No. 24, 1998

SMHI

Reports Oceanography

PROBE
Program for Boundary Layers in the Environment
System description and Manual

Urban Svensson
Computer-aided Fluid Engineering AB

VP

RO
No. 24, 1998

PROBE
Program for Boundary Layers in the Environment
System description and Manual

Urban Svensson
Computer-aided Fluid Engineering AB

Report Summary / Rapportsammanfattning

Issuing Agency/Utgivare Swedish Meteorological and Hydrological Institute SE-601 76 NORRKÖPING Sweden	Report number/Publikation RO No. 24 Report date/Utgivningsdatum May 1998

Author (s)/Författare

Urban Svensson

Title (and Subtitle)

PROBE

Program for Boundary Layers in Environmental system description and Manual.

Abstract/Sammandrag

The manual is intended to provide users of PROBE computer code with necessary background information and assistance for successful use.

PROBE (<u>PRO</u>gram for <u>B</u>oundary Layers in the <u>E</u>nvironment) can be classified as an "equation solver for one-dimensional transient, or two dimensional steady, boundary layers". Typical examples of such boundary layers are the Ekman layer and the developing channel flow. A major difficulty in these kinds of flow is to characterise the turbulent mixing in mathematical terms. PROBE embodies a two-equation turbulence model, the $k-\varepsilon$ model, which calculates mixing coefficients. Together with two momentum equations the turbulence model forms the basis for the hydrodynamical part of the mathematical model. In the basic version six additional variables are provided for: heat energy, salinity, and four concentrations. The number of concentrations can, of course, easily be increased when needed.

Key words/sök-, nyckelord

Environmental flows, equation solver, boundary layers, CFD.

Supplementary notes/Tillägg	Number of pages/Antal sidor 42 + Appendices	Language/Språk English

ISSN and title/ISSN och titel

0283-1112 SMHI Reports Oceanography

Report available from/Rapporten kan köpas från:

SMHI
SE-601 76 NORRKÖPING
Sweden

TABLE OF CONTENTS

		Page
1.	**INTRODUCTION**	**1**
	1.1 Purpose of the manual	1
	1.2 The general features of PROBE	1
	1.3 What PROBE can do	1
	1.4 The history and future of PROBE	4
	1.5 Outline of the manual	4
2.	**BRIEF DESCRIPTION OF BASIC EQUATIONS AND TECHNIQUES**	**4**
	2.1 The general differential equation	4
	2.2 Numerical methods employed	5
3.	**DESCRIPTION OF THE CODE**	**6**
	3.1 Flow diagrams	6
	3.2 General section subroutines	9
	3.3 User section subroutine	11
4.	**HOW TO USE PROBE**	**11**
	4.1 Analysis of the problem considered	11
	4.2 Modification of default data	11
	4.3 The CASE subroutine	21
	4.4 Test calculation	22
5.	**ADVICE ON EFFECTIVE USE**	**22**
	5.1 Grid independence in space and time	22
	5.2 Use of integral checks	22
	5.3 Verification studies	22
	5.4 Causes of diverging solutions	23
	5.5 Some advice on mounting PROBE	23
6.	**CONCLUDING REMARKS**	24
7.	NOMENCLATURE	24
8.	**REFERENCES**	**29**
	8.1 Literature references for the manual	29
	8.2 Presently available CASE-reports	30
	8.3 Papers and reports based on the use of PROBE	33
9.	**ACKNOWLEDGEMENTS**	**38**
10.	**APPENDIX A Mathematical formulation**	
	APPENDIX B Finite difference equations for the one-dimensional transient option.	
	APPENDIX C The finite difference equations for the two-dimensional steady option.	
	APPENDIX D Listing of the code	

1. INTRODUCTION

1.1 Purpose of the manual

This manual is intended to provide users of the PROBE computer code with necessary background information and assistance for successful use. The user in mind is supposed to have some knowledge in the field of computational fluid dynamics, i.e. fluid dynamics, numerical analysis and computer programming. However, the structure of PROBE allows the user to develop his/her understanding of the code and computational fluid dynamics in a gradual manner. PROBE, with its manual, is thus suitable as a teaching aid.

The manual does not contain descriptions of applications of PROBE. These are given in separate CASE-reports, each of which provides a full description of how to apply the code to a specific problem. The CASE-reports are thus essential supplementary material when one gets familiar with PROBE.

After studying the manual and running a few applications from CASE-reports it is believed that the user will be in a position to carry out new applications. The reader without prior experience of computational fluid dynamics should, however, be aware of the fact that numerical prediction of fluid flow phenomena rarely becomes simple or standard. This is due to non-linearities in the basic equations and the boundary conditions. Written material can therefore only assist the user in getting a good result; the intelligence and insight of the user have to be relied upon in most situations.

1.2 The general features of PROBE

PROBE (PROgram for Boundary Layers in the Environment) can be classified as an "equation solver for one-dimensional transient, or two dimensional steady, boundary layers". Typical examples of such boundary layers are the Ekman layer and the developing channel flow. A major difficulty in these kinds of flow is to characterise the turbulent mixing in mathematical terms. PROBE embodies a two-equation turbulence model, the $k-\varepsilon$ model, which calculates mixing coefficients. Together with two momentum equations the turbulence model forms the basis for the hydrodynamical part of the mathematical model. In the basic version six additional variables are provided for: heat energy, salinity, and four concentrations. The number of concentrations can, of course, easily be increased when needed.

PROBE has been structured in a way which is believed to facilitate easy and safe use. The user will only be concerned with one subroutine, called CASE, while the rest of the program should not be subject to modifications. Many applications will only require the insertion of about 15 FORTRAN-statements in CASE.
PROBE is written in standard FORTRAN-77 and requires very little memory. This makes the code suitable for both PC:s and main frame computers. All units are in the SI-system.

1.3 What PROBE can do

As already mentioned, it is boundary layers that is the class of flows considered. This may seem to be a rather narrowly restricted class of flows. However, the number of applications already carried out gives an opposite impression. For environmental flows and idealised one-dimensional analysis can often provide good insight and understanding of a new problem. The name PROBE itself also indicates that a one-dimensional analysis can be a preliminary sensor in a more complex (three-dimensional) analysis. To give a first impression of what PROBE can do, a few examples will be discussed briefly.

A. The entrainment experiment by Kantha et al. (1977)

This laboratory experiment deals with the rate of deepening of an initially two-layered fluid suddenly exposed to shear on the surface, see Figure 1.1a. A race-track type of flume ensures that the experiment is one-dimensional. Predicted and measured deepening is shown in Figure 1.1b.

B. Autumn cooling of the ocean

The ocean Ekman layer, stratified with respect to both temperature and salinity, has been analysed with PROBE (see Omstedt et al., 1983). Unexpected phenomena, like local temperature maxima, are found both in field measurements and predictions, see Figure 1.1b.

C. The adiabatic atmospheric boundary layer

An example of a two-dimensional steady situation is given in Figure 1.1c, where the flow over an island is shown (from Nordblom, 1997).

Hopefully, these examples will give the reader an impression of the kind of flows that PROBE can be applied to. Complete instructions on how to modify the code for these and other applications are provided in separate CASE-reports. These reports contain a description of the problem, the mathematical formulation, a few results of predictions, and a listing of the subroutine CASE. Presently available CASE-reports are listed in Chapter 8.2.

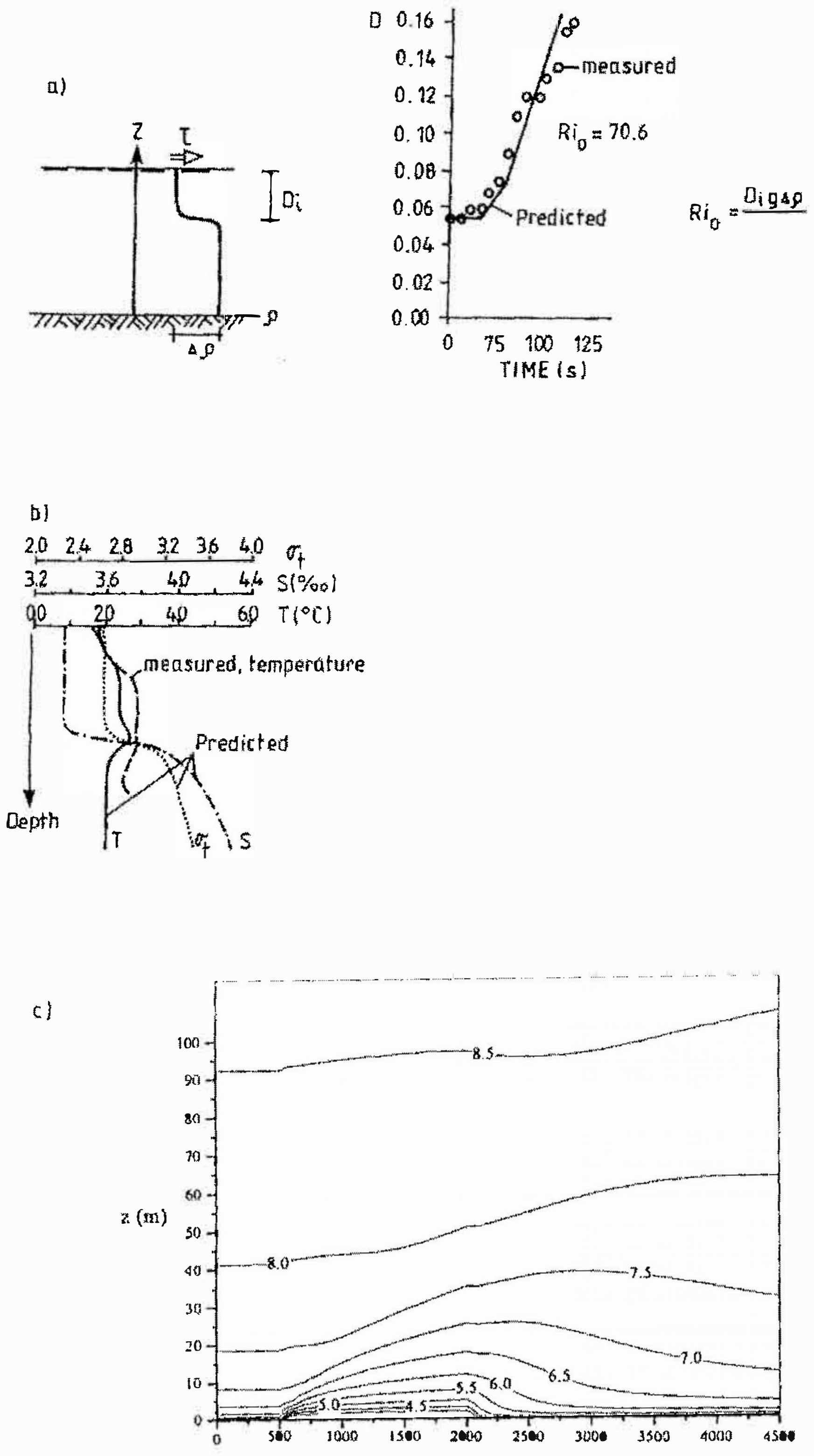


Figure 1.1. *a) The entrainment experiment.*
b) Autumn cooling of the ocean.
c) The atmospheric boundary layer. Horizontal velocity distribution (m/s) in air flowing from left to right across a flat island extending from $x = 500$ m to $x = 2\,000$ m.

1.4 The history and future of PROBE

The first version of PROBE, even though it had no name at the time, was presented in Svensson (1978). That version was designed for studies of the seasonal thermocline, but other applications could also be carried out. In fact, it was the range of possible applications that motivated the construction of the present more general version of PROBE.

The version was first released in 1984 and has now been successfully applied to a wide range of different problems. The 1986-version was further developed in several respects, of which the more important ones are: A series of interacting runs can be performed, a moving free surface is introduced, and more flexibility is provided in terms of number of equations, cells, etc. The present 1997-version extends the capabilities of PROBE further by including two-dimensional steady boundary layers into the class of flows that can be analysed with PROBE.

The direction of future developments is closely related to the kind of applications that will be dominating. Among several possibilities one may mention:

- Dispersed and layered two-phase flows. This is a difficult task, which will only be undertaken, if the development work can be supported and motivated by a major project.
- Re-write the code using object-oriented techniques. The present version does not employ modern concepts in respect of code-construction and coding. When PROBE is more closely integrated with other code-systems it may prove necessary to re-write the code.

1.5 Outline of the manual

A brief description of the basic differential equation and its finite difference counterpart are given in the following chapter. Chapter 3 outlines the general features of the code. The instructions on the use of PROBE are given in Chapter 4. Advice on effective use can be found in Chapter 5, and finally in Chapter 6 some concluding remarks are given. Details of the differential equations employed and the finite difference equations are given in Appendix A, B and C, respectively. A listing of the code is the content of Appendix D.

2. BRIEF DESCRIPTION OF BASIC EQUATIONS AND TECHNIQUES

2.1 The general differential equation

All differential equations can formally be written as:

$$\frac{\partial \phi}{\partial t} + \frac{\partial}{\partial x_i} u_i \phi = \frac{\partial}{\partial z}\left(\Gamma_\phi \frac{\partial \phi}{\partial z}\right) + S_\phi, \qquad (2.1)$$

Change in time Advection Diffusion Source/Sink where ϕ is the dependent variable, t time, z vertical coordinate, x horizontal coordinate, u horizontal velocity, Γ_ϕ exchange coefficient, and S_ϕ source and sink terms. For one-dimensional cases the advection term is not active and for two-dimensional steady cases the transient term is absent. The equation is formulated in a cartesian coordinate system shown in Figure 2.1a. When ϕ, as an example, is heat energy, the source term will contain terms describing the penetration of short wave radiation, while for momentum the pressure gradient is a typical source term. Advection along the vertical space coordinate is included to account for vertical transport in a reservoir due to in- and outflows. However, as it is not yet fully developed for general application, the term is, formally, included in the source term. A complete discussion of all differential equations is given in Appendix A.

Boundary conditions may be specified in two different ways; either the value or the flux of the variable in question is given. If a wind stress on a water surface is prescribed, it is thus the flux alternative that is chosen.

2.2 Numerical methods employed

The general differential equation can be integrated over a specified volume, a grid cell, with the following result:

$$\phi_i\left(D_i + S_i'\right) = \phi_{i+1}A_i + \phi_{i-1}B_i + S_i \qquad (2.2)$$

where D_i, A_i, and B_i are coefficients and S_i and S_i' source terms. The grid arrangement is shown in Figure 2.1b. It is seen that variables are stored in N locations. As two of these are on the boundaries, it follows that the number of cells is $N-2$. Equation (2.2) shows that the value of grid cell i, ϕ_i, is related to the values in the neighbouring cells ϕ_{i+1} and ϕ_{i-1}. The strength of the connection is given by the coefficients A_i and B_i, which, on closer inspection, are found to represent transport effects. The detailed derivation of the finite difference equations is given in Appendixes B and C.

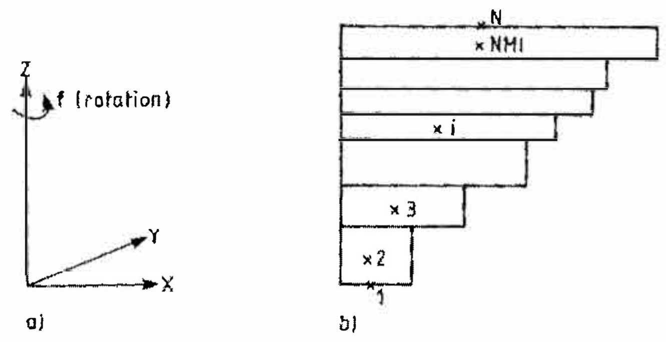


Figure 2.1. *a) Coordinate system.*
b) Grid cell arrangement.

3. DESCRIPTION OF THE CODE

In this chapter the structure of PROBE and the purpose of the different subroutines will be explained. The reader is advised, while reading the following sections, to make a brief inspection of the listing of PROBE, supplied in Appendix D.

3.1 Flow diagrams

A flow diagram is given in Figure 3.1. As seen, the code is divided into two parts; the user section and the general section. In terms of FORTRAN lines the user subroutine CASE will only amount to a few percent of the total code, which amounts to about 1500 lines including all comment statements. The flow diagram shows four links between the general section and the user section. It should be noted that three of these are within the DO-loop in MAIN, which is responsible for the advancement in time (or space in a 2D steady calculation). This DO-loop runs from chapter 4 to 9, as indicated. This arrangement makes it possible to interact with the calculations in a simple way. An example of when this is needed is given by the boundary condition at a water surface for dissolved oxygen. If it is assumed that the oxygen content is at its saturation value, one has to prescribe this value as a function of temperature, which is a calculated variable. A continues interaction is thus needed.

