M WAVE RESULTING IN LARGE MG/L
*     WITHOUT THIS LIMITING DEPTH
      H0=HDEP
       TESTDEP=1.2*HMAX
       IF(H0.LT.TESTDEP) H0=TESTDEP
*
       DESCR = 'DATA FOR FLUID MUD UNDER WAVES AND WEAK CURRENTS'
       !Discretization Parameters
       M0 = 1.0
       NGRIDS = 5
       !Time Parameters
       STT = 0.0
       !Settling Velocity Parameters
       WSC0 = 0.1
       WS0 = 2.1E-05
       WSA = 0.16
       WSB = 7.0
       WSM = 1.5
       WSN = 1.33
       !Stabilized Diffusion Constants
       DSK1 = 0.5      !alpha0
       DSK2 = 0.33     !beta0
       !Define: Program Logic
       MODE = 3     ! SELECTS WAVES + WEAK CURRENTS
```

```
        !Deposition rate
        DSP = 1.0
        !Current Hydrodynamics
        UU = 0.000001
*       UU = 0.1       ! VALUE USED IN PARCHURE REPORT
        QN = 0.02
        !Wave Hydrodynamics
        WDIFFK = 0.2
        !Density
        RHOSED = 2200.0
        RHOW = 1000.0
        !Define: Selection for bottom sediment
        MCHOO = 2
        !Critical Stress of deposition
        TAUDEP = 0.1
        !Wave friction coefficient
        WFRIC = 0.015
        !Bed density parameters
        BDP1 = 0.794
        BDP2 = -0.45
        RHBBAR = 350.0
        !Critical stress parameters
        PHIC = 0.05
        !Erosion rate coefficient parameters
        ERSMAX = 0.2
        ERP1sft = 7.0
        ERP2sft = 0.35
        ERP1MED = 6.5
        ERP2MED = 0.35
        ERP1HRD = 10.0
        ERP2HRD = 0.1
        !Initial concentration points
        jn = 2
        !Initial concentration profile
        zi(1) = 0.2
        zi(2) = 1.0
        ci(1) = 0.0
        ci(2) = 0.0
*
*
        RATOT = 18
*        ratio of tt/tbp - used to calculate tt given tbp
        TRATIO(1) = -0.6
        TRATIO(2) = -0.4
        TRATIO(3) = -0.2
        TRATIO(4) = 0.0
        TRATIO(5) = 0.1
        TRATIO(6) = 0.2
        TRATIO(7) = 0.3
        TRATIO(8) = 0.4
        TRATIO(9) = 0.5
        TRATIO(10) = 0.6
        TRATIO(11) = 0.7
        TRATIO(12) = 0.8
        TRATIO(13) = 0.9
        TRATIO(14) = 1.0
        TRATIO(15) = 1.2
        TRATIO(16) = 1.4
        TRATIO(17) = 1.6
        TRATIO(18) = 1.7
*        ratio of vr/tvrmax - used to calculate vr given tvrmax
        VRATIO(1) = 0.0
        VRATIO(2) = 0.02
        VRATIO(3) = 0.1
        VRATIO(4) = 0.21
        VRATIO(5) = 0.34
        VRATIO(6) = 0.5
        VRATIO(7) = 0.64
        VRATIO(8) = 0.77
        VRATIO(9) = 0.83
        VRATIO(10) = 0.86
```

```
        VRATIO(11) = 0.9
        VRATIO(12) = 0.95
        VRATIO(13) = 1.0
        VRATIO(14) = 0.92
        VRATIO(15) = 0.65
        VRATIO(16) = 0.36
        VRATIO(17) = 0.07
        VRATIO(18) = 0.0
*
        WAVEPERIOD = 2
*
      RHO = 1000      !kg/l
        VNU = 0.000001 !? - kinematic viscosity of water
*
        RHOSFT = 1600.0
        RHOMED = 1900.0
        RHOHRD = 2000.0
        TCP1SF = 1.0
        TCP1MD = 7.0
        TCP1HD = 6.0
        TCP2SF = 1.5
        TCP2MD = 1.5
        TCP2HD = 1.0
*
*      Determines whether to use a slope for calculating
*       h in range 4, in subroutine wavavg. 1=slope, 0=no slope
        r4slopeF = 0
*
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
*
        if (h0 .GT. 1.5) then
                h0 = 1.5
        end if
*
*
        tbp = tl / tvelg
        do 121 i = 1, RATOT     !total number of ratios defined in TRATIO() and VRATIO()
          call calratio(TRATIO(i), tbp, tt)       !tt is the value "returned"
          call calratio(VRATIO(i), tvrmax, vr)  !vr is the value "returned"
  121   continue
*
        close(1)
*
*
        if (sedtyp .EQ. 1) then
*               Soft
                RHOM = RHOSFT
                tcp1 = TCP1SF
                tcp2 = TCP2SF
            ERP1 = ERP1SFT
            ERP2 = ERP2SFT
*               --- vestuns+acker
                call wavavg(hmax,dtrun,WAVEPERIOD,r4slopeF,FILEWAVE,numpnts)
            call vestuns(h0, FILEWAVE, VESTOUT, time, conc)
        else if (sedtyp .EQ. 2) then
*               Medium
                RHOM = RHOMED
                tcp1 = TCP1MD
                tcp2 = TCP2MD
            ERP1 = ERP1MED
            ERP2 = ERP2MED
*               --- vestuns+acker
                call wavavg(hmax,dtrun,WAVEPERIOD,r4slopeF,FILEWAVE,numpnts)
            call vestuns(h0, FILEWAVE, VESTOUT, time, conc)
        else if (sedtyp .EQ. 3) then
*               Hard
                RHOM = RHOHRD
                tcp1 = TCP1HD
                tcp2 = TCP2HD
            ERP1 = ERP1HRD
            ERP2 = ERP2HRD
```

```
*              --- vestuns+acker
               call wavavg(hmax,dtrun,WAVEPERIOD,r4slopeF,FILEWAVE,numpnts)
           call vestuns(h0, FILEWAVE, VESTOUT, time, conc)
       end if
*
*
*      stop
       return
      end
*
* ====================================================================
*
       subroutine calratio(ratio, tbp, ret)
*       ret_tt will contain the value "returned"
*
       real*8 ret, ratio
*
       ret = tbp * ratio
*
      RETURN
      END
*
* ====================================================================
*
      SUBROUTINE wavavg (hmax,dtrun,waveperiod,r4slopeF,FILEWAVE,noi)
C
c   This program calculates H when W and the slope
c   are provided
c
       INTEGER waveperiod
*
*      Determines whether to use a slope for calculating
*       h in range 4. 1=slope, 0=no slope
       INTEGER r4slopeF
       INTEGER noi    !number of data points calculated by wavavg.
      CHARACTER FILEWAVE*256
*
       INTEGER npts
*
      dimension h(3000)
      open(unit=6,file=FILEWAVE,status='unknown')
*       DO JK = 1,3000
*          H(JK) = 0.0
*        END DO
      int = 3
      ix = 0
*      npts = 1915
       npts = dtrun * 30
c    start of specified values
      iw1 = 25
      iw2 = 75
      iw3 = 200
      iw4 = npts
*       units: cms
        h1 = 0.0
        h3 = hmax * 0.2
*        if (h3 .LE. 0.05) h3 = 0.05
        h4 = 0.5*h3
         h5 = 0.0
c    end of specified values
c
c calculate range values
c
      irang1 = iw1
      irang2 = iw2 - iw1
      irang3 = iw3 - iw2
       irang4 = iw4 - iw3       !for decay
c
c
c   calculate slopes
c
```