The flow diagram in Figure 3.2 shows the special arrangements for linked runs (NPROBE>1). In this mode PROBE may be thought of as an empty shell, which is filled only through the contents of the common blocks. The subroutine STORE has the task to store the common blocks and is thus called when it is time to read/write in a new common block.

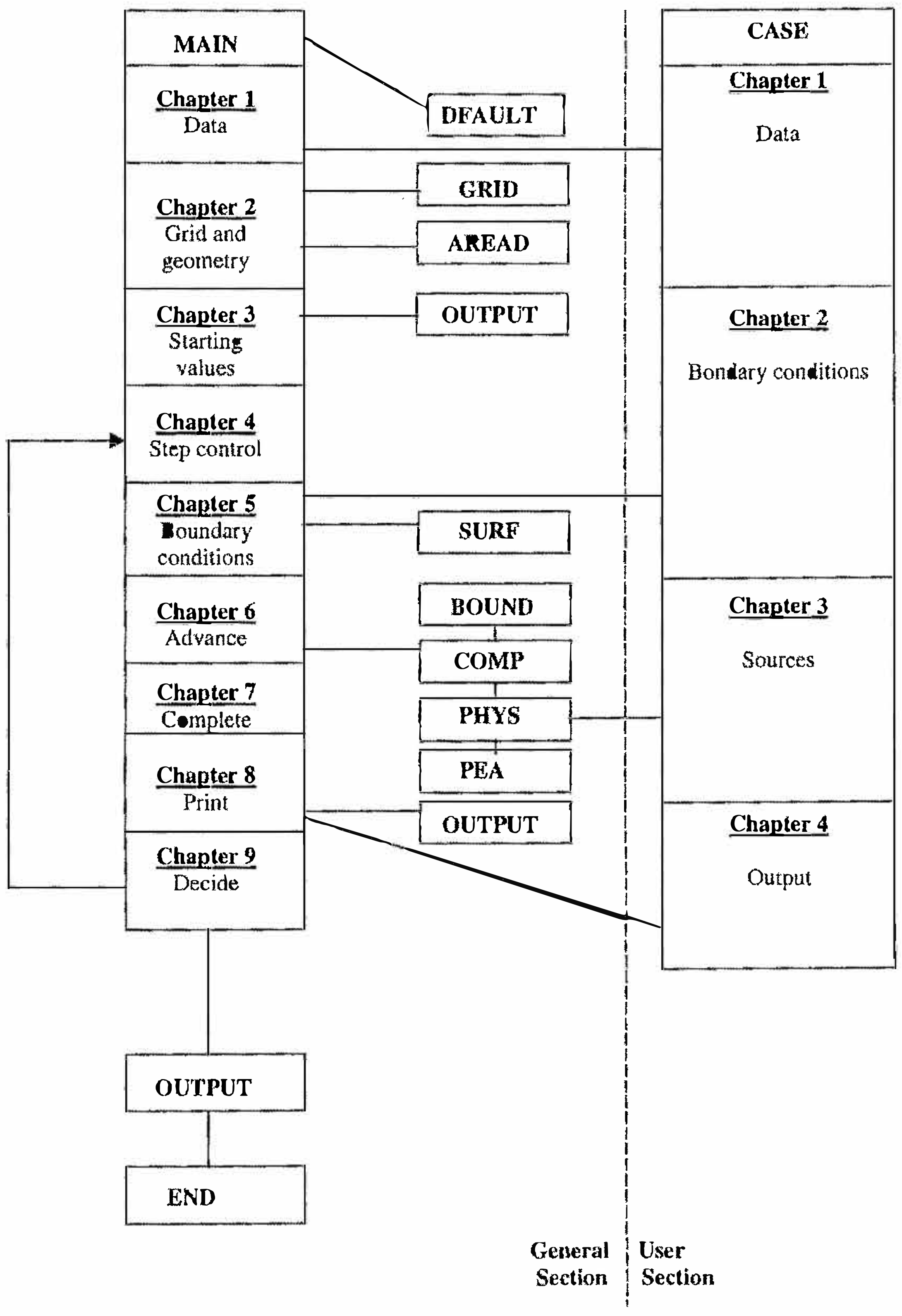


Figure 3.1. Flow diagram.

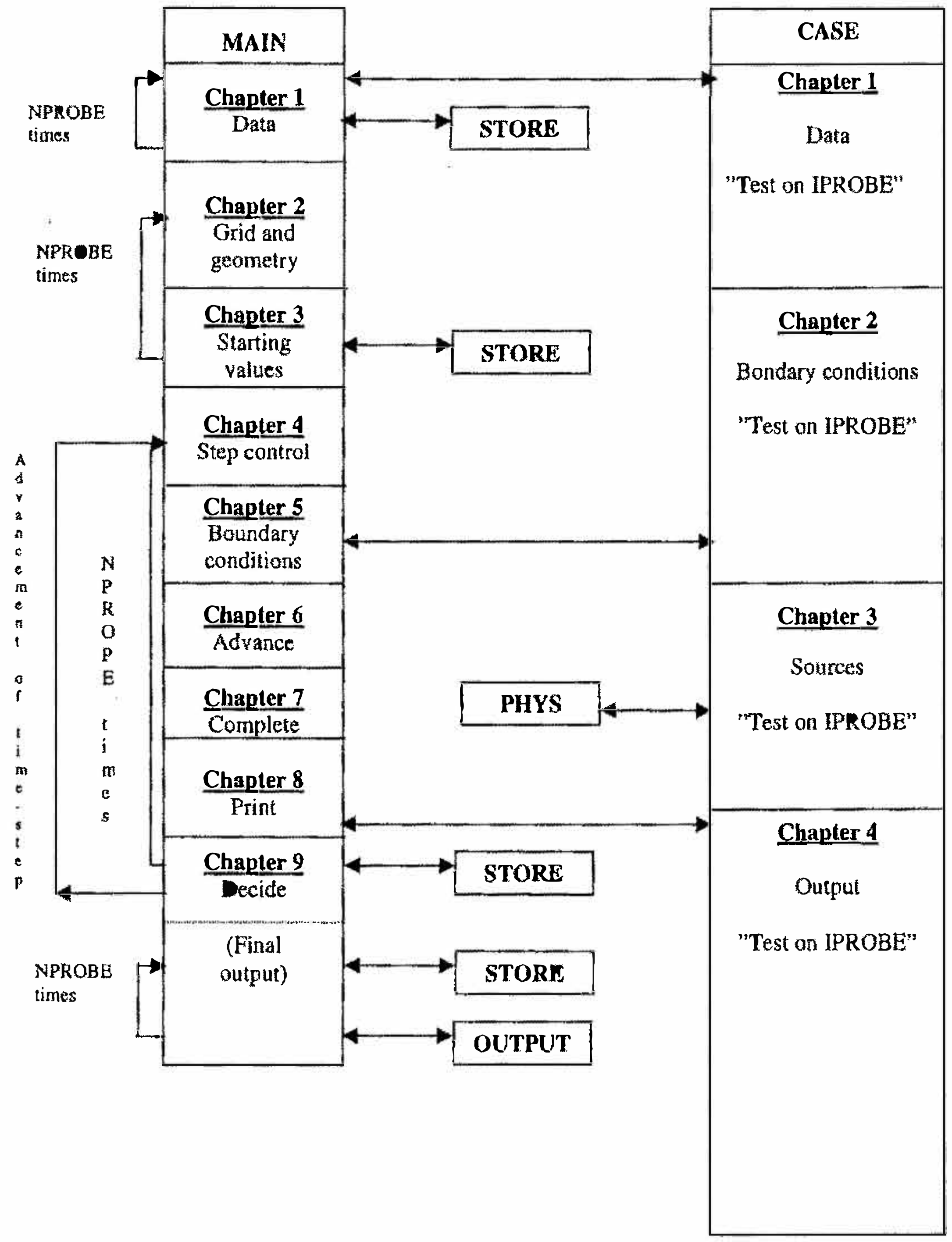


Figure 3.2. Flow diagram showing how linked runs (NPROBE>1) are performed.

3.2 General section subroutines

MAIN

For the purpose of describing only the main features of this subroutine, the special calls and loops for linked runs (NPROBE>1) have not been explained. The reader is referred to Figure 3.2 and suitable CASE-report for further details of use.

The subroutine that arranges and controls the calculation is called MAIN. In order to facilitate understanding, the different chapters in MAIN and their interaction with other subroutines are shown in the flow diagram. Chapter 1 provides input data initially set by DFAULT. Some of these data are modified by the user in subroutine CASE Chapter 1, which is the first subroutine called. The grid and geometry is specified in DFAULT and CASE, and necessary calculations using these data are done in the subroutines GRID and AREAD, which are called from Chapter 2 of MAIN. Chapter 3 initialises dependent variables and variables, which are functions of the dependent variables. The main DO-loop starts in Chapter 4 at the statement-number 400. In this chapter a new time-step is also calculated, according to the information given in CASE. Chapter 5 specifies time-dependent boundary conditions The CALL CASE(2) statement gives a link to CASE Chapter 2, where transient boundary conditions can be provided. Chapter 6 calls the COMP-subroutine, which performs the solution of the equations. When leaving this chapter, the calculation has thus advanced one time step. Then, in Chapter 7, density, temperature, and eddy viscosity are updated. Tests are also made to ensure that turbulent kinetic energy, k, and its dissipation rate ε, are positive. The reason for this is that negative values may be generated, because of strong buoyancy forces, during the calculation. A small positive value is then prescribed. Chapter 8 calls the subroutine OUTPUT and also calls CASE(4), where user specific output may be generated. In Chapter 9 tests are made in order to decide whether to continue or to terminate the calculation. If it is continued, a jump back to Chapter 4 is made.

DFAULT

This subroutine contains default values of all data that a user has to be concerned with. A detailed discussion of this subroutine is given in the next chapter of the manual.

GRID

The computational grid can be arranged in alternative ways (uniform, expanding, etc.) and that necessitates calculations of gridcell sizes, distances, etc. This is done in GRID.

AREAD

Lakes and reservoirs have a variation of horizontal area with depth. Idealised area-distributions can be generated from CASE and calculated in the subroutine AREAD.

OUTPUT

This subroutine, as the name indicates, is responsible for printout in various forms. Options, which are set in CASE, control the frequency of output in the form of integral parameters or profiles.

STORE

When linked runs (NPROBE>1) are performed, all information of a specific run is contained in the common blocks. The subroutine STORE is used to store common blocks, which are presently not active.

SURF

Necessary changes of the grid, when a moving surface is present, are done in this subroutine.

PHYS

As discussed in Chapter 2, all equations may be presented in the general form:

$$\frac{\partial \phi}{\partial t} + \frac{\partial}{\partial x_i} u_i \phi = \frac{\partial}{\partial z}\left(\Gamma_\phi \frac{\partial \phi}{\partial z}\right) + S_\phi.$$

To identify a variable one has to specify the transport coefficient, Γ_ϕ, and the source term, S_ϕ. This is done in subroutine PHYS. In Chapter A, the eddy-viscosity for gridnodes, F(I,JEMU), the Prandtl/Schmidt number, PRSCNU (I), and the effective viscosity, EMU(I), for cell boundaries are calculated. Also a reference transport coefficient DIFREF(I), which is the coefficient for momentum, is calculated. In Chapter B it is determined which variable is considered, and in the relevant chapter the transport coefficients and the source terms are supplied.

COMP

In this subroutine the execution of a forward step is performed, and it is therefore used for each dependent variable at each time or space step. In order to save computer time the F-array, which is the two-dimensional array where all variables are stored, is converted into a one-dimensional array. Necessary changes of indices are made in Chapter A. The results of subroutine PHYS are linked to COMP in Chapter B, where also the transport coefficients at the boundaries are included. The finite difference coefficients, derived in Appendix B, are calculated in Chapter C, and the equation is then solved in Chapter D. Depending on the type of boundary condition the flux or the value of the variable at a boundary is then calculated in Chapter E.

BOUND

The transport coefficients close to the boundaries are calculated assuming logarithmic or linear profiles. When using these profile assumptions, information about the hydrodynamic roughness length is needed. This information is given in CASE by specifying ROULLZ and ROULHZ. Transport laws for heat, salinity, and concentration include the Stanton number for the variable in question. These numbers are specified in CASE, in the array STANTN.

PEA

This subroutine contains the code of the Partial Elimination Algorithm, see Spalding (1976). The algorithm will allow a more stable solution for strongly coupled equations. In the present context it is the Coriolis' force that is responsible for the coupling.

3.3 User section subroutine

Only one subroutine, CASE, is subject to modifications by the user. Going back to the flow diagram it is seen that CASE is divided into four chapters, each one having a specific purpose. Instructions on use of CASE will be given in the next chapter of the manual.

From the flow diagram in Figure 3.2 one may note that the information given in CASE has to be selective for linked runs. This is done by a test ("an if-statement") on IPROBE, which is the running index for linked runs.

4. HOW TO USE PROBE

Suppose that PROBE has been installed on the user's computer and some cases have been run for test purposes. The user is thus in a position to set up a new problem. It is then recommended that the steps outlined in this chapter of the manual are followed.

4.1 Analysis of the problem considered

The first question to address is whether the case considered is in the class of flows solved by PROBE. If not, can a meaningful approximation be made? If PROBE is believed to be applicable, the next step is to characterise the problem in terms of equations and boundary conditions. It is further recommended that an analysis of length and time scales is carried out. This will be helpful when the grid size in space and time is selected. If something like sine-period can be identified, one may, as a rule of thumb, need 20 grid-cells or time-steps to resolve this period. Later, a more careful examination of grid size and time-step independence should always be made.

To summarise, it can be stated that a careful analysis of the problem and a well-founded expected behaviour of the process will significantly simplify the computational task.

4.2 Modification of default data

In this section the groups in DFAULT are explained and discussed. The values given in this subroutine are called the default values and are the values that will enter the calculation if not reset in CASE. The user is recommended to make notes about the modifications in each group that are needed for the case to be set up. The modifications will later be a part of the content of CASE. It should be emphasised that DFAULT belongs to the general section and should never be subject to direct changes.

Group 1

```
C*****GROUP 1. GRID IN SPACE AND TIME
C-----N=NUMBER OF GRID CELLS PLUS 2. MAXIMUM=NIM.
    N=NIM
    TIME=0.
    TLAST=1.E10
    LSTEP=10
C-----GRID DISTRIBUTION IN SPACE
C-----IGRID=INDEX FOR GRID
C       =1 GIVES UNIFORM GRID
C       =2 GIVES EXPANDING GRID FROM LOW Z
C       =3 GIVES EXPANDING GRID FROM HIGH Z
C       =4 INDICATES THAT THE GRID IS SPECIFIED IN CASE
C ----SEE MANUAL FOR DETAILS OF THE EXPANDING GRID
    IGRID=1
    CEXPG=1.1
    DO 11 IJK=1,NIM
    DZCELL(IJK)=0.
  11 CONTINUE
C-----TIME STEP VARIATION
C     A VARIABLE TIME STEP IS SPECIFIED BY THE TFRAC FIELD
C     TFRAC/10.,1.,200.,2.,16*0./GIVES A TIME STEP OF 1.0 S
C     THE FIRST 10 STEPS FOLLOWED BY 200 OF 2.0 S.
C     A CONSTANT TIME STEP IS OBTAINED BY SPECIFYING TFRAC(2)
C     IN CASE.

      DO 12 IJK=1,20
      TFRAC(IJK)=0.
   12 CONTINUE
      TFRAC(1)=1.E8

C   ITYPEF=INDEX FOR TYPE OF FLOW
C       =1 GIVES 1-D TRANSIENT FLOW (DEFAULT)
C       =2 GIVES 2-D PARABOLIC FLOW
    ITYPEF=1
```

The maximum number of grid points that can be specified is NIM, which is a number that can be set by the user in a parameter statement (see Section 5.5) and has a standard value of 100. The actual number of grid points is called N. This means (see Figure 2.1) that the standard number of grid cells is 98. A calculation can be terminated on two criteria; if the maximum number of time-steps, LSTEP, is reached or if the integration time, TIME has reached the maximum time, TLAST.

The expanding grid system is based on the geometrical series. The expansion factor, CEXPG, is the ratio between the height of the two neighbouring cells. Guidance for choosing CEXPG is given by the following formulas:

Size of first cell in expansion = ZDIM*(CEXPG-1)/(CEXPG^{N-2}-1)
Size of last cell in expansion = CEPG^{N-3}*ZDIM*(CEPG-1)/(CEXPG^{N-2}-1)

Where ZDIM is the physical dimension in the Z-direction.

The index ITYPEF is 1 for 1D transient flows and 2 for 2D parabolic steady flows.

Group 2

```
C*****GROUP 2. PHYSICAL DIMENSIONS
    XDIM=1.E10
    YDIM=1.E10
    ZDIM=1.E10
C-----VERTICAL AREA DISTRIBUTION
C
C-----INDARE=INDEX FOR AREA-DISTRIBUTION
C-----     =1 INDICATES UNIFORM AREA
C-----     =2 INDICATES LINEAR DISTRIBUTION
C-----     =3 INDICATES NON-LINEAR DISTRB.,SEE MANUAL
C-----     =4 DISTR. SPECIFIED IN CASE
    INDARE=1
    AREAHZ=1.0
    CEXPA=2.
```

The physical dimensions of the computational domain are given by ZDIM, XDIM and YDIM. ZDIM should always be reset in CASE, while XDIM and YDIM will only be modified for special cases like lakes and reservoirs.

The non-linear area distribution is generated with:

AREA(I) = (Z(I)/Z(N))**CEXPA*AREAHZ.

CEXPA is thus the expansion factor, which has typical values from
-0.5 to 2.0. The default value 2.0 is typical for Swedish lakes. The linear distribution is obtained, if INDARE is put to 2. CEXPA will then automatically be put to 1.0, and the above expression will then generate the linear distribution.

Group 3

```
C*****GROUP 3. DEPENDENT VARIABLES
C    F(I,JRHOU)=X-DIRECTION MOMENTUM
C    F(I,JRHOV)=Y-DIRECTION MOMENTUM
C    F(I,JH)=HEAT-ENERGY
C    F(I,JS)=SALINITY
C    F(I,JK)=TURBULENT KINETIC ENERGY
C    F(I,JD)=DISSIPATION OF TURBULENT KINETIC ENERGY
C    F(I,JC1)=CONCENTRATION NO.1
C    F(I,JC2)=CONCENTRATION NO.2
C    F(I,JC3)=CONCENTRATION NO.3
C    F(I,JC4)=CONCENTRATION NO.4
C    F(I,10+(NJM-10))=ADDITIONAL VARIABLES ACTIVATED FOR NJM>10.
C    F(I,JEMU)=DYNAMICAL EDDY VISCOSITY
C    F(I,JTE)=TEMPERATURE
    JRHOU=1
    JRHOV=2
    JH=3
    JS=4
    JK=5
    JD=6
    JC1=7
    JC2=8
    JC3=9
    JC4=10
    DO 31 IJK=1,NJM
    SOLVAR(IJK)=.FALSE.
  31 CONTINUE
    JEMU=NJMP1
    JTE=NJMP2
```

PROBE solves for up to 30 dependent variables in the standard set up. If more dependent variables are needed, a parameter statement (see Section 5.5) has to be reset. NJM (equal to 30 in the standard set up) defines the number of variables accounted for. Two more, dynamical eddy viscosity and temperature, are stored in the F-array. It

Group 7

```
C*****GROUP 7. SOURCE TERMS
C
C----CORIOLIS PARAMETER
    CORI=1.E-4
C-----PRESSURE GRADIENTS
C   INDPX=INDEX FOR PRESSURE GRADIENTS IN X-DIRECTION
C        =1 GIVES PRESCRIBED CONSTANT PRESSURE
C          GRADIENTS ,DPDXP.
C        =2 GIVES PRESCRIBED MASSFLOW,RHOUP.ONLY
C          RELEVANT FOR STEADY STATE PROBLEMS.
C        =3 GIVES PRESSURE GRADIENT DEVELOPMENT ACCORDING TO
C          HORIZONTAL EXTENT OF WATERBODY.ONLY RELEVANT TO
C          LAKES AND RESERVOIRS.
C        =4 INDICATES THAT THE PRESSURE GRADIENTS ARE TO BE
C          READ FROM SEPARATE FILE AS A TIME SERIES.
C        =-1,-2,-3 OR -4 AS ABOVE,BUT WITH BUOYANCY DAMPING
C          OF PRESSURE GRADIENTS(EFFECT OF TILTED TERMOCLINE).
C   INDPY=SAME FOR Y-DIRECTION
    RHOUP=0.
    RHOVP=0.
    DPDXP=0.
    DPDYP=0.
    PFILT=1.
    INDPX=1
    INDPY=1
C-----IN- AND OUTFLOWS.
C-----SEE MANUAL FOR INSTRUCTIONS ON USE
    DO 71 IJK=1,NIM
    QZ(IJK)=0.
    QINFL(IJK)=0.
    QOUTFL(IJK)=0.
    DO 72 IKJ=1,NJM
    PHIIN(IJK,IKJ)=0.
 72 CONTINUE
 71 CONTINUE
C-----SHORT-WAVE RADIATION
C   ASSUMED TO PENETRATE THE WATER BODY.
C   FLXRAD=SHORT-WAVE RADIATION.
C   RADFRA=FRACTION ASSUMED TO BE A BOUNDARY FLUX
C   BETA=EXTINTION COEFFICIENT
    FLXRAD=0.0
    RADFRA=0.4
    BETA=0.1
```

The details of the technique of calculating pressure gradients are given in Appendix A. When the option INDPX (or INDPY) = 2 is used, one may get a diverging solution, which never reaches a steady state. The user must then reduce the time-step and the factor PFILT, which produces an under relaxation of the development of the pressure gradients.

Unfortunately a trial and error procedure must be carried out to find the optimum values on the time-step and PFILT. When INDPX (or INDPY) equals 3, or −3, a non-unity PFILT has another implication. The pressure-gradient formula for lakes and reservoirs simulates seiches with periods based on the dimensions of the water body. Often the period is of the order minutes, which requires a time-step of the order 10 seconds (one tenth of the seiche period). If PFILT is put to, for example, 0.2, the seiche period will be 5 times larger, and more economical time-step may be used. It should be noted that the main effects of the pressure gradients will still be present. Test calculations should be performed to establish whether this filtering of pressure significantly affects the overall behaviour of, for example, a seasonal stratification.

The volume flux and the properties of in- and outflows can be specified from CASE.

The volume fluxes are specified in QINFL (I) and QOUTFL (I) for in- and outflows respectively. The in- and outflows generate a vertical volume flux, which is calculated from an application of the continuity equation cell by cell. Properties only need to be specified for inflows and are given in
PHIIN (I, J). If QINFL $\neq$ QOUTFL, when integrated over the depth, the moving surface option needs to be activated (see Group 13).

Incoming short-wave radiation varies during the day and should therefore be specified in CASE, Chapter 2. Examples of how this is done can be found in CASE reports on thermocline development.

Group 8

```
C*****GROUP 8. INITIAL DATA
      DO 81 IJK=1,NJM
      DPDX(IJK)=0.
      DPDY(IJK)=0.
      FW(IJK)=0.
      DO 82 IKJ=1,NJMP2
      F(IJK,IKJ)=0.
   82 CONTINUE
   81 CONTINUE
C-----INITIALISE DEPENDENT VARIABLES
C     ISTPR=INDEX FOR STARTING PROFILES
C          =1 PROFILES ARE SPECIFIED WITH VST1(1-NJM)-ZST2(1-NJM)
C             SEE MANUAL.
C          =2 PROFILES ARE SPECIFIED IN CASE WITHOUT THE USE
C             OF VST1(1-NJM)-ZST2(1-NJM).
C     --NOTE:DEFAULT VALUE FOR ALL VARIABLES IS 0.0.
      ISTPR=1
      DO 83 IJK=1,NJM
      VST1(IJK)=0.
      VST2(IJK)=0.
      ZST1(IJK)=0.
      ZST2(IJK)=0.
   83 CONTINUE
```

All variables in the F-array are here given the default value 0.0. Two alternatives are available for the specification of non-zero initial profiles. If ISTPR equals 1, profiles are specified according to Figure 4.1, while ISTPR equal 2 indicates that the profiles are specified directly in the F- array.

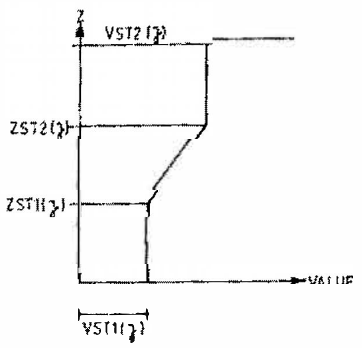


Figure 4.1. Specification of initial profiles of dependent variables.

It is only the dependent variables that should be initialised; density, temperature, eddy viscosity, etc. are calculated as functions of the dependent variables in subroutine MAIN. In this context it is also necessary to remember that momentum and heat energy are the dependent variables, not velocity and temperature.

Group 9

```
C*****GROUP 9. BOUNDARY CONDITIONS
C
C-----ITYPEH=INDEX FOR TYPE OF BOUNDARY AT HIGH Z
C       =1 GIVES SOLID WALL(STATIONARY OR MOVING)
C       =2 GIVES SYMMETRY LINE
C   ITYPEL=SAME FOR LOW Z BOUNDARY
C
C-----IKBHZ(J)=INDEX FOR KIND OF BOUNDARY CONDITION FOR
C         VARIABLE J AT HIGH Z BOUNDARY
C        =1 GIVES PRESCRIBED VALUE
C        =2 GIVES PRESCRIBED FLUX
C   IKBLZ(J)=SAME FOR LOW Z BOUNDARY
C-----ITRHZ(J)=INDEX FOR TIMEDEPENDENCE OF BOUNDARY FOR
C         VARIABLE J
C        =1 GIVES STATIONARY CONDITIONS
C        =2 GIVES TRANSIENT CONDITIONS SPECIFIED FROM CASE-
C         SUBROUTINE.SEE MANUAL FOR INSTRUCTIONS ON USE.
C        =3 GIVES TRANSIENT CONDITIONS READ FROM FILE
C   ITRLZ(J)=SAME FOR LOW Z BOUNDARY
C-----IKBOT(J)=INDEX FOR KIND OF BEHAVIOR AT BOTTOM FOR VARIABLE J
C         ONLY RELEVANT FOR CASES WITH VERTICAL AREA-DISTRIB.
C        =1 GIVES "CONSERVATIVE" CONDITION.SEE MANUAL.
C        =2 GIVES "NON-CONSERVATIVE" CONDITION.SEE MANUAL.
C-----SPECIFICATION FOR STATIONARY BOUNDARY CONDITIONS
C
C-----SPECIFICATION FOR TRANSIENT CONDITIONS(ITRHZ OR ITRLZ=2).SEE MANUAL
C
C  ----SPECIFICATION OF WALL-FKN PARAMETERS.
C
    ITYPEH=1
    ITYPEL=1
    DO 91 IJK=1,NJM
    IKBHZ(IJK)=2
    IKBLZ(IJK)=2
    ITRHZ(IJK)=1
    ITRLZ(IJK)=1
    IKBOT(IJK)=1
    FLUXHZ(IJK)=0.
    FLUXLZ(IJK)=0.
    V1HZ(IJK)=0.
    V2HZ(IJK)=0.
    V3HZ(IJK)=0.
    V4HZ(IJK)=0.
    V5HZ(IJK)=0.
    V1LZ(IJK)=0.
    V2LZ(IJK)=0.
    V3LZ(IJK)=0.
    V4LZ(IJK)=0.
    V5LZ(IJK)=0.
    STANTN(IJK)=1.E-3
 91 CONTINUE
    IKBOT(1)=2
    IKBOT(2)=2
    IKBOT(5)=2
    IKBOT(6)=2
    STANTN(1)=1.
    STANTN(2)=1.
    STANTN(3)=0.05
    STANTN(5)=1.
    STANTN(6)=1.
    CAPPA=0.4
    C3B=9.
    ROULHZ=0.
    ROULLZ=0.
```

If ITYPEH is put to 1, a wall is assumed to be present at the high Z boundary. This will activate the wall functions in subroutine BOUND. The symmetry line conditions can be used when a zero-flux condition prevails at the boundary in question.

Transient boundary conditions can be specified for all dependent variables according to the following instructions:

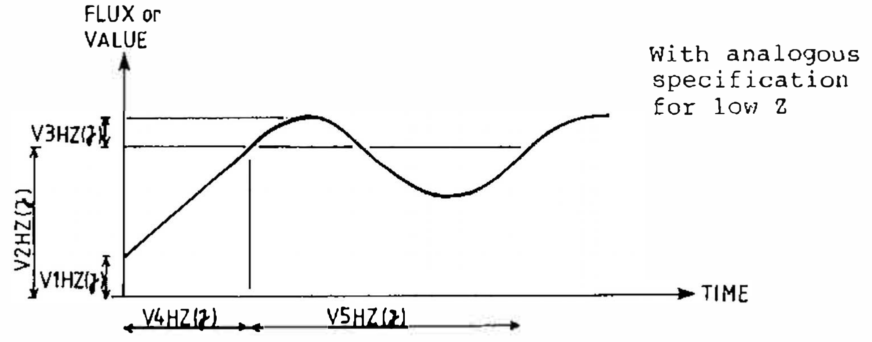


Figure 4.2. Specification of transient boundary conditions.

The user must, of course, have made a decision , whether the boundary condition should be specified as value or a flux when the values above are given. Alternatively the user may specify transient boundary conditions in CASE, Chapter 2.

When a variable horizontal area is specified, the index IKBOT has to be considered. If IKBOT is put to 1, a conservative condition is assumed, which means a zero flux through the bottom area for all cells, see Figure 4.3. This may be suitable for heat and salinity, while momentum is lost in the bottom contact, which indicates that IKBOT should be put to 2 for momentum equations. Appendix B explains this point further.

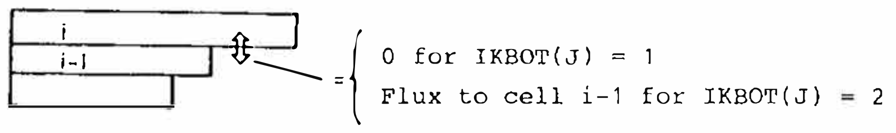


Figure 4.3. Meaning of IKBOT.

Wall functions require information about the roughness of the surfaces. This is specified in ROULHZ and ROULLZ, which are the roughness lengths, z, at high and low Z. A zero value indicates that the surface is hydrodynamically smooth. Heat, salinity, and concentrations are at a wall assumed to obey the following transport law:

$$\text{FLUX}(\phi) = \text{STANTN}\ (\phi) U_* \Delta\phi$$

Where STANTN(ϕ) is the Stanton number for variable ϕ, $\Delta\phi$ is the difference in ϕ between the boundary and the first cell, U_* is the friction velocity.

Group 10

```
C*****GROUP 10. LIMITS AND NUMBERS
     EMTMIN=1.E-6
     FKMIN=1.E-15
     FDMIN=1.E-15
     TAUMIN=1.E-3
     KINDAV=1
```

These numbers are minimum values that ensure that the variables considered never become negative. Normally they should not be changed.

Group 11

```
C*****GROUP 11. PRINT OUT
C--------PRINT CONTROL
C  --SET ITPLOT=2 FOR CROSS-STREAM PLOT, =1 FOR NO PLOT
     ITPLOT=2
C  --SET NSTAT,NPROF,NPLOT TO NUMBER OF STEPS BETWEEN OUTPUT OF
C    STATION VALUES,PROFILES AND CROSS-STREAM PLOTS RESPECTIVELY
     NSTAT=10
     NPROF=50
     NPLOT=100
C  --SET INIOUT .FALSE.  FOR NO INITIAL OUTPUT
     INIOUT=.TRUE.
C
C---- SELECT PROFILES TO BE PRINTED AND PLOTTED.
C-----U,V,T,S,1C,2C,3C,4C,K,E,EMU,SIGM,DPDX,DPDY,W,PRSCN,RIF,N,UW,VW
C    1,2,3,4, 5, 6, 7, 8,9,10,11, 12, 13, 14,15, 16,17,18,19,20
C-----PRINTED
C-----PLOTTED
     DO 111 IJK=1,20
     PRPROF(IJK)=.FALSE.
     PLPROF(IJK)=.FALSE.
 111 CONTINUE
C
C-----PARTICLE TRACKING.SEE MANUAL.
C-----INDPT=INDEX FOR PARTICLE TRACKING
C        =0 GIVES NO TRACKING
C        =1-4 ONE TO FOUR PARTICLES ARE TRACKED
C
     INDPT=0
     ILEVEL(1)=0
     ILEVEL(2)=0
     ILEVEL(3)=0
     ILEVEL(4)=0
     IPSAVE=1000
```

PRPROF is a logical array, which selects variables for printing of profiles. The particle tracking routine is activated by putting INDPT to 1 – 4, then 1 to 4 particles are to be tracked. Also ILEVEL, which is an array dimensioned to four, needs to be considered. If, for example, ILEVEL(2) =30, particle number 2 will be on level Z(30). By IPSAVE an interval, between which coordinates are to be saved, is specified. If IPSAVE = 10, the coordinates will be stored every tenth time step. Maximum number of steps that can be stored are 500. Examples on the use of the particle tracking routine can be found in CASE-reports.