```
      xm1 = (hmax - h1) / irang1
      xm2 = (h3 - hmax) / irang2
      xm3 = (h4 - h3) / irang3
      xm4 = (h5 - h4) / irang4 !for decay
c calculate h for range 1
      do 400 i = 1, irang1
        ix = ix + 1
        h(ix) = xm1 * i + h1
  400 continue
c calculate h for range 2
      do 500 i = 1, irang2
        ix = ix + 1
        h(ix) = xm2 * i + hmax
  500 continue
* calculate h for range 3
      do 600 i = 1, irang3
        ix = ix + 1
        h(ix) = xm3 * i + h3
  600 continue
*    calculate h for range 4     !for decay
      if (r4slopeF .eq. 0) then
          do 700 i = 1, irang4
            ix = ix + 1
            h(ix) = 0.0
  700   continue
       else
*        ... using slope
        do 701 i = 1, irang4
          ix = ix + 1
          h(ix) = xm4 * i + h4
  701   continue
       end if
*
        ipos = 0
      sum = 0
      noi = npts / int
      avg = 0.0
        write(6,315) avg, waveperiod
      do 705 j = 1, noi
        sum = 0.
        do 710 k = 1, int
          ipos = ipos + 1
          sum = sum + h(ipos)
  710   continue
        xint = int
        avg = sum / xint
        write(6,315) avg, waveperiod
  705 continue
  315 format(f8.3, 2x, i1)
*
      close (6)
*
      RETURN
      END
*
* ====================================================================
*
      SUBROUTINE ACKER (h0, va, vr, d50, VNU, RHO, conc)
*       conc is "returned"
*
      common/com/M0,g,pi,kk,rhow,visc,kkss,
     *           MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
*
      real*8 dt, vr
*
*
        if (d50 .EQ. 0.0) then
                write (*,*) 'D50 cannot be zero. Stopping.'
                stop
        end if
      cfc = .06 / (LOG10(12.0 * h0 / (3.0 * d50))) ** 2
```

```
      cfr = (2.87 + 1.58 * LOG10(1.0 / (3.0 * d50))) ** (-2.5)
      TAU = .5 * RHO * cfc * (va + (cfr / cfc) ** .5 * vr) ** 2
        if (RHO .EQ. 0.0) then
                write (*,*) 'RHO cannot be zero. Stopping.'
                stop
        end if
      ustar = (TAU/RHO)**.5
*
      GRAV = 9.805
      IF (ustar.EQ.0.0) GOTO 25
*
*     compute equivalent vbar based on keulegan equation
*
      vbar = ustar * 5.75 *LOG10(11.1 * h0  / (3 * d50))
*
*     ackers white equation
*
        if (VNU .EQ. 0.0) then
                write (*,*) 'VNU cannot be zero. Stopping.'
                stop
        end if
      dgr = d50 * ((GRAV * 1.65 / VNU ** 2) ** .3333)
      IF(dgr.LE.60.0) GO TO 22
*     COARSE PARAMETERS
      EN = 0.0
      A = .17
      EM = 1.78
      C = .025
      GOTO 23
22    CONTINUE
*     TRANSITION PARAMETERS
      EN = 1 - .56 * LOG10(dgr)
      A = .23 / (dgr)**.5 + .14
      EM = 6.83 / dgr + 1.67
      Cc = 2.79 * LOG10(dgr) - 0.98*(LOG10(dgr)) ** 2 - 3.46
      C = 10 ** Cc
23    CONTINUE
*     SEDIMENT MOBILITY
      FGR1 = (GRAV * d50 * 1.65)**.5
      FGR11 = ustar ** EN / FGR1
      FGR2 = 10 * h0 / d50
      FGR21 = 5.675 * LOG10(FGR2)
      FGR22 = (vbar / FGR21) ** (1 - EN)
      FGR = FGR11 * FGR22
*     TEST FOR ZERO TRANSPORT
      ZQS = FGR / A
      IF(ZQS.LE.1.0) GO TO 25
      GGR = C * (FGR / A - 1) ** EM
*     cnew = SED FLUX IN PARTS/PART by volume
      conc = (GGR * d50) / (h0 * (ustar / vbar) ** EN)
*     conc in Mg
      conc = conc * 10 ** 6
*
*     LIMIT CNEW TO 0.3
*      if(cnew.gt.0.3) cnew=0.3
      GOTO 26
25    conc=0
26    CONTINUE
*
      RETURN
      END
*
* ======================================================================
*
       subroutine vestuns (h0, filewave, vestout, time, conc)
*       conc is returned
C
c*************************************************************************
c*                                                                       *
c*           NON-EQUILIBRIUM SUSPENSION PROFILE MODELING                 *
c*                                                                       *
```

```
c**************************************************************************
c*      This program was originally written by Yigong Li and A.J. Mehta
c*
c*        Coastal & Oceanographic Engineering Department
c*                     University of Florida
c*                          Gainesville
c*                             1995
c*      Modified for variable waves and other needs of Upper Miss.
C*      whm Jan-Feb 98, May 98
c*
c*      To run the program you should give input filename--"FILEIN",
c*                                       output filename--"FILEOUT",
c*
c*            Constant filenames         wave data filename--"filewave.inp"
c*                                       Depth-avg conc --- "VESTOUT.out"
c*
C*      For input data file the description is as follows:
C*1,    descr:  TITLE FOR INPUT DATA
C*2,    h0:     WATER DEPTH (m)
C       m0:     MUD DEPTH (m)
C       NGRIDS: WATER LAYERS
C*3,    STT:    STARTING TIME OF PROGRAM  (min)
C       dtrun:  END TIME OF PROGRAM  (min)
C       dt:     CALCULATING TIME STEP (sec)
C       dtout:  OUTPUT TIME STEP  (min)
C       if dtout <= 0: the output time is given at the end of input file.
C       idout:  NUMBER OF SELECTED OUTPUT TIME
C       ddtout(I): SELECTED OUTPUT TIME (min)
C*4,    WSC0:   LIMIT CONCENTRATION FOR FREE SETTLING (kg/m**3)
C       ws0:    FREE FLOC. SETTLING VELOCITY (m/s)
C       wsa,wsb,wsm,wsn:       PARAMETERS FOR FLOCCULATION AND HINDERED SETTLING VELOCITY
C       where,  IF c.LT.WSC0
C                       Ws=WS0
C               IF C.GT.WSC0
C                       Ws=(WSA*c**WSN)/(C*C+WSB*WSB)**WSM
C*5,    dsk1,dsk2:      PARAMETER FOR STABILIZED DIFFUSION
C       where,  k = kn/(1+DSK1*ri)**DSK2 (Munk and Anderson Method)
C               ri:     GLOBAL RICHARDSON NUMBER
C       kn:     NEUTRAL MASS DIFFUSIVITY
C*6,    MODE:   SELECTION 1==CURRENT, 2==WAVE, 3==WAVE+WEAK CURRENT
C*7,    uu:     AVERAGE CURRENT VELOCITY (m/sec)
C       qn:     MANNING COEFFICIENT FOR CURRENT
C*8,    wheight:        WAVE HEIGHT (m)
C       wperiod:        WAVE PERIOD (sec)
C       wlength:        WAVE LENGTH ** CALCULATED IN PROGRAM
C       WDIFFK: MASS DIFFUSIVE COEFFICIENT FOR WAVE
C*9,    RHOSED: SEDIMENT GRANULAR DENSITY (kg/m**3)
C       rhom:   BULK DENSITY OF FLUID MUD OR BED(kg/m^3)
C       RHOW:   DENSITY OF WATER (kg/m^3)
C*10,   MCHOO: SELECTION FOR BOTTOM SEDIMENT:
C               1==FLUID MUD,2==BED
C       ric:    CRITICAL RICHARSON NUMBER FOR ENTRAINMENT
C       gg1,gg2,uu2:    MUD RHEOLOGICAL PARAMETERS
C               (THREE-ELEMENT MODEL - JIANG)
C       where,  IF GG11<0.0: TWO-ELEMENT VOIGT MODEL (G=GG2,U=UU2)
C       ent0,enn:       COEFFICIENTS FOR ENTRAINMENT RATE
C       where,  Erate = RHOM*ub*ent0*[(ric)^2*rig^(-1)-rig]^(enn)
C               ub:VELOCITY MAGNITUDE AT THE WATER BOTTOM
C       rig:    RICHARDSON NUMBER
C*11,   taudep:BED CRITICAL SHEAR STRESS FOR DEPOSITION
C       wfric:  WAVE FRICTION COEFFICIENT
C*12,   bdp1,BDP2,rhobar:       BED DENSITY PARAMETER
C       where,  DRY DENSITY -- rhob=rhobar*BDP1*((m0-dzbed)/m0)**BDP2
C*13,   PHIC,tcp1,tcp2:CRITICAL SHEAR STRESS PARAMETER
C       where,  critical stress--tauero=tcp1*(rhob/RHOSED-PHIC)**tcp2
C*14,   ERSMAX,ERP1,ERP2:       EROSION RATE COEFFICIENT PARAMETER
C       where,  erosion rate -- erort=ERSMAX*exp(-ERP1*tauero**ERP2)*(taubed-tauero)
C*15,   dps:    Deposition probability
C*16,   jn:     NUMBER OF INITIAL CONCENTRATION POINTS
C*17,   zi,ci:  INITIAL CONCENTRATION AND ELEVATION
C               (ZERO AT WATER-BED INTERFACE)
```

```
c
c
      parameter (nd=500)
      character filewave*256, vestout*256
      real*8 mmc,mmt,tt,ctt,dt
      real M0, kk, ks, kn, kkss
*
      dimension c0t(nd),dvel(nd),c0(nd),c(nd),z(nd),
     *          ws(nd),kn(nd),ks(nd)
      dimension zi(500),ci(500),ddtout(500)
      dimension p(nd),q(nd),r(nd)
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
      common/flow/UU,QN
      common/tide/qqn,umax,ptide
      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
      common/mr/gg1,gg2,uu2
      common/out/WSC0,WS0,WSA,WSB,WSM,WSN,DSK1,DSK2
      common/cal/suspmas,eromas
      common/bed/ dzb,TAUDEP,BDP1,BDP2,RHBBAR,PHIC,tcp1,tcp2,
     *             ERSMAX,ERP1,ERP2,tauero
        common/vest/ jn,zi,ci
c
C      integer, save :: resindex
      integer resindex
      save  resindex
      dimension time(2000), conc(2000)
*
      data g, pi, kk, visc                    !Program Constants
     *     / 9.81, 3.14159, 0.4, 1e-06 /
*
      open(10,file=filewave,status='unknown')
      open(11,file=vestout,status='unknown')
        write(11,*) 'VESTUNS output...'
        write(11,*) ' '
        write(11,*) 'Time Series of Suspended Sediment Concentration'
        write(11,*) 'due to vessel induced waves'
        write(11,*) ' '
      write(11,101)
101   Format(5x,'  TIME     WAVE HT WAVE PERIOD  CBAR',/
     1       5X,'  (sec)       (m)     (sec)     mg/l')
102   Format(5x,'  TIME     WAVE HT WAVE PERIOD  ABOVE BED   CONC',/
     1       5X,'  (sec)       (m)     (sec)        (m)      mg/l')
*
*
C ???
      idt = 0
      resindex = 0
C ???
      if(dtout.le.0.000001) then
          iii=idt+1
          write(*,*)iii,dtout
      else
          iii=int((dtrun-STT)/dtout)+1
      end if
c
      if(wheight.lt.1e-10) then
          wheight=1e-10
      end if
      dz=h0/float(NGRIDS)
      st=STT*60.0
      mmc=dtout*60.0
      mmt=dtrun*60.0
c*      initial data **
      do 1 i=1,NGRIDS
         z(i)=(float(i)-0.5)/float(NGRIDS)*h0
1     continue
c
      if(jn.eq.1) then
          do 2 i=1,NGRIDS
```

```
              c0(i)=ci(1)
2         continue
      end if
c
      if(jn.gt.1.and.jn.lt.NGRIDS) then
          do i=1,jn-1
              if(zi(i+1).le.zi(i)) then
      write(*,*)'Incorrect selection for initial concentration input.'
                 write(*,*)'The zi(i) is not in an increasing order.'
                 stop
              end if
          end do
          do 3 j=1,jn-1
            do 4 i=1,NGRIDS
              ss=(float(i)-0.5)*dz
              if(z(i).le.zi(1)) then
                c0(i)=ci(1)
              else if(z(i).ge.zi(jn)) then
                c0(i)=ci(jn)
              else
                if(z(i).gt.zi(j).and.z(i).lt.zi(j+1)) then
                  c0(i)=ci(j)+(ci(j+1)-ci(j))
     *                  *(z(i)-zi(j))/(zi(j+1)-zi(j))
                else if(z(i).eq.zi(j)) then
                  c0(i)=ci(j)
                else if(z(i).eq.zi(j+1)) then
                  c0(i)=ci(j+1)
                end if
              end if
4           continue
3         continue
      end if
c
      if(jn.eq.NGRIDS) then
          do 5 i=1,NGRIDS
              c0(i)=ci(i)
5         continue
      end if
c
c
      tt=st-dt
      ctt=-dt
      eromas=0.0
c
      do i=1,NGRIDS
          c(i)=c0(i)
      end do
c
c ****************************************************************
c
c     BEGIN TIME STEPPING
c
999   tt=tt+dt
      ctt=ctt+dt
c
      do 6 i=1,NGRIDS
        if(c0(i).lt.0.000001) then
           c0(i)=0.000001
        end if
6     continue
c
c
c      begin mods to vary wave.  whm jan 98
c
      read(10,*,err=909)wheight, wperiod
      if(MODE.eq.1) then
      if(MCHOO.eq.1) then
          write(*,*)'For entrainment of fluid mud under current,'
          write(*,*)'This case is not appropriate for practical use.'
          write(*,*)'Sorry!!!'
          stop
```

```
c           call hydroflow1(h0,dvel,kn)
c           ustar=UU
      else if(MCHOO.eq.2) then
          call hydroflow2(h0,dvel,kn,taubed)
      else
      end if
      end if
c
      if(MODE.eq.2) then
c      call prowmita(PWD1,PWD2,PDEN1,PDEN2,PWP,PAMP1,
c     *                     PMU1,Puu,PGG1,PGG2,ip,RRR,rr2)
      if(MCHOO.eq.1) then
          pmu1=visc*RHOW
          ip=10
          call prowmita(h0,M0,RHOW,RHOM,wperiod,wheight,
     *                     pmu1,uu2,gg1,gg2,ip,rrr,rr2)
          dudb=rrr
          call hydrowave1(h0,dvel,kn,rr2)
          ustar=ub
      else if(MCHOO.eq.2) then
          call hydrowave2(h0,dvel,kn,taubed)
      else
      end if
      end if
      if(MODE.eq.3) then
c      call prowmita(PWD1,PWD2,PDEN1,PDEN2,PWP,PAMP1,
c     *                     PMU1,Puu,PGG1,PGG2,ip,RRR,rr2)
      if(MCHOO.eq.1) then
          pmu1=visc*RHOW
          ip=10
          call prowmita(h0,M0,RHOW,RHOM,wperiod,wheight,
     *                     pmu1,uu2,gg1,gg2,ip,rrr,rr2)
          dudb=rrr
          call wavecurrent1(h0,dvel,kn,rr2)
          ustar=ub
      else if(MCHOO.eq.2) then
          rr2=0.0
          call wavecurrent2(h0,dvel,kn,taubed)
      end if
      end if
c      if(MODE.eq.4) then
c        call hydrotide(dvel,kn,tt,ustar)
c      end if
c
c      end mods whm
c
      do 12 i=2,NGRIDS
c*    settling velocity          *****
        ccc=abs((c0(i)-c0(i-1))/(c0(i)+c0(i-1)))
        if(ccc.gt.0.7) then
          cc=c0(i)
        else
        cc=(c0(i)+c0(i-1))/2.0
        end if
c        cc=(c0(i)+c0(i-1))/2.0
        if(cc.lt.WSC0) then
          ws(i)=(WSA*WSC0**WSN)/(WSC0*WSC0+WSB*WSB)**WSM
        else
          ws(i)=(WSA*cc**WSN)/(cc*cc+WSB*WSB)**WSM
        end if
c*    diffusion coefficient      *****
        if(MODE.eq.1) then
          vcc=0.01
        elseif (MODE.eq.2.) then
          vcc=0.0001
        elseif (MODE.eq.3) then
          vcc=0.01
        end if
        vcc=0.001
        if(cc.lt.vcc.or.dvel(i).le.0.0)then
          ri=0.0
```