Group 12

```
C*****GROUP 12.LINKED RUNS.
     DO 121 IJK=1,NPM
     NSTPDT(IJK)=1
 121 CONTINUE
     NPROBE=1
```

For linked runs, NPROBE is the number of runs to be done. NSTPDT(J) provides a means of having different time steps in different runs. One run should have NSTPDT =

1, which then indicates that this run should have the specified time step, DT. If another run has, as an example, NSTPDT(5) = 4, it gives a time step of DT/4 for run number 5. Note that it is not recommended to specify different time steps in different runs directly by TFRAC(2), due to the arranged interactions between the runs and the formulation of output sequences.

Group 13

```
C*****GROUP 13. MOVING FREE SURFACE.
   MOVE=.FALSE.
   ZSSTRT=0.
   PREEVA=0.
C*****************************************************************
   RETURN
   END
```

MOVE is a logical, which is set to true, if a moving free surface is present. PREEVA is precipitation and evaporation with dimension [m/s] and positive along the vertical space coordinate. Rain on a lake surface is thus specified in [m/s] and has a negative value. ZSSTRT means "Z-surface start" and gives the initial water surface level. This value needs to be smaller than ZDIM, which is the maximum surface level that is to be considered.

4.3 The CASE subroutine

Modifications of default values are included in Chapter 1 of CASE.

Chapter 2 of CASE provides a link to the MAIN subroutine. The link is intended for the supply of transient boundary conditions, which can not be handled by the prepared functions. An example is meteorological data obtained from field measurements, which in this chapter should be read from a separate file and be included as transient boundary conditions. Additional source terms should be supplied in Chapter 3, which provides a link to the subroutine PHYS. A call is made from every dependent variable, and the user has to select the appropriate variable to be supplied with extra source terms. The following example shows a typical coding sequence:

```
      IF (J.NE.JC1) RETURN
      DO 10 I = 2,NM1
      FJC1N = F(I+1,JC1)*WSED
      FJC1S = F(I,JC1)*WSED
      IF(I.EQ.2) FJC1S = 0.0
      IF(I.EQ.NM1) FJC1N = 0.0
10    SI(I) = SI(I) - (FJC1N - FJC1S)/DZCELL(I)
      RETURN
```

A source term for variable C1, which describes sedimentation with the settling velocity WSED, is thus added. Further examples can be found in CASE-reports.

Additional output can be generated from Chapter 4 of CASE. The call to this chapter is also from MAIN but this time from the position where the standard output is called for. This ensures that the generated output is at the same integration time as the standard output. Extra output may be useful, for example, when the dependent variables are requested in a non-dimensional form. For linked runs one needs, as mentioned earlier, to select the correct run (test on IPROBE) when providing information in subroutine CASE. Examples can be found in CASE-reports.

4.4 Test calculation

It is advisable to make a test calculation with LASTEP = 10 to make some preliminary checking. Assuming that compilation errors have been eliminated and that numbers are produced, the user should proceed through the following steps.

- Check the section "PRINCIPAL DATA USED". Is everything according to expectations?
- Check grid and initial profiles in the profile output called "INITIAL PROFILES".
- Is the output generated after 10 steps according to expectations?

If no objections have been raised to the results produced, it is time to proceed with a longer run. If the output shows an unrealistic or unexpected behaviour, one has to go through the process of analysing and coding again.

5. ADVICE ON EFFECTIVE USE

5.1 Grid independence in space and time

A coarse grid, i.e. few cells and large time steps, needs less computer time and should be used during the preliminary stages of the calculations. However, only the grid-independent solution, in space and time, represents the true implication of the differential equations. A systematic refinement of the grid must therefore always be carried out, if a claim that the differential equations have been solved is to be made. It is thus recommended that a coarse grid, which typically could be 15 grid-cells, is used in the preliminary stage and a grid refinement study is carried out before the final calculations are performed.

5.2 Use of integral checks

Integral checks for heat and salinity are supplied by PROBE. These should always be studied, as they may indicate errors in boundary conditions or in the stability of the numerical solution. Note that the integral checks are not valid, if extra source or sink terms are added to the equations for heat and salinity.

When concentration equations are solved for, the user is advised to make estimates of the integral balances, when possible.

5.3 Verification studies

In order to get confidence in predicted results, some form of verification is needed. Some or all of the following steps may then be considered.

- Is it possible to idealise the situation in such way that an analytical solution can be obtained? If so, one may set up PROBE to solve the same situation, and an agreement that is only limited by the grid dependence should be the result. One should, of course, never expect more than 5 – 6 correct figures, due to the limitation of the computer.
- Are there any laboratory experiments, which consider the basic physical processes, available? If so, these may be very useful for verification studies, as boundary

conditions, initial conditions, and the quality of the recording of the process are normally known with good accuracy.

- Are there any other model-predictions for the problem considered available? If so, and if these may be regarded as "well established and accepted", one may consider to repeat these.
- The final test is, of course, the application to the environmental problem itself. This is the most difficult part with transient and often incomplete boundary conditions. This makes it often hard to judge the degree of success when comparing predicted and measured behaviour.

5.4 Causes of diverging solutions

A diverging solution is normally easy to detect; integral checks are not fulfilled, and unrealistic profiles are predicted. Assuming that the user by studying CASE and initial output, has checked that the problem specification is correct, one may consider the following points:

- Has it been firmly established that a solution to the problem, as it has been defined, exists? One should, in this context, be particularly observant on the prescribed boundary conditions.
- Have all length and time scales in the problem been identified? If a typical period in space or time can be found, one may need 10 – 20 grid-cells or time-steps to resolve the process.
- If a lake or reservoir is considered, the seiche period will enter through the pressure gradient formula. Once again a time-step of the order one tenth of the seiche period is needed.
- If a sedimentation process is considered, one should estimate the time-step required with respect to the settling velocity. The time it takes for particle to travel across a grid-cell may be used as an estimate of the time-step required.

5.5 Some advice on mounting PROBE

Test installation of the present version of PROBE have been carried out on VAX 8600, UNIVAC 1108, CRAY, SUN and PC:s. The experiences from these installations can be summarised as:

- The inclusion of the parameter statements and the common-blocks needs to be arranged according to the computer used.
- The unlabelled common-block IA1 in subroutine STORE needs to be dimensioned to NSTR1 (and not 1) on some computers. Note that one then needs to recompile the code, when the maximum number of cells, equations or runs are reset.
- Of the two common-blocks, which are to be included in most subroutines, one is unlabelled. This one corresponds to IA1 in subroutine STORE. It may be necessary, on some computers, to have these two common-blocks as labelled and then also, as mentioned above, give IA1 the dimension NSTR1.
- TFRAC(1) is the number of time steps with time step TFRAC(2). TFRAC(1) is converted into an integer in the code. The default value 10^8 may be too large for some computers (especially PCs) to convert into an integer . Reset TFRAC(1) in CASE, Chapter 1, if this is the case.

When the code has been mounted and found to reproduce results from test cases, the user may wish to change the pre-set maximum number of cells, equations or linked runs. This is done in the parameter statements proceeding the common-blocks. When any of these values (NIM, NJM or NPM) is reset, one also needs to reset NSTOR1 (for the first common-block) and NSTORE (the size of both the common-blocks, NSTOR1 + 107). NSTOR1 is calculated according to:

NSTOR1 = (27 * NIM + 27 * NJM + NIM * (NJM+2) + NIM * NJM + NPM + 64), which is equal to 9804 for the pre-set values.

6. CONCLUDING REMARKS

It is time to recall a sentence from the introduction, stating that computational fluid dynamics seldom becomes standard or simple. It is therefore not possible, and has not been the objective, to write a manual that ensures safe use of PROBE. Instead it is hoped that it will assist a potential user, who is expected to add his/her own insight and intelligence.

7. NOMENCLATURE

The following glossary of FORTRAN variable names is arranged with reference to the GROUPS in the subroutine DFAULT.

Group	Name	Type	Meaning
1	N	Integer	Number of grid points
1	Time	Real	Integration time
1	TLAST	Real	Maximum integration time
1	LSTEP	Integer	Maximum number of time steps
1	IGRID	Integer	Index for grid
1	CEXPG	Real	Expansion factor for grid
1	DZCELL(NIM)	Real array	Vertical dimension of cells
1	TFRAC (20)	Real array	Specification of time step
1	ITYPEF	Integer	Type of flow, 1D or 2D
2	ZDIM	Real	Physical dimension in Z-direction
2	XDIM	Real	Physical dimension in X-direction
2	YDIM	Real	Physical dimension in Y-direction
2	INDARE	Integer	Index for area-distr.
2	AREAHZ	Real	Horizontal area of top cell
2	CEXPA	Real	Expansion factor for area-distr.
3	F(NIM, NJM+2	Real array	Dependent variables, eddy viscosity and temperature for all cells
3	SOLVAR (NJM)	Logical array	Select variables to be solved for
4	CPHEAT	Real	Specific heat
4	RHOREF	Real	Reference density
4	EMULAM	Real	Laminar viscosity
4	PRL (NJM)	Real array	Laminar Prandtl/Schmidt numbers
4	AGRAV	Real	Acceleration due to gravity

Group	Name	Type	Meaning
5	C(1-5)RHO	Real	Coefficient in eqn of state
5	TREF	Real	Temperature of max. density
6	ITURBM	Integer	Index for turbulence model
6	IPRSC	Integer	Index for Prandtl/Schmidt number
6	EMUCON	Real	Constant turbulence viscosity
6	PRT(NJM)	Real array	Turbulent Prandtl/Schmidt number
6	CD->CKSURF	Real	Constants in turbulence model
7	CORI	Real	Coriolis' parameter
7	INDPX	Integer	Index for pressure gradients
7	INDPY	Integer	Index for pressure gradients
7	RHOUP	Real	Prescribed mass flow
7	RHOVP	Real	Prescribed mass flow
7	DPDXP	Real	Prescribed pressure gradient
7	DPDYP	Real	Prescribed pressure gradient
7	PFILT	Real	Pressure filtering coeff.
7	QZ(NIM)	Real array	Vertical volume flux
7	QINFL(NIM)	Real array	Inflow
7	QOUTFL(NIM)	Real array	Outflow
7	PHIIN(NIM, NJM)	Real array	Properties of inflow
7	FLXRAD	Real	Short wave radiation
7	RADFRA	Real	Fraction of RADIN absorbed at surface
7	BETA	Real	Extinction coefficient
8	F(NIM, NJM+2)	Real array	See Group 3
8	DPDX(NIM)	Real array	Pressure gradient, X-dir.

Group	Name	Type	Meaning
8	DPDY(NIM)	Real array	Pressure gradient, Y-dir.
8	ISTPR	Integer	Index for starting profiles
8	VST1(NJM) VST2(NJM)	Real array	Vales for starting profiles
8	ZST1(NJM) ZST2(NJM)	Real array	Z-levels for starting profiles
9	ITYPEH	Integer	Index for boundary at high Z
9	ITYPEL	Integer	Index for boundary at low Z
9	IKBHZ(NJM)	Integer array	Index for boundary conditions at high Z
9	IKBLZ(NJM)	Integer array	Index for boundary conditions at low Z
9	ITRHZ(NJM)	Integer array	Index for time-dependence at high Z
9	ITRLZ(NJM)	Integer array	Index for time-dependence at low Z
9	IKBOT(NJM)	Integer array	Index for behaviour at bottom
9	FLUXHZ(NJM)	Real array	Flux at high Z
9	FLUXLZ(NJM)	Real array	Flux at low Z
9	V1HZ(NJM) V5LZ(NJM)	Real array	Specify transient boundary conditions
9	STANTN(NJM)	Real array	Stanton number
9	CAPPA	Real	Von Karman´s constant
9	C3B	Real	Constant in wall-function
9	ROULHZ	Real	Roughness length at high Z
9	ROULLZ	Real	Roughness length at low Z
0	EMTMIN	Real	Min. value for eddy visc.

Group	Name	Type	Meaning
10	FKMIN	Real	Min. value for turb. energy
10	FDMIN	Real	Min. value for dissipation
10	TAUMIN	Real	Min. shear for wall-functions
10	KINDAV	Integer	Index for harmonic or aritmetic averaging of diffusion coefficient
11	NSTAT	Integer	Steps between station values
11	NPROF	Integer	Steps between profiles
11	PRPROF (20)	Logical array	Selected printed profiles
11	INDPT	Integer	Index for particle tracking
11	ILEVEL(4)	Integer array	Levels for tracking
11	IPSAVE	Integer	Steps between saved coordinates
11	INIOUT	Logical	Controls initial output
12	NSTPDT(NPM)	Integer array	Numbers of steps on each time step for each run
12	NPROBE	Integer	Number of linked runs
13	MOVE	Logical	Activates the moving surface mode
13	ZSSTRT	Real	Initial water surface level
13	PREEVA	Real	Precipitation/evaporation

8. REFERENCES

8.1 Literature references for the manual

Kantha, L.H., Phillips, O.M., and Azad, R.S. (1977).
On turbulent entrainment at a stable density interface.
J. Fluid Mech., Vol. 79, part 4, p. 753.

Launder, B.E. (1975).
On the effects of a gravitational field on the transport of heat and momentum.
J. Fluid Mech., Vol. 67, part 3, p. 569.

Nordblom, O. (1997).
Numerical simulation of the atmospheric surface layer. Master's thesis. 1997:351 CIV.
Luleå University of Technology.

Omstedt, A., Sahlberg, J., and Svensson, U. (1983).
Measured and numerically simulated autumn cooling in the Bay of Bothnia.
Tellus, 35 A, pp231 240.

Patankar, S.V. (1980).
Numerical heat transfer and fluid flow.
Hemisphere Publishing Corporation, Washington, New York, London.

Rodi, W. (1980).
Turbulence models and their application in hydraulics -a state of the art review.
International association for Hydraulic Research (IAHR), Delft, The Netherlands

Spalding, D.B. (1977).
GENMIX –A general computer program for two-dimensional parabolic phenomena.
Imperial College, London, Mech. Eng. Dept., Report No. HTS/77/9.

Spalding, D.B. (1976).
Basic equations of fluid mechanics and heat and mass transfer, and procedures for their solution.
Imperial College, Heat Transfer Section, Report HTS/76/6.

Spalding, D.B., Gunton, M.C., Rosten, H.J., and Tatchel, D.G. (1983).
PHOENICS. An instruction manual.
CHAM, London, Report CHAM TR/75.

Svensson, U. (1978)
A mathematical model of the seasonal thermocline.
Lund Inst. of Technology, Dept. Of Water Res. Eng., Report No. 1002, Lund, Sweden.

8.2 Presently available CASE-reports.

A. Basic fluid mechanics, heat and mass transfer

A 1. The laminar plane Poiseuille flow.
Urban Svensson (1984).

A 2. The constant viscosity Ekman layer.
Urban Svensson (1984).

A 3. The plane Couette flow.
Anders Omstedt (1984).

A 4. The wind-induced channel flow.
Jörgen Sahlberg (1984).

A 5. The turbulent plane Poiseuille flow.
Urban Svensson (1984).

A 6. The extrainment experiment by Kantha, Phillips, and Azad.
Jörgen Sahlberg (1984).

A 7. The entrainment experiment by Deardorff, Willis, and Lilly.
Urban Svensson (1984).

A 8. The frazil ice experiments by Tsang and Hanley
Anders Omstedt (1984).

A 9. Dispersion of marked fluid elements.
Urban Svensson (1985).

B. Oceanography

B 1. The homogeneous Ekman layer.
Anders Omstedt (1984)

B 2. Mixed layer deepening in a continuously stratified rotating fluid.
Jörgen Sahlberg (1984).

B 3. The Ekman boundary layer stratified with respect to salinity and temperature.
Anders Omstedt (1984).

B 4. Supercooling and ice formation in a turbulent Ekman layer.
Anders Omstedt (1984).

B 5. Wind-forced sea ice motion.
Anders Omstedt (1987).

B 6. The development of a seasonal thermocline in the ocean.
Urban Svensson (1984).

B 7. Inertial trajectories in the Baltic.
Urban Svensson (1984).

B 8. Frazil ice and grease ice formation in the upper layers of the ocean.
Anders Omstedt (1985).