```
        else
          if((-c0(i)+c0(i-1)).lt.0.0) then
            ri=-0.5/DSK1
          else
            ri=(g/cc)*((-c0(i)+c0(i-1))/dz)/(dvel(i))**2.0
          end if
        end if
        ks(i)=kn(i)/(1.0+DSK1*ri)**DSK2
12    continue
c*    Bottom boundary condition        **********
      if(c(1).lt.WSC0) then
        wsbb=(WSA*WSC0**WSN)/(WSC0*WSC0+WSB*WSB)**WSM
      else
        wsbb=(WSA*c0(1)**WSN)/(c0(1)*c0(1)+WSB*WSB)**WSM
      end if
      if(MCHOO.eq.1) then
          call bedflux1(wsbb,c0(1),ustar,fs,erconst)
      else
          call bedflux2(wsbb,c0(1),taubed,fs,erconst)
      end if
        eromas=eromas+fs*dt
c*    initial conditions               **********
      if(abs(tt).le.(st+1.0e-6)) then
        call result(h0,tt,c,ws,cbar)
           !store tow times and concentrations in arrays
           !  for use by calling routine
          resindex = resindex + 1
          time(resindex) = tt
          conc(resindex) = cbar
      end if
c*    implicit calculation             **********
      r(1)=-dt/(dz*dz)*ks(2)
      p(1)=1.0-r(1)
      c0t(1)=c0(1)+dt/dz*fs
      do 13 i=2,NGRIDS
        q(i-1)=r(i-1)-dt/dz*ws(i)
        r(i)=-dt/(dz*dz)*ks(i+1)
        p(i)=1.0-q(i-1)-r(i)
          c0t(i)=c0(i)
13    continue
      call stm(NGRIDS,c,c0t,p,q,r)
C*    output results                   *************
      if(mmc.gt.0.0001) then
           if(abs(ctt-mmc).le.0.0001 .and. tt.gt.0.00001) then
           call result(h0,tt,c,ws,cbar)
                 !store tow times and concentrations in arrays
                 !  for use by calling routine
          resindex = resindex + 1
              time(resindex) = tt
              conc(resindex) = cbar
            ctt=ctt-mmc
          end if
      else
          do i=1,idt
          if(abs(tt/60.0-ddtout(i)).lt.0.001) then
            call result(h0,tt,c,ws,cbar)
                 !store tow times and concentrations in arrays
                 !  for use by calling routine
          resindex = resindex + 1
              time(resindex) = tt
              conc(resindex) = cbar
          end if
          end do
      end if
      do 14 i=1,NGRIDS
        c0(i)=c(i)
14    continue
      if(tt.lt.mmt) then
      goto 999
      end if
       close(10)
```

```
       close(11)
*      stop
       return
909   end
C
C     Following is program VESTSUB.FOR
C
C
C     Subroutine for tri-diagonal matrix equations **
C
        subroutine stm(n,x,z,a,b,c)
        dimension x(n),z(n),a(n),b(n),c(n)
        dimension y(500),p(500),q(500),r(500)
        q(1)=a(1)
        do 1 i=1,n-1
          r(i)=b(i)
          p(i)=c(i)/q(i)
          q(i+1)=a(i+1)-p(i)*b(i)
1       continue
        y(1)=z(1)
        do 2 i=1,n-1
          y(i+1)=z(i+1)-y(i)*p(i)
2       continue
        x(n)=y(n)/q(n)
        do 3 i=n-1,1,-1
          x(i)=(y(i)-r(i)*x(i+1))/q(i)
3       continue
        end
C
C                   BEDFLUX Subroutine for Fluid Mud
C
        subroutine bedflux1(wsbb,c0bb,ustar,fs,erconst)
        real*8 dt
        real M0, kk, kkss, ustar
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
c
      if(rig.ge.ric) then
          fen=0.0
      else
          fen=RHOM*ustar*ent0*(ric*ric/rig-rig)**enn
      end if
      fse=-1.0*dps*wsbb*c0bb
      fs=fen+fse
      return
      end
C
C                   BEDFLUX Subroutine for Bed
C
        subroutine bedflux2(wsbb,c0bb,taubed,fs,erconst)
        real*8 dt
        real M0,kk,kkss
        common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
        common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
        common/bed/ dzb,TAUDEP,BDP1,BDP2,RHBBAR,PHIC,tcp1,tcp2,
     *              ERSMAX,ERP1,ERP2,tauero
c*        bed density           *****
        if(dzb.ge.M0) then
          write(*,*)'All the sediment at the bottom is eroded!'
      write(*,*)'Your bed depth is incorrect or time step is too large!'
          stop
        else if(dzb.lt.0.0) then
          rhob=RHBBAR*BDP1
        else
          rhob=RHBBAR*BDP1*((M0-dzb)/M0)**BDP2
        end if
c*        eosion shear strength     *****
        tauero=tcp1*(rhob/RHOSED-PHIC)**tcp2
        erconst=ERSMAX*exp(-ERP1*tauero**ERP2)
```

```
c*        Erosion or depsition rate         *****
        if(taubed.lt.TAUDEP) then
          fs=-1.0*wsbb*c0bb*(1.0-taubed/TAUDEP)
        else if(taubed.ge.tauero) then
          fs=ERSMAX*exp(-ERP1*tauero**ERP2)*(taubed-tauero)
        else
        fs=0.0
        end if
c*        Bed change          ** ?????  ******
        dzbt=dzb+dt*fs/rhob
        if(dzbt.le.M0) then
          dzb=dzbt
        else
          fs=(M0-dzb)*rhob/dt
          dzb=M0
        end if
        return
        end
C
C       HYDRODYNAMICS Subroutine--FLOW for fluid mud
C
c        subroutine hydroflow1(h0,dvel,kn)
c        real*8 dt
c        real M0, kk, kn, kkss
c        dimension dvel(1), kn(1)
c      common/com/M0,g,pi,kk,RHOW,visc,kkss,
c     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
c      common/flow/UU,QN
c      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
cc
c         kkss=3.15*100000.0*9.8**3.0*QN**6.0
c         y0=kkss/30.1
c         do 2 i=2,NGRIDS+1
c           z=(float(i-1))*dz
c           dvel(i)=UU/z/(alog(h0/y0)-1.0)
c           kn(i)=(kk*kk*UU)/(alog(h0/y0)-1.0)*z*(h0-z)/h0
c2        continue
c         rig=g*h0*(RHOM-RHOW)/RHOW/(UU*UU)
c         return
c         end
C
C       HYDRODYNAMICS Subroutine--FLOW for bed
C
        subroutine hydroflow2(h0,dvel,kn,taubed)
        real*8 dt
        real M0, kk, kn, kkss
        dimension dvel(1), kn(1)
        common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
        common/flow/UU,QN
        kkss=3.15*100000.0*9.8**3.0*QN**6.0
        y0=kkss/30.1
        ff=8.0*9.8*QN**2.0/h0**(1.0/3.0)
        taubed=RHOW*ff*UU**2.0/8.0
        do 2 i=2,NGRIDS+1
          z=(float(i-1))*dz
          dvel(i)=UU/z/(alog(h0/y0)-1.0)
          kn(i)=(kk*kk*UU)/(alog(h0/y0)-1.0)*z*(h0-z)/h0
2       continue
        return
        end
C
C       HYDRODYNAMICS Subroutine--WAVE for fluid mud
C
        subroutine hydrowave1(h0,dvel,kn,rd)
        real*8 dt
        real M0, kk, kn, kkss, rd
        dimension dvel(1), kn(1)
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
```

```
      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
c
        sig=2.0*3.14159265/wperiod
        qkk=(sig*sig)/9.8
1       qk=sig**2.0/(9.8*tanh(qkk*h0))
        if(abs(qk-qkk).gt.0.000001) then
          qkk=0.5*(qk+qkk)
          goto 1
        end if
        wnu=0.5*(qk+qkk)
        wlength=2.0*3.14159265/wnu
          wfr=2.0*pi/wperiod
        do 2 i=2,NGRIDS+1
          z=(float(i-1))*dz
          dvel(i)=wheight/2.0*wfr*wnu
     *            *sinh(wnu*z)/sinh(wnu*h0)
c         kn(i)=WDIFFK*wfr*wheight**2.0*(sinh(wnu*z))**2.0
c     *                                 /(2.0*(sinh(wnu*h0))**2.0)
        kn(i)=WDIFFK*wfr*wheight**2.0*sinh(wnu*h0/(h0+rd)*(z+rd))
     *        *cosh(wnu*z)/(2.0*(sinh(wnu*h0))**2.0)
2       continue
        rig=g*(visc*wperiod)**0.5*(RHOM-RHOW)/RHOW/(dudb*dudb)
        ub=sig/2.0*wheight/(sinh(wnu*h0))
        return
        end
C
C       HYDRODYNAMICS Subroutine--WAVE for bed
C
        subroutine hydrowave2(h0,dvel,kn,taubed)
        real*8 dt
        real M0, kk, kn, kkss
        dimension dvel(1), kn(1)
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *           MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
        sig=2.0*3.14159265/wperiod
        qkk=(sig*sig)/9.8
1       qk=sig**2.0/(9.8*tanh(qkk*h0))
        if(abs(qk-qkk).gt.0.000001) then
          qkk=0.5*(qk+qkk)
          goto 1
        end if
        wnu=0.5*(qk+qkk)
        wlength=2.0*3.14159265/wnu
          wfr=2.0*pi/wperiod
        do 2 i=2,NGRIDS+1
          z=(float(i-1))*dz
          dvel(i)=wheight/2.0*wfr*wnu*sinh(wnu*z)/sinh(wnu*h0)
          kn(i)=WDIFFK*wfr*wheight**2.0*(sinh(wnu*z))
     *          *cosh(wnu*z)/(2.0*(sinh(wnu*h0))**2.0)
2       continue
        taubed=RHOW*WFRIC/2.0*(wheight/2.0*wfr/sinh(wnu*h0))**2.0
        return
        end
C
C       HYDRODYNAMICS Subroutine--WAVE + WEAK CURRENT
C                                for fluid mud
C     WAVE -- entrainment and WAVE + WEAK CURRENT -- diffusion
        subroutine wavecurrent1(h0,dvel,kn,rd)
        real*8 dt
        real M0, kk, kn, kkss,knc,knw,rd
        dimension dvel(1), kn(1)
        dimension knc(500), knw(500), dvelc(500), dvelw(500)
      common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *           MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
      common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
      common/flow/UU,QN
      common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
c
        kkss=3.15*100000.0*9.8**3.0*QN**6.0
        y0=kkss/30.1
```