B 9. The diurnal thermocline.
Göran Lindström (1985).

B 10. Dispersion in an ocean Ekman layer.
Urban Svensson (1985).

B 11. The water level response in two coupled ocean basins to tides and rivers.
Anders Omstedt (1986).

B 12. The exchange of properties between two ocean basins.
Anders Omstedt (1986).

B 13. Fjords with wide sounds.
Anders Omstedt (1986).

B 14. Autumn Cooling in the Kattegat, the Belt Sea, Öresund and the Arkona Basin.
Anders Omstedt (1986).

B 15. Vertically coupled Ekman layers.
Urban Svensson (1986).

B 16. Wind-forced sea ice motion during freezing and melting.
Anders Omstedt (1987).

B 17. The ocean boundary layer beneath drifting melting ice.
Anders Omstedt (1988)

B 18. Fjord exchange driven by coastal variations.
Anders Omstedt (1988)

B 19.Seasonal variations of a sea ice cover.
Anders Omstedt (1990)

B 20. Seasonal cycle of salinity in the Mackenzie Shelf/Estuary.
Anders Omstedt (1993)

C. HYDROLOGY

C 1. Autumn cooling in a lake.
Jörgen Sahlberg (1984).

C 2. Thermocline development in a lake.
Urban Svensson (1984).

C 3. Thermocline development in a reservoir with in- and outflows.
Urban Svensson (1984).

C 4. Ice covered lake with sediment heat flux.
Jörgen Sahlberg (1984).

C 5. Heat loss in an ice covered lake due to a heat pump.
Jörgen Sahlberg (1984).

C 6. Formation of frazil ice, slush and anchor ice in rivers.
Anders Omstedt.

C 7. Transient groundwater flow.
Urban Svensson (1985).

C 8. Heat and mass transfer in unsaturated soils.
Urban Svensson (1985).

C 9. Limestone treatment of acid lakes.
Urban Svensson (1985).

C 10. Dynamics of coupled reservoirs.
Urban Svensson (1986).

C 11. The seasonal freezing and thawing of soils.
Urban Svensson (1987).

C 12. Coupled unsaturated and saturated groundwater flow.
Göran Lindström and Urban Svensson (1987).

D. METEOROLOGY

D 1. The steady neutral atmospheric Ekman layer.
Urban Svensson (1984).

D 2. Dispersion of a chimney plume.
Urban Svensson (1985).

8.3 Papers and reports based on the use of PROBE

Andreasson, P., and Svensson, U. (1989)
A mathematical simulation of energy conversions in a fully developed channel flow.
Journal of Hydraulic Research. Vol. 27, No. 3, pp 401-416.

Andreasson, P., and Svensson, U. (1992)
A note on a generalized eddy-viscosity hypothesis.
Journal of Fluids Engineering, Vol. 114, pp 463-466.

Andreasson, P (1992)
Energy conversions in turbulent wind-induced countercurrent flow.
Journal of Hydraulic Research. Vol. 30, No. 6, pp 783-799.

Broman, B. (1984).
Consequences of heat extraction by the heat pump method in Lake Drevviken.
The Swedish Meteorological and Hydrological Institute.

Broström, G. 1997.
Interaction between mixed layer dynamics, gas exchange and biological production in the oceanic surface layer with application to the northern North Atlantic.
PhD thesis. Department of Oceanography, Göteborg University, Gothenburg, Sweden.

Broström, G. 1998.
A note on the C/N and C/P ratio of the biological production in the Nordic seas.
Tellus, 50B, 1,pp 93-109.

Broström, G., and Rodhe, J. 1996.
Velocity shear and vertical mixing in the Ekman layer in the presence of a horizontal density gradient.
Continental Shelf Research, 16, pp 1245-1257.

Eidner, G., Utnes, T., and McClimans, T.A. (1984).
On wind mixing of a stratified shear flow.
Norwegian Hydrodynamic Laboratories, Trondheim, Norway.

Engqvist, A., and Omstedt, A. (1992)
Water exchange and density structure in a multi-basin estuary.
Continental Shelf Research, Vol. 12, No. 9, 1003 - 1026.

Gustafsson,N., Nyberg, L. and Omstedt, A. (1998)
Coupling high resolution atmosphere and ocean models for the Baltic Sea. Montly weather Review. In press.

Larsson, R., and Svensson, U. (1980).
A one-dimensional numerical model study of some basic features of the flow in ice-covered lakes.
J. of Hydraulic Res., 18, No. 3, pp 251-267.

Ljungemyr, P., Gustafsson, N., and Omstedt, A. (1996)
Parameterization of lake thermodynamics in a high resolution weather forecasting model.
Tellus, 48 A, No. 5, 608-621

Omstedt, A., and Svensson, U. (1992)
On the melt rate of drifting ice heated from below.
Cold Regions Science and Technology, 21, 91 - 100.

Omstedt, A., and Wettlaufer, J.S. (1992)
Ice growth and oceanic heat flux: Models and measurements.
J. Geoph. Res., Vol. 97, No. C 6, 9383 - 9390.

Rahm, L-A., and Svensson, U. (1986).
Dispersion of marked fluid elements in a turbulent Ekman layer.
J. Phys. Oceanogr., 16, 2084-2096.

Rahm, L-A., and Svensson, U. (1989).
Dispersion in a stratified benthic boundary layer.
Tellus, 41A, 148-161.
Rahm, L-A., and Svensson, U. (1989).
On the mass transfer properties of the benthic boundary layer with an application to oxygen fluxes.
Netherlands Journal of Sea Research, 24 (1): 27-35.

Rahm, L-A., and Svensson, U. (1993).
Note on dispersion in an ocean surface Ekman layer due to variable wind forcing.
Dt. Hydrogr. Z. 45.

Roos, J. (1996).
The Finnish Research programme on climate change.
Publications of the Academy of Finland 4/96.
"In this report a number of lake applications can be found".

Sahlberg, J. (1984).
A hydrodynamical model for heat contents calculations on lakes at the ice formation date.
Swedish Council for Building Research, Sweden. Document D4:1984.

Sahlberg, J. (1983).
A hydrodynamical model for calculating the vertical temperature profile in lakes during cooling.
Nordic Hydrology, Vol. 14, No. 4, pp 239-254.

Spalding, D.B., and Svensson, U. (1977).
The development and erosion of the thermocline .
In: Heat transfer and turbulent Buoyant convection, studies and applications for natural environment buildings, engineering systems.
Spalding, D.B., and Afgan, N., eds., Hemisphere Publishing Corp., Washington, D.C., USA.

Svensson, U. (1978).
A mathematical model of the seasonal thermocline.
Dept. Of Water Resources Eng., Univ. Of Lund, Sweden, Report No. 1002.

Svensson, U. (1978).
Examination ●f the summer stratification.
Nordic Hy●rology, 9, pp 105-120.

Svensson, U. (1979).
The structure of the Ekman layer.
Tellus, 31, pp 340-350.

Svensson, U. (1980).
On the numerical prediction of vertical turbulent exchange in stratifie● flows.
Second IAHR Symposium on Stratified Flows, Trondheim, Norway, June, 19●0.

Svensson, U. (1981).
On the influence of buoyancy on the turbulent Ekman layer.
Proc. Third Symposium on Turbulent Shear Flows, Univ. Of California, Davis.

Svensson, U. (1982).
Modelling the turbulence structure ●f the adiabatic atmospheric boundary layer.
Water Resources Eng., Univ. Of Luleå, Swe●en, Report TULEA.

Svensson, U. (1984).
PROBE – A c●mputer code for lake water quality modelling.
Presented at N●rdic Hydr●logical Conference, Nyborg, Denmark.

Svensson, U. (1985).
Applications of a two-equation turbulence mo●el to geophysical boundary layers.
Presented at IUTAM Symposium on Mixing in Stratified Fluids, Margaret River, Western Australia.

Svensson, U., and Omstedt, A. (1990).
A mathematical model of the boundary layer under drifting melting ice.
J●urnal of Physical Oceanography, Vol. 20, No. 2, pp 161-171.

Svensson, U., and Omstedt, A. (1994)
Simulation of supercooling and size distribution in frazil ice dynamics.
Cold Regions Science and Tech., 22, 221-233.

Svensson, U. and Omstedt, A.(1998)
Numerical simulation ●f frazil ice dynamics in the upper layer of the ocean. Cold Regions Sciences and Tech., in press.

Svensson, U., and Rahm, L. (1988).
Modeling the near-bottom region of the benthic boundary layer.
Journal of Geophysical Research, Vol. 93, No. C6, pp 6909-6●15.

Svensson, U., Sahlberg J. (1989).
Formulae for pressure gradients in one-dimensional lake models.
Journal of Geophysical Research, Vol. ●4, No. C4, pp 4●39-4946.

9. ACKNOWLEDGEMENTS

The basic philosophy, structure and features of PROBE have much in common with the two fluid dynamic codes GENMIX and PHOENICS, developed by Spalding (1977, 1983). There are two basic reasons for this. Firstly, these codes are very well structured and written and therefore well suited to be good examples worth following. Secondly, PHOENICS is in use at SMHI, and it is desirably for users of both codes that the general features as well as variable names are the same in both codes.

Anders Omstedt, Jörgen Sahlberg and Ola Nordblom at SMHI, have made major contributions in the development of PROBE. Their assistance in developing, testing and refining of the program is acknowledged with thanks.

APPENDIX A

MATHEMATICAL FORMULATION

1. Basic assumptions

Most assumptions are related to the one-dimensional treatment of the situations considered. All gradients in the horizontal directions are then neglected. The effect of a horizontal distribution of heat and momentum flux at a lake surface is thus not possible to include.

It will further be assumed that turbulent mixing processes can be described by turbulent exchange coefficients. This description is based on Reynold's averaging of Navier-Stoke's equations, which accordingly is assumed to be valid. The introduction of exchange coefficients and gradient laws exclude the proper treatment of counter gradient fluxes. Internal absorption of short wave radiation is assumed to follow an exponential decay law. Gravitational effects are assumed to obey the Boussinesq approximation, and the effect of the rotation of the earth is described by the Coriolis′ parameter.

In PROBE vertical advection due to in- and outflows at different levels in a reservoir is accounted for. However, since the treatment is not general (for example, advective momentum transport across boundaries is not allowed), the advective term will not be included in the general treatment of the equations but considered as a source/sink term in the special case mentioned above.

In the 1997 version of PROBE an option for two-dimensional steady parabolic flows is introduced. In the presentation below the set of equations for this option can be obtained by replacing the time derivative $(\partial\phi/\partial t)$ with an advective term $(\partial\phi u/\partial x)$. A full account of the two-dimensional option is given in Nordblom (1997).

2. Momentum equations

Within the assumptions made, the momentum equations read:

$$\frac{\partial \rho U}{\partial t} = -\frac{\partial p}{\partial x} + \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho}\frac{\partial \rho U}{\partial z}\right) + f\rho v \tag{A 1}$$

$$\frac{\partial \rho V}{\partial t} = -\frac{\partial p}{\partial y} + \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho}\frac{\partial \rho V}{\partial z}\right) - f\rho U \tag{A 2}$$

where t is time coordinate, x and y horizontal space coordinates, z vertical space coordinate, U and V horizontal velocities, p pressure, f Coriolis' parameter, and ρ density. The dynamical effective viscosity, μ_{eff}, is the sum of the turbulent viscosity, μ_T, and the laminar viscosity, μ. Pressure gradients may be treated in several ways, depending on the problem considered.

a) Prescribed
b) Calculated with respect to a prescribed total mass flux. The formula employed is iterative of the following type:

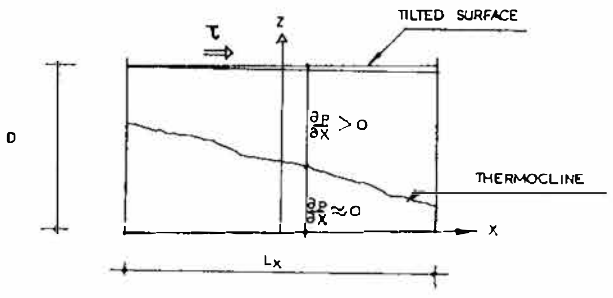
$$\frac{\partial p^{i+1}}{\partial x} = \frac{\partial p^{i}}{\partial x} + PFILT * \left(\overline{\rho u} - \overline{\rho u_p}\right) \qquad \text{(A 3)}$$

where i is iteration step, $PFILT$ a constant, $\overline{\rho u}$ total mass flux and $\overline{\rho u_p}$ prescribed total mass flux. The formula produces a pressure gradient, which in the steady state gives $\overline{\rho u}$ equal to $\overline{\rho u_p}$. From the formula it can be understood that the value of $PFILT$ will not affect the converged solution.
c) Pressure formula for lakes and reservoirs. In Svensson (1978) (see also Svensson and Sahlberg (1989)) pressure formulas for lakes and reservoirs were derived, which simulate the effect of the limited horizontal extent of a water body:

$$\frac{\partial}{\partial t}\left(\frac{\partial p}{\partial x}\right) = \rho g \frac{\pi^2 \bar{u} \times D}{L_x^2} \qquad \text{(A 4)}$$

$$\frac{\partial}{\partial t}\left(\frac{\partial p}{\partial y}\right) = \rho g \frac{\pi^2 \bar{v} \times D}{L_y^2} \qquad \text{(A 5)}$$

where g is the acceleration due to gravity, D depth, $\bar{u}$ and $\bar{v}$ mean velocities, $\pi = 3.1416$, and L_x and L_y horizontal dimensions of the water body.



TILTED SURFACE
τ
z
D
$\frac{\partial p}{\partial x} > 0$
THERMOCLINE
$\frac{\partial p}{\partial x} \approx 0$
x
L_x

Figure A.1. *Illustration of stratification effects on the pressure gradient.*

It is however, necessary to include the effect of stratification on the pressure gradients, as illustrated in Figure A.1. The tilted thermocline shown has been observed both in lakes and in the laboratory. Realising that the effect of the tilt is to eliminate pressure gradients below the interface, one may formulate the following expressions:

$$\frac{\partial p^{i+1}}{\partial x} = \left[\frac{\partial p^i}{\partial x} + \frac{\Delta t \times \rho g \pi^2 \times \bar{u} D}{L_x^2} \right] \times \frac{T - T_{bottom}}{T_{surface} - T_{bottom}} \qquad \text{(A 6)}$$

$$\frac{\partial p^{i+1}}{\partial y} = \left[\frac{\partial p^i}{\partial y} + \frac{\Delta t \times \rho g \pi^2 \times \bar{v} D}{L_y^2} \right] \times \frac{T - T_{bottom}}{T_{surface} - T_{bottom}} \qquad \text{(A 7)}$$

where i is time level and Δt time step. It is thus formulae (A 4) and (A 5) with the time derivative expressed as a finite difference, that are the basic equations. From the formulae it is seen that the effects of stratification will be that pressure gradients are zero close to the bottom, since T then equals T_{bottom}, and that they will be unaffected close to the surface, T then equals $T_{surface}$. These implications are qualitatively correct. The formulae do, however, not contain any mechanism for the generation or description of internal oscillations. It should be mentioned that the formulae A 4 – A 7 are tentative and have not yet been fully tested.

3. Heat energy equation

$$\frac{\partial}{\partial t}(\rho c_p T) = \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho \sigma_{eff}} \frac{\partial}{\partial z}(\rho c_p T) \right) + R(1-\eta)e^{-\beta(D-z)} \qquad \text{(A 8)}$$

where

$$\frac{\mu_{eff}}{\sigma_{eff}} = \frac{\mu}{\sigma} + \frac{\mu_T}{\sigma_T} \qquad \text{(A 9)}$$

temperature is denoted by T, c_p is specific heat, R incoming short wave radiation, η fraction of R absorbed at surface, β extinction coefficient, and σ_{eff}, σ_T, and σ effective, turbulent and laminar Prandtl numbers respectively.

4. Salinity and concentration equations

These equations can be expressed in the general form:

$$\frac{\partial \phi}{\partial t} = \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho \sigma_{eff}} \right) \frac{\partial \phi}{\partial z} + S_\phi \qquad \text{(A 10)}$$

where ϕ stands for salinity, s, or one of the concentrations c1, c2, c3 or c4. No source terms are provided for these variables. The user thus has to supply these explicitly, when it has been established what source and sink terms the concentration equation considered has.