```
        do 1 i=2,NGRIDS+1
          z=(float(i-1))*dz
          aatt=alog(h0/y0)
          dvelc(i)=UU/z/(aatt-1.0)
          knc(i)=(kk*kk*UU)/(aatt-1.0)*z*(h0-z)/h0
1       continue
c
        sig=2.0*3.14159265/wperiod
        qkk=(sig*sig)/9.8
2       qk=sig**2.0/(9.8*tanh(qkk*h0))
        if(abs(qk-qkk).gt.0.000001) then
          qkk=0.5*(qk+qkk)
          goto 2
        end if
        wnu=0.5*(qk+qkk)
        wlength=2.0*3.14159265/wnu
          wfr=2.0*pi/wperiod
        do 3 i=2,NGRIDS+1
          z=(float(i-1))*dz
          dvelw(i)=wheight/2.0*wfr*wnu
     *            *sinh(wnu*h0/(h0+rd)*(z+rd))/sinh(wnu*h0)
c        kn(i)=WDIFFK*wfr*wheight**2.0*(sinh(wnu*z))**2.0
c     *                                 /(2.0*(sinh(wnu*h0))**2.0)
          knw(i)=WDIFFK*wfr*wheight**2.0*sinh(wnu*h0/(h0+rd)*(z+rd))
     *         *cosh(wnu*z)/(2.0*(sinh(wnu*h0))**2.0)
          dvel(i)=dvelw(i)+dvelc(i)
          kn(i)=knw(i)+knc(i)
3       continue
        rig=g*(visc*wperiod)**0.5*(RHOM-RHOW)/RHOW/(dudb*dudb)
        ub=sig/2.0*wheight/(sinh(wnu*h0))
        return
        end
C
C       HYDRODYNAMICS Subroutine--WAVE + WEAK CURRENT
C                                for bed
C
        subroutine wavecurrent2(h0,dvel,kn,taubed)
        real*8 dt
        real M0, kk, kn, kkss,knc,knw
        dimension dvel(1), kn(1)
        dimension knc(500), knw(500), dvelc(500), dvelw(500)
     common/com/M0,g,pi,kk,RHOW,visc,kkss,
     *           MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
     common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
     common/flow/UU,QN
c
          kkss=3.15*100000.0*9.8**3.0*QN**6.0
          y0=kkss/30.1
          do 1 i=2,NGRIDS+1
            z=(float(i-1))*dz
            aatt=alog(h0/y0)
            dvelc(i)=UU/z/(aatt-1.0)
            knc(i)=(kk*kk*UU)/(aatt-1.0)*z*(h0-z)/h0
1         continue
          ff=8.0*9.8*QN**2.0/h0**(1.0/3.0)
          taubedc=RHOW*ff*UU**2.0/8.0
c
          sig=2.0*3.14159265/wperiod
          qkk=(sig*sig)/9.8
2         qk=sig**2.0/(9.8*tanh(qkk*h0))
          if(abs(qk-qkk).gt.0.000001) then
            qkk=0.5*(qk+qkk)
            goto 2
          end if
          wnu=0.5*(qk+qkk)
          wlength=2.0*3.14159265/wnu
          wfr=2.0*pi/wperiod

          do 3 i=2,NGRIDS+1
            z=(float(i-1))*dz
            dvelw(i)=wheight/2.0*wfr*wnu
```

```
     *             *sinh(wnu*z)/sinh(wnu*h0)
c         kn(i)=WDIFFK*wfr*wheight**2.0*(sinh(wnu*z))**2.0
c     *                                 /(2.0*(sinh(wnu*h0))**2.0)
          knw(i)=WDIFFK*wfr*wheight**2.0*sinh(wnu*z)
     *         *cosh(wnu*z)/(2.0*(sinh(wnu*h0))**2.0)
          dvel(i)=dvelw(i)+dvelc(i)
          kn(i)=knw(i)+knc(i)
3         continue
          taubedw=RHOW*WFRIC/2.0*(wheight/2.0*wfr/sinh(wnu*h0))**2.0
          taubed=taubedw+taubedc
          return
          end
cC
cC         HYDRODYNAMICS Subroutine--tide
cC
c         subroutine hydrotide2(h0,dvel,kn,tt,taubed)
c         real*8 tt, dt
c         real M0, kk, ks, kn, kkss
c         dimension dvel(1), kn(1)
c       common/com/M0,g,pi,kk,RHOW,visc,kkss,
c      *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
c       common/tide/qqn,umax,ptide
c           u=abs(umax*sin((tt/3600.0)*(6.2832/ptide)))
c           kkss=3.15*100000.0*9.8**3.0*qqn**6.0
c           y0=kkss/30.1
c           ff=8.0*9.8*qqn**2.0/h0**(1.0/3.0)
c           taubed=RHOW*ff*u**2.0/8.0
c         do 2 i=2,NGRIDS+1
c           z=(float(i-1))*dz
c           dvel(i)=u/z/(alog(h0/y0)-1.0)
c           kn(i)=(kk*kk*u)/(alog(h0/y0)-1.0)*z*(h0-z)/h0
c2        continue
c         return
c         end
C
C              OUTPUT Subroutine
C
C      routine modified to remove file and graph outputs, used only to compute
C      depth-averaged concentration.  May 98
C
          subroutine result(h0,tt,c,ws,cbar)
*           cbar is returned
*
          real*8 tt,dt
          real M0, kk, kkss
          dimension c(1),ws(1)
       common/com/M0,g,pi,kk,RHOW,visc,kkss,
      *            MODE,NGRIDS,dz,dtrun,dtout,dt,STT,dps,MCHOO
       common/wave/wheight,wperiod,WDIFFK,wlength,ub,WFRIC
       common/flow/UU,QN
       common/tide/qqn,umax,ptide
       common/mud/RHOSED,RHOM,ric,rig,dudb,ent0,enn
       common/mr/gg1,gg2,uu2
       common/out/WSC0,WS0,WSA,WSB,WSM,WSN,DSK1,DSK2
       common/cal/suspmas,eromas
       common/bed/ dzb,TAUDEP,BDP1,BDP2,RHBBAR,PHIC,tcp1,tcp2,
      *              ERSMAX,ERP1,ERP2,tauero
*
c
c       if(abs(tt-STT*60.0).gt.0.01) goto 30
*        WRITE(*,*) 'Program IN PROGRESS - DO NOT TOUCH KEYBOARD'

30      suspmas = 0.0
        wssum=0.
*        WRITE(*,*) 'Program in Progress: Time=',tt/60.0,'(min)'
*       write(*,103, advance='no')
*103    format('.')
       do 11 i=NGRIDS,1,-1
          zz=(float(i)-0.5)*dz
          suspmas=suspmas+c(i)*dz
          wssum=wssum+ws(i)
```

```
11    continue
c     mod by whm jan 98
102   format(5x,4(g10.4),g12.6)
      cbar=suspmas/h0 * 1000
      wbar=wssum/NGRIDS
*
        write(11,101)tt,wheight,wperiod,cbar
101   format(5x,3(g10.4),g12.6)
c     mod end
9999  return
      end
C
C     VEST1>FOR
C
C
C     MAIN PROGRAM FOR WAVE-MUD INTERACTION
C
      subroutine prowmita(PWD1,PWD2,PDEN1,PDEN2,PWP,PAMP1,
     *                    PMU1,Puu,PGG1,PGG2,ip,RRR,rr2)
      integer ip
      real*4   PWD1,PWD2,PDEN1,PDEN2,PWP,PAMP1,
     *                    PMU1,Puu,PGG1,PGG2,RRR
c       REAL*8 V1(101),V2(101),V1S(101),V2S(101),
c     *         U1(101),U2(101),U1S(101),U2S(101),
c     *         P1(101),P2(101),P1S(101),P2S(101),
c     *         Z1(101),Z2(101)
c       REAL*8 VA1(101),VA2(101),VA1S(101),VA2S(101),
c     *         UA1(101),UA2(101),UA1S(101),UA2S(101),
c     *         PA1(101),PA2(101),PA1S(101),PA2S(101)
c      real*8 u1,u2