5. Turbulence model

PR●BE embodies a two equation turbulence model, the $k-\varepsilon$ model. A detailed description of the derivation and application of this model is given by Rodi (1980). The dynamical eddy viscosity is calculated from the turbulent kinetic energy, k, and its dissipation rate, ε, by the Prandtl/Kolmogorov relation:

$$\mu_T = C_\mu \rho \frac{k^2}{\varepsilon} \tag{A 11}$$

where C_μ is an empirical constant. The equations for k and ε can be derived from the Navier-Stokes equations and thereafter modelled to the following form:

Turbulent kinetic energy:

$$\frac{\partial k}{\partial t} = \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho\sigma_k}\frac{\partial k}{\partial z}\right) + \frac{\mu_T}{\rho}\left[\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2\right] + P_b - \varepsilon \tag{A 12}$$

where

$$P_b = \frac{\mu_T}{\rho}\left(-\frac{g2\alpha(T-T_0)}{\sigma_T}\frac{\partial T}{\partial z} + \frac{g\alpha_S}{\sigma_{TS}}\frac{\partial S}{\partial z} + \frac{g\alpha_{C1}}{\sigma_{TC1}}\frac{\partial_{C1}}{\partial z} \ldots + \frac{g\alpha_{C4}}{\sigma_{TC4}}\frac{\partial C_4}{\partial z}\right) \tag{A13}$$

Dissipation of turbulent kinetic energy:

$$\frac{\partial \varepsilon}{\partial t} = \frac{\partial}{\partial z}\left(\frac{\mu_{eff}}{\rho\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial z}\right) + C_{1_\varepsilon}\frac{\varepsilon}{k}\left[\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2\right] + C_{3_\varepsilon}\frac{\varepsilon}{k}P_b - C_{2_\varepsilon}\frac{\varepsilon^2}{k} \tag{A 14}$$

where P_b is the production due to buoyancy, which includes contributions from heat energy, salinity and the four concentration equations. The turbulent Prandtl/Schmidt numbers and coefficients of expansion will then enter the expression. Further details can be found in Rodi (1980).

6. Turbulent Prandtl/Schmidt numbers

Two options are available for the turbulent Prandtl/Schmidt numbers. The numbers can be given constant values or be calculated from the following formula.

$$\sigma = \frac{\phi}{\phi_T}\frac{1+\phi_T'(C_T'-\phi_T)\times B}{1+B\sigma\sigma_T} \tag{A 15}$$

where

$$B = g\frac{k^2}{\varepsilon^2}\left[2\alpha(T - T_0)\frac{\partial T}{\partial z} - \alpha_S\frac{\partial S}{\partial z} - \alpha_{C_1}\frac{\partial C_1}{\partial z}...\alpha_{C4}\frac{\partial C_4}{\partial z}\right] \qquad \text{(A 16)}$$

is a buoyancy parameter. This formula was originally suggested by Launder (1975), where details about the derivation and numerical values of constants can be found, for the case of only one buoyancy affecting variable. Above it is extended to include several variables by simply adding the effects. It should be noted that this procedure has not been verified by a detailed derivation similar to the one done by Launder.

7. Boundary conditions

For momentum, heat energy, salinity, and concentrations, boundary conditions can be applied in two different ways; either the flux of the variable or the value of the variable at the boundary is given. A shear stress at a water surface, for example, is a "flux condition", while the zero velocity at a bottom is a "value condition".

The boundary conditions for k and ε are somewhat different. When a shear stress or a turbulence producing buoyancy flux is present at a boundary, k and ε are specified close to the boundary in relation to these fluxes. Details can be found in Svensson (1978) and Rodi (1980). If no shear or bouyancy flux is present, k and ε are treated as if the boundary was a symmetry plane, i.e. a zero gradient condition is assumed.

8. Equation of state

The equation of state assumes a quadratic relationship between temperature and density and linear relationship for salinity and concentration, thus:

$$\rho = \rho_0\left(1 - \alpha_1(T - T_r)^2 + \alpha_2 S + \alpha_3 C_1 + \alpha_4 C_2 + \alpha_5 C_3 + \alpha_6 C_4\right) \qquad \text{(A 17)}$$

where ρ_0 is a reference density, T_r the temperature of maximum density and $\alpha_1 - \alpha_6$ coefficients. In order to obtain maximum accuracy it may be needed to tune T_r and the coefficients. It is, for example, necessary to choose T_r with respect to the salinity interval under consideration.

APPENDIX B

THE FINITE DIFFERENCE EQUATIONS FOR THE ONE-DIMENSIONAL TRANSIENT OPTION.

1. Introduction

There are several ways of deriving the finite different form of differential equations. In this appendix they will be derived by integrating the differential equations over control volumes. The general outline of the technique follows from Spalding (1976) or Patankar (1980).

2 The grid arrangement and the general differential equation

All the differential equations given in Appendix A may be presented in the general form:

$$\frac{\partial \phi}{\partial t} = \frac{\partial}{\partial z}\left(\Gamma_\phi \frac{\partial \phi}{\partial z}\right) + S_\phi \qquad \text{(B 1)}$$

where ϕ stands for ρu, when x-direction momentum is considered, ρcT when heat energy is considered, etc. The source term for the variable ϕ is denoted by S_ϕ and Γ_ϕ is a transport coefficient defined by:

$$\Gamma_\phi = \frac{\mu_{eff}}{\rho \sigma_{eff,\phi}} \qquad \text{(B 2)}$$

where $\sigma_{eff,\phi}$ is the effective Prandtl/Schmidt number for the variable ϕ.

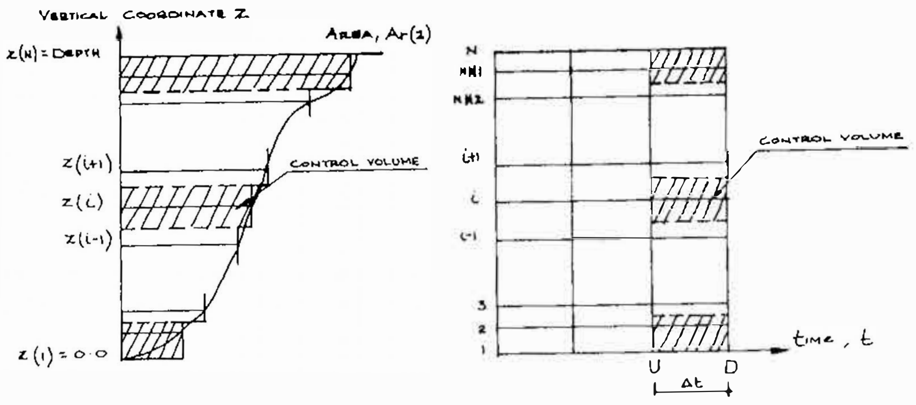


Figure 1. *Illustration of grid and control volumes.*

This general equation is to be integrated over the control volume with index i, see Figure 1. From this figure it can also be seen that the vertical variation of horizontal area, $Ar(z)$, will be considered. This variation will be taken as stepwise, as illustrated.

Time will be denoted by *t*, and when considering a control volume, *U* stands for up and *D* for down, along the time axis. *N* is the number of grid lines in the vertical direction, and *NM*1 means *N*-1, *NM*2 means *N*-2, etc.

2. Integration over a control volume

Equation (B 1) is to be integrated over horizontal area, vertical distance and time. This will be done for the general control volume *i*. Thus:

$$\int_0^{Ar(i)} \int_{z\left(i-\frac{1}{2}\right)}^{z\left(i+\frac{1}{2}\right)} \int_U^D \left[\underset{(a)}{\frac{\partial\phi}{\partial t}} = \underset{(b)}{\frac{\partial}{\partial z}\left(\Gamma_\phi \frac{\partial\phi}{\partial z}\right)} + \underset{(c)}{S_\phi} \right] dt\ dz\ dAr \qquad \text{(B 3)}$$

Integrate this equation term by term.

$$(a) \int_0^{Ar(i)} \int_{z(i-½)}^{z(i+½)} \int_U^D \frac{\partial\phi}{\partial t} dt\ dz\ dAr = \Delta z(i) Ar(i) (\phi_D(i) - \phi_U(i))$$

(B4)

$$(b) \int_0^{Ar(i)} \int_U^D \int_{z(i-½)}^{z(i+½)} \left(\frac{\partial}{\partial z}\left(\Gamma_\phi \frac{\partial\phi}{\partial z}\right)\right) dz\ dt\ dAr =$$

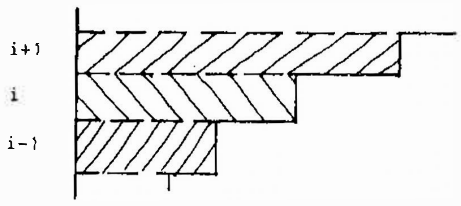


Figure 2. Detail of the control volumes.

$$= \int_0^{Ar(i)} \int_U^D \left[\left(\Gamma_\phi \frac{\partial\phi}{\partial z}\right)_{i+½} - \left(\Gamma_\phi \frac{\partial\phi}{\partial z}\right)_{i-½} \right] dt\ dAr =$$

(B 5)

$$= \Delta t \left[\left(Ar\Gamma_\phi \frac{\partial\phi}{\partial z}\right)_{i+½,t^*} - \left(Ar\Gamma_\phi \frac{\partial\phi}{\partial z}\right)_{i-½,t^*} \right]$$

where t^* is some time between *U* and *D*. To increase the numerical stability of the scheme, time level *D* will be used for t^* whenever possible. With this choice the numerical solution technique is of the fully implicit kind.

In the above expression $Ar(i+½)$ and $Ar(i-½)$ are used. The stepwise specification of Ar is , however, discontinuous at these locations, and the question then arises, which Ar should be used. To settle this, one has to look into the physical significance of (B 5), see Figure 2.

The term $\left(Ar\Gamma_\phi \frac{\partial\phi}{\partial z}\right)_{i-½}$ represents a loss (assume $\frac{\partial\phi}{\partial z} > o$) for control volume i at the lower boundary due to diffusive transport. It also represents a gain for control volume i-1 at the upper boundary. If there is no loss associated with the bottom contact, we will require that all the flux leaving control volume i shall enter control volume i-1. An example of such a variable is heat energy, as it is well known that only a negligible part of the vertical heat flux will be stored in the bottom sediments. The correct area at $i-½$ is thus $Ar(i-1)$, and with the same arguments $Ar(i)$ will be the appropriate area $i+½$. This area specification should be used for all variables, which exhibit this "conservative" nature in contact with the bottom . If, on the other hand, the variable in question experiences losses in contact with the bottom, it is clear from Figure 2 that the flux leaving control volume $i-½$ is not the same as entering control volume $i-1$ at the upper boundary. Momentum is an example of such a "non-conservative" variable. This because of the losses at the bottom due to friction. For all "non-conservative" variables the most reasonable choice is $Ar(i)$ for both $i+½$ and $i-½$ when studying control volume i. This is the area specification normally used for all hydrodynamical variables, while the heat energy, salinity, and concentrations will normally be treated as "conservative".

These conclusions will now be introduced into (B 5) through the definitions:

$$T_+ = Ar(i)\Gamma_\phi(i+½)/\Delta z(i+½) \tag{B 6}$$

$$T_- = \begin{cases} Ar(i)\Gamma_\phi(i-½)/\Delta z(i-½); \text{if } \phi \text{ is not "conservative"} \\ Ar(i-1)\Gamma_\phi(i-½)/\Delta z(i-½); \text{if } \phi \text{ is "conservative"} \end{cases} \tag{B 7}$$

With these expressions one may write (B 5) as.

$$\Delta t[T_+(\phi_D(i+1)-\phi_D(i))-T_-(\phi_D(i)-\phi_D(i-1))] \tag{B 8}$$

$$(c)\int_{O}^{Ar(i)}\int_{z(i-½)}^{z(i+½)}\int_{U}^{D} S_\phi dt\ dz\ dAr = Ar(i)\Delta z(i) S_{\phi,t*}\Delta t \tag{B 9}$$

The source term will be divided into two parts, one of which contains the variable itself. Thus:

$$S_{\phi,t*} = S(i)+S'(i)\phi_D \tag{B 10}$$

With this definition (B 9) becomes:

$$Ar(i)\Delta z(i)\Delta t(S(i)+S'(i)\phi_D) \qquad \text{(B 11)}$$

Collect terms (B 4), (B 8), and (B 11) and obtain:

$$\begin{aligned}&\Delta z(i)Ar(i)(\phi_D(i)-\phi_U(i))\\&=\Delta t[T_+(\phi_D(i+1)-\phi_D(i))-T_-(\phi_D(i)-\phi_D(i-1))]\\&+Ar(i)\Delta z(i)\Delta t[S(i)+S'(i)\phi_D]\end{aligned} \qquad \text{(B 12)}$$

Which may be rearranged to:

$$\begin{aligned}&\phi_D(i)[Ar(i)\Delta z(i)+\Delta t(T_+ + T_-)-Ar(i)\Delta z(i)\Delta tS'(i)]\\&+\phi_D(i+1)(-\Delta tT_+)+\phi_D(i-1)(-\Delta tT_-)+\phi_U(i)(-Ar(i)\Delta z(i))\\&-Ar(i)\Delta z(i)\Delta tS(i)=0\end{aligned} \qquad \text{(B 13)}$$

or

$$D(i)\phi_D(i)=A(i)\phi_D(i+1)+B(i)\phi_D(i-1)+C(i) \qquad \text{(B 14)}$$

where:

$$A(i)=T_+/Ar(i) \qquad \text{(B 15)}$$

$$B(i)=T_-/Ar(i) \qquad \text{(B 16)}$$

$$C(i)=\phi_U(i)\Delta z(i)/\Delta t+\Delta z(i)S(i) \qquad \text{(B 17)}$$

$$\begin{aligned}D(i)&=\Delta z(i)/\Delta t+(T_+ + T_-)/Ar(i)-\Delta z(i)S'(i)=\\&=A(i)+B(i)+\Delta z(i)/\Delta t-\Delta z(i)S'(i)\end{aligned} \qquad \text{(B 18)}$$

Equation (B 14) is in a form, which is easily solved using a tri-diagonal matrix algorithm. For a presentation of such an algorithm see for example Spalding (1976).

3. Coefficients for control volumes at the boundaries

Background

Close to the boundaries the transport coefficients often vary steeply. Special attention must therefore be paid to the coefficients in these regions. In PROBE the coefficients are calculated with special wall functions, which are based on logarithmic and linear laws.

In this section it will be shown how the coefficients are incorporated into the finite difference formulation. Two different cases may be distinguished, depending on if the value or the flux of ϕ is prescribed.

The value of ϕ is prescribed

For this boundary condition one only has to introduce the new boundary coefficients:

$$B(2) = TB / Ar(2) \tag{B 19}$$

$$A(N) = TS / Ar(NM1) \tag{B 20}$$

Where the *TB* and *TS* are transport coefficients at the bottom and surface respectively.

The flux of ϕ is prescribed

For the surface:

$$Ar(NM1)\gamma_\phi = TS(\phi_D(NM1) - \phi_D(N)) \tag{B 21}$$

where γ_ϕ is flux of ϕ per unit area and time.
From (B21):

$$\phi_D(N) = \gamma_\phi Ar(NM1)/TS + \phi_D(NM1) \tag{B 22}$$

Substitute from this for ϕ_N in equation (B 14) with $i = NM1$

$$\begin{gathered} D(NM1)\phi_D(NM1) = TS / Ar(NM1) \\ [-\gamma_\phi Ar(NM1)/TS + \phi_D(NM1)] + B(NM1)\phi_D(NM2) + C(NM1) \end{gathered} \tag{B 23}$$

which may be written as:

$$\begin{gathered} D'(NM1)\phi_D(NM1) = A'(NM1)\phi_D(N) + B(NM1) \\ \phi(NM2) + C'(NM1); \end{gathered} \tag{B 24}$$

where

$$D'(NM1) = D(NM1) - TS / Ar(NM1) \tag{B 25}$$

$$C'(NM1) = C(NM1) - \gamma_\phi \tag{B 26}$$

$$A'(NM1) = 0.0 \tag{B 27}$$

This is the set of coefficients to be used when the flux of ϕ at the surface is prescribed. The expressions for the bottom boundary are analogue.

APPENDIX C

THE FINITE DIFFERENCE EQUATIONS FOR THE TWO-DIMENSIONAL STEADY OPTION.
(From Nordblom (1997)).

Before deriving the finite-difference equations, one has to decide the order in which the equations are solved. It is assumed here that the first equation solved at each new integration step is the horizontal momentum equation. Thereafter, the vertical velocity component is calculated from the continuity equation. Then, the heat equation, the turbulent kinetic energy equation and the dissipation rate equation are solved, one after the other. Thus, after the horizontal momentum equation has been solved and the vertical velocity component has been obtained from the continuity equation, the velocity field can be regarded as known when the remaining equation are solved. This fact will be referred to below.