      REAL*8 DEN1,DEN2,WP,AMP1,SIGN,SIGN2,PI
      REAL*8 WD1,WD2,CRIT,WDY1,WDY2
      REAL*8 MU1,NU1
      REAL*8 GG1,GG2,UU2,AQ1,BQ0,BQ1

      COMPLEX*16 LSINH,LCOSH

      COMPLEX*16 A2,B2,C2,D2,E2,F2,G2,H2,AMP2,BAMP2
      COMPLEX*16 MU2,NU2
      COMPLEX*16 A1,B1,C1,D1,E1,F1,G1,H1,BAMP1,WN
      COMPLEX*16 UY1,UY2
      COMPLEX*16 II,LAM1,LAM2,BET1,BET2

      PI=DACOS(-1.0D0)
      II=(0.0,1.0)
      SIGN=.1D1
      SIGN2=.1D1
      CRIT=1.0D-15

      WD1=DBLE(PWD1)
      WD2=DBLE(PWD2)
      DEN1=DBLE(PDEN1)
      DEN2=DBLE(PDEN2)
      WP=DBLE(PWP)
      AMP1=DBLE(PAMP1)
      MU1=DBLE(PMU1)
      GG1=DBLE(PGG1)
      GG2=DBLE(PGG2)
      uu2=dble(puu)

C     OPEN(1,FILE='LI.IN',STATUS='UNKNOWN')
C     READ(1,*)WD1,WD2,DEN1,DEN2,WP,AMP1,MU1,uu2,GG1,GG2,ip
C     CLOSE(1)

C*****  MUD RHEOLOGICAL MODEL  ***************
      IF(PGG1.GE.0.0) THEN
      AQ1=uu2/(GG1+GG2)
      BQ0=(GG1*GG2)/(GG1+GG2)
      BQ1=(uu2*GG1)/(GG1+GG2)
```

```
      MU2=((6.2832D0/WP*(BQ1-BQ0*AQ1))
     *         +II*(BQ0+AQ1*BQ1*(6.2832D0/WP)**2.0))
     *        /(6.2832D0/WP*(1.0D0+(AQ1*6.2832D0/WP)**2.0))
      ELSE
      MU2=uu2+II*(GG2*dble(wp/(2.0*3.14159265)))
      END IF

      NU1=MU1/DEN1
      NU2=MU2/DEN2

C*****  CALL JIANG'S (1993) SUBROUTINE  *****************
      CALL CWAVE2ND(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,AMP1,SIGN,
     #               SIGN2,WN,A1,B1,C1,D1,E1,F1,G1,H1,BAMP1,
     #               A2,B2,C2,D2,E2,F2,G2,H2,BAMP2,AMP2,CRIT)


      LAM1=SIGN*(WN*WN-II*(0.62832D1/WP)/NU1)**0.5
      LAM2=SIGN*(WN*WN-II*(0.62832D1/WP)/NU2)**0.5
      BET1=SIGN2*(4.0D0*WN*WN-2.0D0*II*(0.62832D1/WP)/NU1)**0.5
      BET2=SIGN2*(4.0D0*WN*WN-2.0D0*II*(0.62832D1/WP)/NU2)**0.5

      ip1=ip

      WDY1=dsqrt(nu1*wp/2.0d0/pi/2.0d0)
      WDY2=WD2
      UY1=II*(A1*LCOSH(WN*WDY1)+B1*LSINH(WN*WDY1)
     *      +C1*LAM1/WN*CDEXP(LAM1*(WDY1-WD1))
     *      -D1*LAM1/WN*CDEXP(-LAM1*WDY1))
      UY2=II*(E1*LCOSH(WN*WDY2)+F1*LSINH(WN*WDY2)
     *      +G1*LAM2/WN*CDEXP(LAM2*(WDY2-WD2))
     *      -H1*LAM2/WN*CDEXP(-LAM2*WDY2))
      rr2=sngl(cdabs(bamp1))

      WDY1=dsqrt(nu1*2.0d0*wp/2.0d0/pi)
      UY1=II*(A1*LCOSH(WN*WDY1)+B1*LSINH(WN*WDY1)
     *      +C1*LAM1/WN*CDEXP(LAM1*(WDY1-WD1))
     *      -D1*LAM1/WN*CDEXP(-LAM1*WDY1))
      RRR=SNGL(CDABS(UY1-UY2))

      RETURN
      END

      FUNCTION LSINH(X)
      COMPLEX*16 LSINH
      COMPLEX*16 X
      LSINH=(CDEXP(X)-CDEXP(-X))/0.2D1
      RETURN
      END

      FUNCTION LCOSH(X)
      COMPLEX*16 LCOSH
      COMPLEX*16 X
      LCOSH=(CDEXP(X)+CDEXP(-X))/0.2D1
      RETURN
      END
C
C     VEST2.FOR
C
C**********************************************************************C
C     This subroutine is solving for the coefficients:                 C
C     A2,CB2,C2,D2,E2, F2,G2,H2,AMP2,B2 for second order solution      C
C     SH1=sinh(2*k*h1), CH1=cosh(2*k*h1)                               C
C     SH2=sinh(2*k*h2), CH2=cosh(2*k*h2)                               C
C     need subroutine  WNIST1.FOR                                      C
C     need subroutine  SINHCOSH.FOR                                    C
C**********************************************************************C
C
      SUBROUTINE CWAVE2ND(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,AMP1,SIGN,
     #                  SIGN2,WN,A1,B1,C1,D1,E1,F1,G1,H1,BAMP1,
     #                  A2,B2,C2,D2,E2,F2,G2,H2,BAMP2,AMP2,CRIT)
C.....................................................................
```

```
      COMPLEX*16 Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8
      COMPLEX*16 W1,W2,W3,W4,W5,W6,W7,W8,W9,W10
      COMPLEX*16 W11,W12,W13,W14,W15,W16,W17,W18,W19,W20
      COMPLEX*16 W21,W22,W23,W24,W25,W26,W27
      COMPLEX*16 A2,B2,C2,D2,E2,F2,G2,H2,AMP2,BAMP2
      COMPLEX*16 BETA1,BETA2,SH1,SH2,CH1,CH2,SINH1,SINH2,COSH1,COSH2
      COMPLEX*16 AN1,AN2,DA1,DA2
      COMPLEX*16 A1,B1,C1,D1,E1,F1,G1,H1,BAMP1,WN,DWN
      REAL*8 DEN1,DEN2,WP,SMA,AMP1,SIGN,SIGN2
      REAL*8 WD1,WD2,WD,DWD1,DWD2,CRIT
      REAL*8 MU1,NU1,RBAMP1,RBAMP2,RAMP2
      COMPLEX*16 MU2,NU2
C.....................................................
C       OPEN(UNIT=10,FILE='OUT.DAT',STATUS='NEW')
C.....................................................
      WD=WD1+WD2
      NU1=MU1/DEN1
      NU2=MU2/DEN2
      SMA=.628D1/WP
C       PRINT *,'WD1=',WD1
C       PRINT *,'WD2=',WD2
C       PRINT *,'DEN1=',DEN1
C       PRINT *,'DEN2=',DEN2
C       PRINT *,'NU1=',NU1
C       PRINT *,'NU2=',NU2
C       PRINT *,'WP=',WP
C       PRINT *,'AMP1=',AMP1
C       PRINT *,'SIGN=',SIGN
C.....................................................
      CALL WN1ST1(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,AMP1,SIGN,
     #         WN,A1,B1,C1,D1,E1,F1,G1,H1,BAMP1,CRIT)
C.....................................................
C       PRINT *,'SIGN2='
C       READ *,SIGN2
      DWN=(.2D1,0.)*WN
      BETA1=SIGN2*CDSQRT(DWN**2-(0.,.2D1)*SMA/NU1)
      BETA2=SIGN2*CDSQRT(DWN**2-(0.,.2D1)*SMA/NU2)
C.....................................................
      AN1=(0.,.1D1)*DEN1*SMA/WN-(.4D1,0.)*WN*MU1
      AN2=(0.,.1D1)*DEN2*SMA/WN-(.4D1,0.)*WN*MU2
C.....................................................
      DWD1=.2D1*WD1
      DWD2=.2D1*WD2
      CALL SINHCOSH(WN,DWD1,SH1,CH1)
      CALL SINHCOSH(WN,DWD2,SH2,CH2)
      CALL SINHCOSH(WN,WD1,SINH1,COSH1)
      CALL SINHCOSH(WN,WD2,SINH2,COSH2)
      DA1=SIGN*CDSQRT(WN**2-(0.,.1D1)*SMA/NU1)
      DA2=SIGN*CDSQRT(WN**2-(0.,.1D1)*SMA/NU2)
C.....................................................
      Y1=AMP1*(WN*A1*COSH1+WN*B1*SINH1+DA1*C1)
      Y2=AMP1*MU1*(((.2D1,0.)+DA1**2/WN**2)*(WN**2*(
     #     A1*SINH1+B1*COSH1)+DA1**2*C1)-(WN**2*(
     #     A1*SINH1+B1*COSH1)+DA1**4*C1/WN**2))
      Y3=-(.2D1,0.)*AMP1*((.2D1,0.)*WN**3*(A1*COSH1+
     #     B1*SINH1)+DA1*(WN**2+DA1**2)*C1)
      Y4=BAMP1*(WN*A1-DA1*D1)
      Y5=(.2D1,0.)*BAMP1*(WN**2*(E1*SINH2+F1*COSH2)+
     #     DA2**2*G1-WN**2*B1-DA2**2*D1)
      Y6=AMP1*(WN*(E1*COSH2-F1*SINH2)+DA2*G1-WN*A1+DA1*D1)
      Y7=AMP1*(-MU2*(WN**2+DA2**2)*(E1*SINH2+F1*COSH2)-
     #     MU2*(.2D1,0.)*DA2**2*G1+MU1*(WN**2+DA1**2)*B1
     #     +(.2D1,0.)*MU1*DA1**2*D1)
      Y8=AMP1*(MU2*((.4D1,0.)*WN**3*(E1*COSH2+F1*SINH2)+
     #     (.2D1,0.)*DA2*(DA2**2+WN**2)*G1)-MU1*((.4D1,0.)*
     #     WN**3*A1-(.2D1,0.)*DA1*(DA1**2+WN**2)*D1))
C.....................................................
      W1=BETA2*SH1/DWN-CH1
      W2=BETA2*CH2-DWN*SH2
C.....................................................
      W3=AN1*CH1-(0.,.1D1)*DEN1*.98D1*SH1/SMA/(.2D1,0.)
```

```
      W4=AN1*SH1-(0.,.1D1)*DEN1*.98D1*CH1/SMA/(.2D1,0.)
C.....................................................................
      W5=DWN*((.2D1,0.)*MU1*BETA1+(0.,.1D1)*
     #     DEN1*.98D1/SMA/(.2D1,0.))
      W6=(W2*W3+DWN*W1*W4)/W5
      W7=BETA2*W3/W5
      W8=(BETA1*W3-DWN*W4)/W5
      W9=(W3*Y5+DWN*W4*Y6-DWN*(Y2+
     #     (0.,.1D1)*DEN1*.98D1*Y1/SMA/(.2D1,0.)))/W5
C.....................................................................
      W10=(.8D1,0.)*WN**2*(W2*SH1/DWN+W1*CH1)+
     #      W6*(BETA1**2+DWN**2)
      W11=(.8D1,0.)*WN**2*BETA2*SH1/DWN+
     #      W7*(BETA1**2+DWN**2)
      W12=(.8D1,0.)*WN**2*(BETA1*SH1/DWN-CH1)+
     #      W8*(BETA1**2+DWN**2)
      W13=Y3-(.8D1,0.)*WN**2*(SH1*Y5/DWN+CH1*Y6)
     #      -W9*(BETA1**2+DWN**2)
C.....................................................................
      W14=W2*AN1/DWN-BETA2*AN2*CH2/DWN+
     #      AN2*SH2+(0.,.1D1)*(DEN2-DEN1)*.98D1*W1/SMA/(.2D1,0.)
      W15=BETA2*AN1/DWN+(.2D1,0.)*MU2*BETA2*AN2
      W16=BETA1*AN1/DWN+(.2D1,0.)*MU1*BETA1
      W17=Y7-(0.,.1D1)*(DEN2-DEN1)*.98D1*(Y6+Y4)/SMA/(.2D1,0.)
     #      -AN1*Y5/DWN
C.....................................................................
      W18=(.8D1,0.)*MU1*WN**2*W1-(.8D1,0.)*MU2*WN**2*(BETA2
     #      *SH2/DWN-CH2)
      W19=-(.8D1,0.)*MU2*WN**2*(BETA2**2+DWN**2)
      W20=MU1*(BETA1**2-DWN**2)
      W21=Y8-(.8D1,0.)*MU1*Y6*WN**2
C.........................
      W22=W11*W14-W15*W10
      W23=W12*W14-W16*W10
      W24=W13*W14-W17*W10
      W25=W11*W18-W19*W10
      W26=W12*W18-W20*W10
      W27=W13*W18-W21*W10
C.......................................
      D2=(W24*W25-W27*W22)/(W23*W25-W26*W22)
      G2=(W24-W23*D2)/W22
      H2=(W13-W11*G2-W12*D2)/W10
C.......................................
      F2=-H2
      E2=BETA2*H2/DWN
      BAMP2=(0.,.1D1)*(W1*H2+Y6+Y4)/SMA/(.2D1,0.)
      B2=W1*H2-D2+Y6
      A2=(H2*W2+BETA2*G2+BETA1*D2+Y5)/DWN
      C2=W6*H2+W7*G2+W8*D2+W9
      AMP2=(0.,.1D1)*(A2*SH1+B2*CH1+C2+Y1)/SMA/(.2D1,0.)
C.......................................
      RBAMP1=DREAL(BAMP1)
      RBAMP2=DREAL(BAMP2)
      RAMP2=DREAL(AMP2)
C       WRITE(10,140) WD1,WD2,WP,DEN1,DEN2,MU1,MU2
C       WRITE(10,150) WN
C       WRITE(10,200) AMP1,RBAMP1
C       WRITE(10,300) RAMP2,RBAMP2
140   FORMAT(2X,'WD1=',F10.5,2X,'WD2=',F10.5,2X,'WP=',F10.5,2X,
     #        'DEN1=',F10.5,2X,'DEN2=',F10.5,2X,'MU1=',F10.5,2X,
     #        'MU2=',F10.5)
150   FORMAT(2X,'WAVE NUMBER=',F10.6,5X,F10.6)
200   FORMAT(2X,'AMP1=',F10.5,5X,'BAMP1=',F10.6)
300   FORMAT(2X,'AMP2=',F10.6,5X,'BAMP2=',F10.6)
C.......................................
      RETURN
      END
C
C*********************************************************************C
C       WN1ST1.FOR                                                    C
C       The solutions to second-order wave over soft mud are          C
```

```
C       composed of two parts: first order part and second            C
C       order part. This program is for solving (complex) wave number   C
C        using the secant method, and all coefficients                C
C       A1,CB1,C1,D1,E1,F1,G1,H1,B1 for the first part.               C
C       Units are in Metric system, Double precision                  C
C       Need subroutines AIRY,FUNC1ST1,SINHCOSH.                       C
C*********************************************************************C
      SUBROUTINE WN1ST1(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,AMP1,SIGN,
     #                  WN,A1,B1,C1,D1,E1,F1,G1,H1,SAMP1,CRIT)
C.............................................
      COMPLEX*16 WFUN1,WFUNX,WN1,WN2,WN,DX
      COMPLEX*16 X,FOLD,FNEW,TED,TEG,TEH
      COMPLEX*16 A1,B1,C1,D1,E1,F1,G1,H1,SAMP1
      COMPLEX*16 DA1,DA2,SINH1,COSH1,SINH2,COSH2
      REAL*8 WP,SMA,WD1,WD2,DEN1,DEN2,WDTEP
      REAL*8 WD,MU1,NU1,AMP1,SIGN,CRIT
      COMPLEX*16 MU2,NU2
C.............................................
      WD=WD1+WD2          ! TOTAL DEPTH
      NU1=MU1/DEN1
      NU2=MU2/DEN2
      SMA=.628D1/WP
C       CRIT=0.000000001
C       IF(WP.GT.1.9) THEN
C       CRIT=0.00000001
C       PRINT *,'Need new CRIT='
C       READ *,CRIT
C       ENDIF
C.............................................................
      WDTEP=WD1+WD1            ! FOR ITERATION PURPOSE
      CALL AIRY(WP,WD1,WN1)            !WN1>WN2
      CALL AIRY(WP,WDTEP,WN2)
C       CALL AIRY(WP,WD,WN2)
C       PRINT *,'WN1=',WN1
C       PRINT *,'WN2=',WN2
C............................
      DX=WN2-WN1
      X=WN2
      CALL FUNC1ST1(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,SIGN,
     #              AMP1,WN1,WFUN1,TED,TEG,TEH,H1,
     #              DA1,DA2,SINH1,COSH1,SINH2,COSH2)
      FOLD=WFUN1
      N=0
100   CALL FUNC1ST1(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,SIGN,
     #              AMP1,X,WFUNX,TED,TEG,TEH,H1,
     #              DA1,DA2,SINH1,COSH1,SINH2,COSH2)
      N=N+1
      IF(N.GT.100) GO TO 200
      FNEW=WFUNX
      DX=-FNEW*DX/(FNEW-FOLD)
      X=X+DX
C       PRINT *,'N=',N,',   ','WAVE NUMBER=',X
      IF(CDABS(DX).GT.CRIT) THEN
       FOLD=FNEW
       GO TO 100
      ENDIF
C.................................
200   WN=X
C.................................
      PRINT *,'N=',N
      D1=TED*H1
      G1=-TEG*H1
      F1=-H1
      E1=DA2*H1/WN
      SAMP1=(0.,.1D1)*(E1*SINH2+F1*COSH2+G1)/SMA
C             FIRST ORDER INTERFACIAL AMPLITUDE
      B1=E1*SINH2+F1*COSH2+G1-D1
      A1=(E1*WN*COSH2+F1*WN*SINH2+G1*DA2+D1*DA1)/WN
      C1=-(.2D1,0.)*NU1*AMP1*WN**2
C..................
      RETURN
```