While the numerical scheme used in PROBE for the one-dimensional transient case can be characterized as fully implicit, the finite-difference equations are here derived for the general case where the level between the fully explicit and the fully implicit scheme is expressed by a weighting factor. It is then easy to select a specific scheme, e.g. of the Crank-Nicholson type or of the fully-implicit type, simply by adjusting the weighting factor.

The starting point in the derivation is the general differential equation for two-dimensional parabolic steady flows, here written with all terms on the left

$$\frac{\partial}{\partial x}(u\phi)+\frac{\partial}{\partial z}(w\phi)-\frac{\partial}{\partial z}\left(\Gamma\frac{\partial\phi}{\partial z}\right)-S=0 \qquad \text{(C 1)}$$

In this equation, when $\phi = \rho u$, we get the horizontal momentum equation and when $\phi = \theta, \phi = \mathrm{k}$ and $\phi = \varepsilon$, we get the heat equation, the turbulent kinetic energy equation and the dissipation rate equation, respectively. S denotes the source term and Γ denotes the vertical exchange coefficient, corresponding to the variable ϕ.

In the Cartesian coordinate system used here, the horizontal axis is denoted by x and the vertical axis by z. The calculation domain is divided into a rectangular mesh and a part of this is shown in Figure 1 below. The horizontal distance between the grid cells Δx is assumed to be constant while the vertical distance Δz can vary.

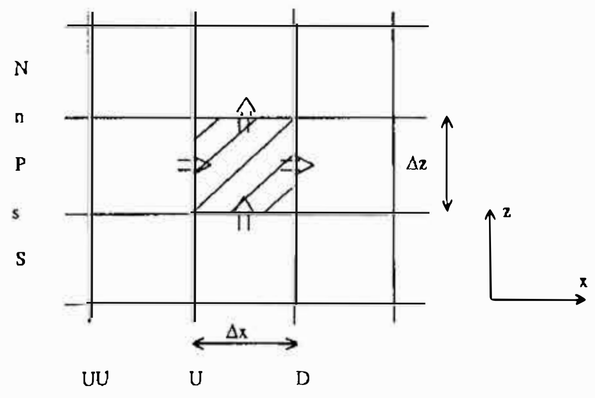


Figure 1. *A part of the finite-difference mesh.*

Equation (C 1) is to be integrated over the dashed grid cell shown in Figure 1. The direction of flow is assumed to be from left to right. With reference to Figure 1, the letters U and D stands for Up and Down and are the limits of integration in the horizontal direction, UU denotes the x-coordinate one integration step upstream of $x = U$, the lower case letters s and n refers to the z-coordinate of the lower and upper boundaries of the grid cell, respectively, and are the limits of integration in the vertical direction, P stands for the z-coordinate at the center of the dashed grid cell, while S and N refers to the z-coordinate at the center of the adjacent grid cells below (South) and above (North) of the grid cell considered. The arrows indicate the actual location of the points where the velocities are calculated (the vertical velocity is here arbitrary directed upward). The different terms in the differential equation are divided into three groups which are integrated separately. Group I is the horizontal convection term, group II is the vertical convection and diffusion terms (handled together) and group III is the source term.

Group I: The horizontal convection term.

When performing the integration over the vertical extent of the grid cell, it is assumed that $u\phi$ is constant with z and equal to the center point value $(u\phi)_P$. With this assumption, we get

$$\int_z\int_x \frac{\partial}{\partial x}(u\phi)\; dxdz = \int_z (u\phi)_D - (u\phi)_U \, dz = \Delta z\; \left[u_{P,D}\phi_{P,D} - u_{P,U}\phi_{P,U}\right], \text{(C 2)}$$

where the coefficients $u_{P,D}$ and $\mathrm{u}_{\mathrm{P,U}}$ denote the horizontal wind speeds in the cell walls at $\mathrm{x} = \mathrm{D}$ and $x = U$, respectively.

The coefficients $u_{P,D}$ and $\mathrm{u}_{\mathrm{P,U}}$ will be determined in different ways depending on if $\phi = \rho u$ or not. If ϕ is any of the variables θ, k or ε, both $u_{P,D}$ and $\mathrm{u}_{\mathrm{P,U}}$ can be regarded as known since the horizontal wind speed is determined from the horizontal momentum equation before the other variables are solved. On the other hand, when $\phi = \rho u$, $\mathrm{u}_{\mathrm{P,D}}$ is unknown and must be approximated.

One way of approximating $\mathrm{u}_{\mathrm{P,D}}$ is to set $u_{P,D}$ equal to $u_{P,U}$ where $u_{P,U}$ is known from the previous integration step. The error term following from this approximation can be determined by setting $u_{P,D} = u_{P,U} + \Delta u$, where $\Delta\mathrm{u}$ is the change in horizontal wind speed between $x = U$ and $x = D$. Inserting the relation $u_{P,D} = u_{P,U} + \Delta u$ in expression (C 2), we get

$$\Delta \mathrm{z}\left[\mathrm{u}_{\mathrm{P,U}}\phi_{\mathrm{P,D}} - \mathrm{u}_{\mathrm{P,U}}\phi_{\mathrm{P,U}} + \Delta \mathrm{u}\phi_{\mathrm{P,D}}\right]$$

Thus, the error introduced by replacing $u_{P,D}$ by $u_{P,U}$ is $\Delta z \Delta u \phi_{P,D}$ which is equal to $\Delta z \Delta u \rho u_{P,D}$ (since $\phi = \rho u$ when the momentum equation is solved).

A smaller error term can, however, be achieved if both $u_{P,D}$ and $u_{P,U}$ are replaced by the upstream values $u_{P,U}$ and $u_{P,UU}$, respectively. To see this, we write $u_{P,D}$ and $u_{P,U}$ in terms of the upstream values and the change over the horizontal grid distance Δx, according to

$$u_{P,U} = u_{P,UU} + \Delta \ u_1$$
$$u_{P,D} = u_{P,U} + \Delta \ u_2$$

If the horizontal grid distance is constant, the change in horizontal wind speed from $x = U$ to $x = D$ will be nearly the same as the change from $x = UU$ to $x = U$, i.e. $\Delta u_2 \approx \Delta u_1 = \Delta u$. Inserting the relations $u_{P,U} = u_{P,UU} + \Delta u$ and $u_{P,D} = u_{P,U} + \Delta u$ in expression $(C\,2)$ and recognizing that $\Delta u(\phi_{P,D} - \phi_{P,U}) = \rho\Delta u(u_{P,D} - u_{P,U}) = \rho(\Delta u)^2$, we get

$$\Delta z \ \left[u_{P,U}\phi_{P,D} - u_{P,UU}\phi_{P,U} + \rho(\Delta \ u)^2\right]$$

In this case, provided that $\Delta u_1 = \Delta u_2$, the resulting error term is $\Delta z\rho(\Delta \ u)^2$. Comparing the two error terms, it is seen that the error is reduced by the factor $\Delta u / u_{P,D}$ which is a significant improvement since the change in u over the grid distance Δx is only a small fraction of the absolute value of u, i.e. $\Delta u / u_{P,D} << 1$.

If the coefficients $u_{P,D}$ and $u_{P,U}$ in expression $(C\,2)$ are replaced by the symbols C_D and C_U, respectively, and the error term above is dropped, we get the following final expression for the integrated horizontal convection term

$$\Delta z \ \left[C_D\phi_{P,D} - C_U\phi_{P,U}\right], \qquad (C\,3)$$

where $C_D = u_{P,U}$, $C_U = u_{P,UU}$ if $\phi = \rho u$, and $C_D = u_{P,D}$, $C_U = u_{P,U}$, otherwise. (When $x = 0$ and $\phi = \rho u$, we must set $C_D = C_U = u_{P,U}$, where $u_{P,U}$ is the prescribed velocity at the upstream boundary.)

Group II: The horizontal convection and diffusion terms.

When performing the integration over the horizontal extent of the grid cell, it is assumed that all terms are constant with x and equal to a representative value at $x = x^* \in [U, D]$. With this assumption, we get

$$\int_x\int_z \frac{\partial}{\partial z}(w\phi) - \frac{\partial}{\partial z}\left(\Gamma\frac{\partial\phi}{\partial z}\right) dz dx = \int_x (w\phi)_n - \left(\Gamma\frac{\partial\phi}{\partial z}\right)_n - ((w\phi)_s - \left(\Gamma\frac{\partial\phi}{\partial z}\right)_s)dx =$$
$$= \Delta x\left[(w\phi)_{n,x^*} - \left(\Gamma\frac{\partial\phi}{\partial z}\right)_{n,x^*} - ((w\phi)_{s,x^*} - \left(\Gamma\frac{\partial\phi}{\partial z}\right)_{s,x^*})\right]$$

For convenience, the index x* is dropped here, but will be included later in the derivation. The expression then takes the following form

$$\Delta x\left[w_n\phi_n - \Gamma_n\left(\frac{\partial\phi}{\partial z}\right)_n - (w_s\phi_s - \Gamma_s\left(\frac{\partial\phi}{\partial z}\right)_s)\right] \qquad \text{(C 4)}$$

The value and the gradient of ϕ at the lower and the upper boundary of the grid cell are now to be expressed in terms of ϕ_S, ϕ_P and ϕ_N. This will, however, require knowledge of the variation of ϕ with z which is, of course, unknown since the variation of ϕ in the x- and z-direction is the outcome of the numerical solution. Instead, we must use approximate relations for w_n, $\left(\frac{\partial\phi}{\partial z}\right)_n$, w_s and $\left(\frac{\partial\phi}{\partial z}\right)_s$ expressed in the grid point values ϕ_S, ϕ_P and ϕ_N.

In Patankar (1980), several methods are discussed. The simplest approach to the problem is "The Upwind Scheme" and the somewhat more advanced methods are variants of "The Exponential Scheme". These schemes are presented below.

The Upwind Scheme:

In The Upwind Scheme, the value of ϕ at a cell wall is replaced by the upwind value and the gradient of ϕ is calculated from a central difference approximation. Using the FORTRAN operator $MAX[\ ]$ which returns the greater of its arguments, the convective terms $w_n\phi_n$ and $w_s\phi_s$ can be written in a compact form according to

$$w_n\phi_n = \phi_P \mathrm{MAX}[w_n,0] - \phi_N \mathrm{MAX}[-w_n,0]$$
$$w_s\phi_s = \phi_S \mathrm{MAX}[w_s,0] - \phi_P \mathrm{MAX}[-w_s,0]$$

The above expressions will always assign the upwind value to ϕ at a cell wall, regardless of the flow direction.

The diffusive flux at the upper and lower cell walls is calculated from a central difference approximation according to

$$\Gamma_n\left(\frac{\partial\phi}{\partial z}\right)_n = \frac{\Gamma_n}{z_N - z_P}(\phi_N - \phi_P)$$

$$\Gamma_s\left(\frac{\partial\phi}{\partial z}\right)_s = \frac{\Gamma_s}{z_P - z_S}(\phi_P - \phi_S)$$

Introducing the variables DIF_n for $\frac{\Gamma_n}{z_N - z_P}$ and DIF_s for $\frac{\Gamma_s}{z_P - z_S}$, expression (C 4) takes the following form

$$\Delta x[(MAX[w_n,0]+DIF_n+MAX[-w_s,0]+DIF_s)\phi_P -$$
$$(MAX[-w_n,0]+DIF_n)\phi_N-(MAX[w_s,0]+DIF_s)\phi_S]$$

Setting $A=MAX[-w_n,0]+DIF_n$ and $B=MAX[w_s,0]+DIF_s$, we get the final expression for the integrated convection and diffusion terms for The Upwind Scheme

$$\Delta x[(w_n-w_s+A+B)\phi_P-A\phi_N-B\phi_S] \qquad \text{(C 5)}$$

The Exponential Scheme and variants:

In The Exponential Scheme, an exact expression for the variation of ϕ with z is derived for an idealized convection-diffusion flow; a one-dimensional stationary flow without source terms and with constant density ρ and a constant diffusion coefficient Γ . The differential equation for this situation read

$$w\frac{\partial\phi}{\partial z}-\Gamma\frac{\partial^2\phi}{\partial z^2}=0 \text{ or } \frac{\partial^2\phi}{\partial z^2}-\frac{w}{\Gamma}\frac{\partial\phi}{\partial z}=0$$

Since this is a linear ordinary differential equation with constant coefficients, the equation is easily solved by analytical methods. (Note that not only Γ is constant here, w is also constant in a one-dimensional flow with constant density, from continuity reasons.) The solution in the interval $[z_P,z_N]$, subject to the boundary conditions $\phi(z_P)=\phi_P$ and $\phi(z_N)=\phi_N$ becomes

$$\phi(z)=\phi_P+(\phi_N-\phi_P)\frac{\exp(\frac{w}{\Gamma}(z-z_P))-1}{\exp(\frac{w}{\Gamma}(z_N-z_P))-1}, z\in[z_P,z_N] \qquad \text{(C 6)}$$

By differentiating this function, we get

$$\frac{\partial\phi}{\partial z}(z)=(\phi_N-\phi_P)\frac{w}{\Gamma}\frac{\exp(\frac{w}{\Gamma}(z-z_P))}{\exp(\frac{w}{\Gamma}(z_N-z_P))-1}, \; z\in[z_P,z_N] \qquad \text{(C 7)}$$

The functional relationships for the value and the gradient of ϕ in the interval $[z_S,z_P]$ will be analogous, all indices N are just replaced by P and P by S.

Since the actual flow is two-dimensional with a non-zero source term (in general) and a variable diffusion coefficient, we do not expect the analytical functions to be exact for the flow considered. From these functions we can, however, probably do the best assumption possible regarding the value and the gradient of ϕ at the cell walls.

Thus, we insert the functions (C 6) and (C 7) for a z-coordinate in the interval $[z_P, z_N]$ and the corresponding functions for a z-coordinate in the interval $[z_S, z_P]$, in expression (C 4). After some manipulations, we get

$$\Delta x\left[w_n(\phi_P + \frac{\phi_P - \phi_N}{\exp(P_n/\rho_n)-1}) - w_s(\phi_s + \frac{\phi_S - \phi_P}{\exp(P_s/\rho_s)-1})\right], \qquad \text{(C 8)}$$

where P_n and P_s are the Peclet numbers at the upper and lower walls of the grid cell, respectively. The Peclet numbers express the relative strength of convection and diffusion at the cell walls and are defined according to

$$P_n = \frac{\rho_n w_n (z_N - z_P)}{\Gamma_n} = \frac{\rho_n w_n}{DIF_n}$$

$$P_s = \frac{\rho_s w_s (z_P - z_S)}{\Gamma_s} = \frac{\rho_s w_s}{DIF_s},$$

where, as before, the variables DIF_n and DIF_s stand for $\frac{\Gamma_n}{z_N - z_P}$ and $\frac{\Gamma_s}{z_P - z_S}$, respectively.

By factoring out ϕ_P, ϕ_N and ϕ_S in expression (C 8), we get

$$\Delta x\left[(w_n + \frac{w_n}{\exp(P_n/\rho_n)-1} + \frac{w_s}{\exp(P_s/\rho_s)-1})\phi_P - \right.$$
$$\left. (\frac{w_n}{\exp(P_n/\rho_n)-1})\phi_N - (w_s + \frac{w_s}{\exp(P_s/\rho_s)-1})\phi_S \right]$$

The above expression can be simplified to the same expression as (C 5) by defining the coefficients A and B according to

$$A = \frac{w_n}{\exp(P_n/\rho_n)-1} = DIF_n \frac{P_n/\rho_n}{\exp(P_n/\rho_n)-1}$$

$$B = w_s + \frac{w_s}{\exp(P_s/\rho_s)-1} = DIF_s(P_s/\rho_s + \frac{P_s/\rho_s}{\exp(P_s/\rho_s)-1}),$$

With these definitions, the integrated convection and diffusion terms for The Exponential Scheme become

$$\Delta x[(w_n - w_s + A + B)\phi_P - A\phi_N - B\phi_S]$$

Following Patankar (1980), A and B will now be approximated by polynomial functions. There are two reasons for doing that; the polynomials are somewhat less expensive to compute than the exponentials, and they are well-defined and equal to the limit value of A and B at the point $P/\rho = 0$. From these approximate functions, we get "The Hybrid Scheme" and "The Power-law Scheme", (Patankar, 1980).