```
      END
C
C
C*************************************************************C
C     The following approximate solution to the dispersion     C
C     relation was proposed by Hunt (1979)                     C
C     Units are in Metric System                               C
C*************************************************************C
      SUBROUTINE AIRY(PERIOD,DEPTH,WN)
      COMPLEX*16 WN
      REAL*8 GRAV,SUM,PERIOD,DEPTH,Y,CGM,DUM,D(10),KH
      GRAV=.98D1
      CGM=.62832D1/PERIOD
      Y=CGM**2*DEPTH/GRAV
      D(1)=0.066666666D1
      D(2)=0.035555555D1
      D(3)=0.01608465608D1
      D(4)=0.00632098765D1
      D(5)=0.00217540484D1
      D(6)=0.00065407983D1
      SUM=.1D1
C.................................................
      DO J=1,6
      DUM=Y**J
      SUM=SUM+D(J)*DUM
      END DO
C.................................................
C      KH=SQRT(Y**2+Y/SUM)
      KH=DSQRT(Y**2+Y/SUM)
      WN=KH/DEPTH
C.................................................
      RETURN
      END
C
C*****************************************************************C
C*      This subroutine is for preparing the function for the      *C
C*      secant method for solving the first order problem           *C
C*      Units are in Metric system, Double precision               *C
C*      Input information:                                         *C
C*      WD1=WATER DEPTH IN UPPER LAYER                              *C
C*      WD2=WATER DEPTH IN LOWER LAYER                              *C
C*      DEN1=DENSITY IN UPPER LAYER                                 *C
C*      DEN2=DEMSITY IN LOWER LAYER                                 *C
C*      MU1=DYNAMIC VISCOSITY IN UPPER LAYER                        *C
C*      MU2=DYNAMIC VISCOSITY IN LOWER LAYER                        *C
C*      WP=INCOMING WAVE PERIOD                                     *C
C*      AMPL1=FIRST ORDER WAVE AMPLITUDE                            *C
C*      WN=WAVE NUMBER (COMPLEX)                                    *C
C*      RETURNED INFORMATION:                                       *C
C*      FUNCTION WHICH WILL BE USED FOR SECANT THE METHOD           *C
C*****************************************************************C
      SUBROUTINE FUNC1ST1(WD1,WD2,DEN1,DEN2,MU1,MU2,WP,SIGN,
     #                    AMP1,WN,WFUN,TED,TEG,TEH,H1,
     #                    DA1,DA2,SINH1,COSH1,SINH2,COSH2)
C.................................................................
      COMPLEX*16 DA1,DA2,S2,Q2,TED,TEG,TEH,WN
      COMPLEX*16 AM1,AM2,SINH1,SINH2,COSH1,COSH2
      COMPLEX*16 WFUN,H1
      REAL*8 WP,SMA,WD1,WD2,DEN1,DEN2,NU1
      REAL*8 MU1,AMP1,SIGN
      COMPLEX*16 MU2,NU2
C.................................................................
      NU1=MU1/DEN1
      NU2=MU2/DEN2
      SMA=.628D1/WP
C.....................................
      CALL SINHCOSH(WN,WD1,SINH1,COSH1)
      CALL SINHCOSH(WN,WD2,SINH2,COSH2)
C.....................................................
      DA1=SIGN*CDSQRT(WN**2-(0.,0.1D1)*SMA/NU1)  ! `+' OR `-'
      DA2=SIGN*CDSQRT(WN**2-(0.,0.1D1)*SMA/NU2)
```

```
      S2=SINH2-COSH2*DA2/WN
      Q2=COSH2-SINH2*DA2/WN
      AM1=(0.,.1D1)*SMA*DEN1/WN-.2D1*WN*MU1
      AM2=(0.,.1D1)*SMA*DEN2/WN-.2D1*WN*MU2
      TEG=(.2D1*DA1*(MU2*Q2-MU1*Q2)+
     #         (AM2-AM1)*S2-(0.,.1D1)*(DEN2-DEN1)*.98D1*Q2
     #       /SMA)/(.2D1*MU1*DA1+DA2*AM1/WN
     #   +.2D1*MU2*DA2+(0.,.1D1)*(DEN2-DEN1)*.98D1/SMA
     #   -DA1*MU2*(DA2**2+WN**2)/WN/WN)
      TED=(.2D1*WN**2*(MU2*Q2-MU1*Q2)-TEG*
     #        (.2D1*MU1*WN**2-MU2*(DA2**2+
     #         WN**2)))/SMA/DEN1/(0.,.1D1)
      TEH=(DA1*COSH1/WN-SINH1)*TED-
     #        TEG*(DA2*COSH1/WN+SINH1)-
     #        S2*COSH1-Q2*SINH1
      H1=(DEN1*.98D1*AMP1-.4D1*DEN1*(NU1**2)*AMP1*
     #      (WN**2)*DA1)/AM1/TEH
C.........................................
      WFUN=(DA1*SINH1/WN-COSH1)*TED*H1-
     #        (DA2*SINH1/WN+COSH1)*TEG*H1-
     #        (S2*SINH1+Q2*COSH1)*H1+
     #        AM1*WN*AMP1/DEN1
C.............................................
      RETURN
      END

C*****************************************************C
C     This subroutine is for calculation of functions    C
C       sinh(x) and cosh(x) with double precision        C
C       complex variables                                C
C*****************************************************C
      SUBROUTINE SINHCOSH(WN,WD,SINHX,COSHX)
      COMPLEX*16 XKH,WN,TEMP1,TEMP2,SINHX,COSHX
      REAL*8 WD
C.....................................
      XKH=WN*WD
      TEMP1=CDEXP(XKH)
      TEMP2=CDEXP(-XKH)
      SINHX=(TEMP1-TEMP2)/.2D1
      COSHX=(TEMP1+TEMP2)/.2D1
C       PRINT *,'KH=',XKH,',   ','COSH(KH)=',COSHX
C.....................................
      RETURN
      END
C.....................................
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
December 2004	Interim report	

4. TITLE AND SUBTITLE

User's Manual for NAVSED

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Clay LaHatte and Stephen T. Maynord

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Coastal and Hydraulics Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

See reverse

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

ENV Report 46

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The NAVSED program generates sediment concentration time histories due to passage of shallow draft navigation. This user's manual describes how to run the program interactively or how to set up batch files to run the program. The various input and output file formats are described and example files are presented.

15. Subject Terms

Computer program | Navigation | Sediment | User's manual

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED		60	

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESSES (Concluded).

U.S. Army Engineer District, Rock Island, Clock Tower Building, P.O. Box 2004, Rock Island, IL 61204-2004

U.S. Army Engineer District, St. Louis, 1222 Spruce Street, St. Louis, MO 63103-2833

U.S. Army Engineer District, St. Paul, Army Corps of Engineers Centre, 190 5th Street East, St. Paul, MN 55101-1638