In The Hybrid Scheme, A and B are approximated by piecewise linear functions according to

$$A = DIF_n \cdot MAX\left[-P_n/\rho_n, 1 - \frac{P_n/\rho_n}{2}, 0\right] = MAX\left[-w_n, DIF_n - \frac{w_n}{2}, 0\right]$$

$$B = DIF_s \cdot MAX\left[P_s/\rho_s, 1 + \frac{P_s/\rho_s}{2}, 0\right] = MAX\left[w_s, DIF_s + \frac{w_s}{2}, 0\right]$$

In The Power-law Scheme, A and B are approximated by a 5th-degree polynomial for $P/\rho \in [-10,10]$ and linear functions outside this interval. The definitions read

$$\begin{aligned} A &= DIF_n \cdot \left(MAX\left[(1 - 0.1 \cdot |P_n/\rho_n|)^5, 0\right] + MAX\left[-P_n/\rho_n, 0\right]\right) = \\ &= MAX\left[DIF_n (1 - 0.1 \cdot |w_n/DIF_n|)^5, 0\right] + MAX\left[-w_n, 0\right] \end{aligned} \qquad \text{(C 9)}$$

$$\begin{aligned} B &= DIF_s \cdot \left(MAX\left[(1 - 0.1 \cdot |P_s/\rho_s|)^5, 0\right] + MAX\left[P_s/\rho_s, 0\right]\right) = \\ &= MAX\left[DIF_s (1 - 0.1 \cdot |w_s/DIF_s|)^5, 0\right] + MAX\left[w_s, 0\right] \end{aligned} \qquad \text{(C10)}$$

To sum up, it is recognized that the difference between the schemes presented, lies in the coefficients A and B. In The Upwind Scheme, A and B have the simplest form. The Exponential Scheme and its variants are somewhat more complicated, but are believed to perform better than The Upwind Scheme. As is pointed out in Patankar (1980), for high lateral flow (large values of the Peclet number), the gradient of ϕ will become very small, making the diffusive flux negligible. For this case, The Upwind Scheme has the drawback that it overestimates the diffusion since it always calculates the diffusion from a central difference approximation. In the other schemes where the coefficients A and B are functions of the Peclet number, the influence from diffusion at large values of the Peclet number is reduced automatically. It is true that all schemes will produce the same result when the grid distance is made small enough since a finer grid will also reduce the Peclet number. From a computational point of view, we should, however, choose a method that produces reasonable results also with a course grid. Thus, the scheme to be suggested here is The Hybrid Scheme or The Power-law Scheme. It is probably quite arbitrary which one is chosen. Following the recommendation in Patankar (1980), The Power-law Scheme will be used, with A and B from equation (C 9) and equation (C 10).

Now, introducing the index $x^* \in [U,D]$ that was dropped earlier, the expression (C 5) (valid for all schemes) become

$$\Delta x\left[(w_n - w_s + A + B)\phi_{P,x^*} - A\phi_{N,x^*} - B\phi_{S,x^*}\right] \qquad \text{(C 11)}$$

The value of ϕ at $x = x^*$, will now be expressed in terms of the old value from the previous integration step at $x = U$ and the new value from the current integration step at $x = D$ according to the linear relation $\phi_{x^*} = (1-f)\phi_U + f\phi_D$. When $f = 0$, we get the so called fully explicit scheme while $f = 0.5$ and $f = 1$ lead to the Crank-Nicholson scheme and the fully implicit scheme, respectively.

Inserting the relation, $\phi_{x^*} = (1-f)\phi_U + f\phi_D$ in expression (C 11), we get the following final expression for the integrated convection/diffusion term

$$\begin{aligned}&\Delta x[(w_n - w_s + A + B)[(1-f)\phi_{P,U} + f\phi_{P,D}] -\\ &A[(1-f)\phi_{N,U} + f\phi_{N,D}] - B[(1-f)\phi_{S,U} + f\phi_{S,D}]]\end{aligned} \qquad \text{(C 12)}$$

Group III: The source term.

When performing the integration over the vertical extent of the grid cell, it is assumed that the source term S is constant with z and equal to center point value S_P. In the integration over the horizontal extent of the grid cell, it is assumed that S_P is constant with x and equal to a representative value S_{P,x^*} at $x = x^* \in [U, D]$. Also, to prepare for situations where the source term is a function of ϕ, we use a linear expression for this dependence according to $S_P = SI + SIP\phi_{P,x^*}$, where SI and SIP are coefficients. With these assumptions, we get

$$\int_x \int_z S dz dx = \Delta z \int_x S_P dx = \Delta x \Delta z S_{P,x^*} = \Delta x \Delta z (SI + SIP\phi_{P,x^*})$$

Inserting the relation, $\phi_{x^*} = (1-f)\phi_U + f\phi_D$ in the expression above, we get

$$\Delta x \Delta z (SI + SIP[(1-f)\phi_{P,U} + f\phi_{P,D}]) \qquad \text{(C13)}$$

Now, adding together expression (C 3), (C 12) and (C 13), we arrive at the final finite-difference equation for two-dimensional parabolic flows. The equation read

$$\left[\frac{\Delta z}{\Delta x}C_D + (w_n - w_s + A + B)f - \Delta z SIPf\right]\phi_{P,D} = Af\phi_{N,D} + Bf\phi_{S,D} +$$
$$\left[\frac{\Delta z}{\Delta x}C_U\phi_{P,U} - (w_n - w_s + A + B)(1-f)\phi_{P,U} + A(1-f)\phi_{N,U} + \right.$$
$$\left. B(1-f)\phi_{S,U} + \Delta z(SI + SIP(1-f)\phi_{P,U})\right]$$

or

$$D'\phi_{P,D} = A'\phi_{N,D} + B'\phi_{S,D} + C', \qquad \text{(C 14)}$$

where

$$D' = \left[\frac{\Delta z}{\Delta x}C_D + (w_n - w_s + A + B)f - \Delta z SIPf\right], \; A' = Af \, , \; B' = Bf \text{ and}$$

$$C' = \left[\frac{\Delta z}{\Delta x}C_U\phi_{P,U} - (w_n - w_s + A + B)(1-f)\phi_{P,U} + A(1-f)\phi_{N,U} + \right.$$
$$\left. B(1-f)\phi_{S,U} + \Delta z(SI + SIP(1-f)\phi_{P,U})\right]$$

Calculation of the vertical velocity:

The vertical velocity at the cell boundaries is obtained from the continuity equation applied to each grid cell after the horizontal momentum equation (giving the horizontal velocities) has been solved. With reference to the dashed grid cell in Figure (1) and assuming constant density, the continuity equation gives

$$\Delta x(w_n - w_s) = \Delta z(u_{P,U} - u_{P,D})$$

Solving for the vertical velocity at the upper wall of the grid cell, w_n, we get

$$w_n = w_s + \frac{\Delta z}{\Delta x}(u_{P,U} - u_{P,D}) \qquad \text{(C 15)}$$

At a solid wall, the vertical velocity is known and equal to zero. Assuming a solid wall at the lower boundary, the vertical velocity at the upper wall of each grid cell in the finite-difference mesh can be determined by iterating equation. (C 15) through all the grid cells from bottom to top.

APPENDIX D

LISTING OF THE CODE

```
      PROGRAM PROBE97
C
C**********************************************************************
C     CODE NAME:     PROBE97
C     **********
C
C     PC-VERSION:
C     ***********
C
C     DEVELOPED BY:  URBAN SVENSSON
C     *************
C
C     DOCUMENTATION:
C     **************
C
C     COMMENTS:
C     *********
C
C**********************************************************************
C********************** MAIN PROGRAM **********************************
C
      INCLUDE 'comp97.inc'
C
      DIMENSION ISTORE(NSTORE,NPM)
C
C----------------------------------------------------------------------
CHAPTER 1  1  1  1  1  1  DATA 1  1  1  1  1  1  1  1  1  1  1  1
C

      CALL DFAULT
      CALL CASE(1)
      IF(NPROBE.EQ.1) GOTO 200
      CALL  STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
 100  IPROBE=IPROBE+1
      CALL CASE(1)
      CALL  STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      IF(IPROBE.LT.NPROBE) GOTO 100
      CALL STORE('R',1,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      IPROBE=1
C----------------------------------------------------------------------
CHAPTER 2  2  2  2  2  2  GRID AND GEOMETRY 2  2  2  2  2  2  2  2  2
C
 200  CONTINUE
C-----VERTICAL GRID DISTRIBUTION
C
      CALL GRID
C
C-----AREA VESUS DEPTH
C
      IF(INDARE.NE.4) CALL AREAD
C
C----------------------------------------------------------------------
CHAPTER 3  3  3  3  3  3  STARTING  VALUES  3  3  3  3  3  3  3  3
C
C-----INITIALISE DEPENDENT VARIABLES
      IF(ISTPR.NE.1) GOTO 300
      DO 32 J=1,NJM
      DO 33 I=2,NM1
      IF(Z(I).LE.ZST1(J)) F(I,J)=VST1(J)
      IF(Z(I).GT.ZST1(J))
     1 F(I,J)=VST1(J)+(Z(I)-ZST1(J))*(VST2(J)-VST1(J))
     2 /(ZST2(J)-ZST1(J)+TINY)
      IF(Z(I).GE.ZST2(J)) F(I,J)=VST2(J)
 33   CONTINUE
 32   CONTINUE
 300  CONTINUE
C-----INITIALISE OTHER VARIABLES
C
      DO 30 I=1,N
      RHO(I)=RHOREF*(1.-C1RHO*(F(I,JTE)-TREF)*(F(I,JTE)-TREF)+
     1C2RHO*F(I,JS)+C3RHO*F(I,JC1)+C4RHO*F(I,JC2)+
     2C5RHO*F(I,JC3)+C6RHO*F(I,JC4))
      F(I,JTE)=F(I,JH)/RHO(I)/CPHEAT
      EMU(I)=EMULAM
      IF(F(I,JK).LE.FKMIN.OR.F(I,JD).LE.FDMIN) THEN
      F(I,JK)=FKMIN
      F(I,JD)=FDMIN
      ENDIF
      IF(ITURBM.EQ.4) GOTO 31
      F(I,JEMU)=RHO(I)*CD*F(I,JK)*F(I,JK)/(F(I,JD)+TINY)+EMTMIN
      IF(ITURBM.EQ.1) F(I,JEMU)=EMUCON
 31   CONTINUE
30    CONTINUE
      IF(ITURBM.EQ.1.OR.ITURBM.EQ.4) IPRSC=1
C
      CALL OUTPUT
      CALL CASE(4)
C
      IF(NPROBE.EQ.1) GOTO 302
      IF(IPROBE.EQ.NPROBE) GOTO 301
      CALL STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      IPROBE=IPROBE+1
      CALL STORE('R',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      GOTO 200
 301  CONTINUE
      CALL STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      CALL STORE('R',1,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
      IPROBE=1
 302  CONTINUE
C---------------------------------------------------------------------
CHAPTER 4 4 4 4 4 4 STEP CONTROL 4 4 4 4 4 4 4 4 4 4
C
      ITIME1=1
      ITIME2=2
      NSTEP=INT(TFRAC(1))
      DT=TFRAC(2)
      NUMB=1
      ISTPDT=1
      DO 40 I=1,NPM
      IF(NSTPDT(I).EQ.1) THEN
      INDEXP=I
      GOTO 41
      ENDIF
 40   CONTINUE
 41   CONTINUE
C
400   CONTINUE
      IF(IPROBE.NE.INDEXP) GOTO 402
      IF(NUMB.LE.NSTEP) GOTO 401
      ITIME1=ITIME1+2
      ITIME2=ITIME2+2
      NUMB=1
      NSTEP=INT(TFRAC(ITIME1))
401   CONTINUE
      NUMB=NUMB+1
C
 402  CONTINUE
      DT=TFRAC(ITIME2)/NSTPDT(IPROBE)
      TIME=TU+DT
C-----------------------------------------------------------------------
CHAPTER 5  5  5  TIMEDEPENDENT  BOUNDARY CONDITIONS 5  5  5  5  5  5  5
C
      DO 50 J=1,NF
      IF(.NOT.SOLVAR(J)) GOTO 500
      IF(ITRHZ(J).NE.2) GOTO 501
      IF(TIME.LE.V4HZ(J)) VALUE=
     F V1HZ(J)+(V2HZ(J)-V1HZ(J))*TIME/V4HZ(J)
      IF(TIME.GT.V4HZ(J)) VALUE=
     F V2HZ(J)+V3HZ(J)*SIN(2.*PI*(TIME-V4HZ(J))/V5HZ(J))
      IF(IKBHZ(J).EQ.1) F(N,J)=VALUE
      IF(IKBHZ(J).EQ.2) FLUXHZ(J)=VALUE
C
501   IF(ITRLZ(J).NE.2) GOTO 502
      IF(TIME.LE.V4LZ(J)) VALUE=
     F V1LZ(J)+(V2LZ(J)-V1LZ(J))*TIME/V4LZ(J)
      IF(TIME.GT.V4LZ(J)) VALUE=
     F V2LZ(J)+V3LZ(J)*SIN(2.*PI*(TIME-V4LZ(J))/V5LZ(J))
      IF(IKBLZ(J).EQ.1) F(1,J)=VALUE
      IF(IKBLZ(J).EQ.2) FLUXLZ(J)=VALUE
502   CONTINUE
500   CONTINUE
50    CONTINUE
C
      CALL CASE(2)
C
C-----IN- AND OUTFLOWS
C   --CALCULATE VOLUME FLUX ALONG Z-AXIS
C
      TESTQ=ABS(QINFL(NM1))+ABS(QOUTFL(NM1))
      IF(TESTQ.GT.TINY.AND.MOVE) THEN
      WRITE(6,*)'WARNING IN- OR OUT-FLOW IN CELL NM1'
      ENDIF
      DO 51 I=2,NM1
      QZ(I)=QZ(I-1)+QINFL(I)-QOUTFL(I)
 51   CONTINUE
      QZ(N)=0.
      QSURF=QZ(NM1)-PREEVA*AREA(NM1)
      IF(MOVE) CALL SURF
      IF(ABS(QSURF).GT.TINY.AND..NOT.MOVE)THEN
      WRITE(6,*)'WARNING IN-AND OUTFLOW NOT IN BALANCE.
     1QSURF=',QSURF,'M3/S'
      ENDIF
C-----------------------------------------------------------------------
CHAPTER 6  6  6  6  6  6  ADVANCE 6  6  6  6  6  6  6  6  6  6  6  6  6
C
      CALL COMP
C
```

```
C-------------------------------------------------------------------
CHAPTER 7 7 7 7 7 7  COMPLETE 7 7 7 7 7 7 7 7 7 7 7 7 7
C
C-----PROPERTIES
     DO 70 I=1,N
     RHO(I)=RHOREF*(1.-C1RHO*(F(I,JTE)-TREF)*(F(I,JTE)-TREF)+
    1C2RHO*F(I,JS)+C3RHO*F(I,JC1)+C4RHO*F(I,JC2)+
    2C5RHO*F(I,JC3)+C6RHO*F(I,JC4))
     F(I,JTE)=F(I,JH)/RHO(I)/CPHEAT
     IF(ITURBM.EQ.1.OR.ITURBM.EQ.4) GOTO 71
     IF(F(I,JK).LE.FKMIN.OR.F(I,JD).LE.FDMIN)THEN
     F(I,JK)=FKMIN
     F(I,JD)=FDMIN
     ENDIF
     F(I,JEMU)=CD*RHO(I)*F(I,JK)*F(I,JK)/(F(I,JD)+TINY)+EMTMIN
 71  CONTINUE
 70  CONTINUE
C
C
     TU=TIME
     IF(ISTPDT.EQ.1)ISTEP=ISTEP+1
C-------------------------------------------------------------------
CHAPTER 8 8 8 8  PRINT 8 8 8 8 8 8 8 8 8 8
C
     IF(ISTPDT.EQ.NSTPDT(IPROBE))THEN
     CALL CASE(4)
     CALL OUTPUT
     ENDIF
C
C-------------------------------------------------------------------
CHAPTER 9 9 9 9 DECIDE 9 9 9 9 9 9 9 9 9 9
C
     IF(ISTEP.LT.LSTEP.AND.TU.LT.TLAST) GOTO 901
     IF(IPROBE.EQ.NPROBE.AND.ISTPDT.EQ.NSTPDT(IPROBE)) THEN
     CALL STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
     GOTO 900
     ENDIF
 901 CONTINUE
     IF(NPROBE.EQ.1) GOTO 902
     IF(ISTPDT.LT.NSTPDT(IPROBE)) THEN
     ISTPDT=ISTPDT+1
     GOTO 402
     ELSE
     ENDIF
     ISTPDT=1
C
     CALL STORE('W',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
     IPROBE=IPROBE+1
     IF(IPROBE.GT.NPROBE) IPROBE=1
     CALL STORE('R',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
 902 CONTINUE
     GOTO 400
 900 CONTINUE
C
     DO 90 IPROBE=1,NPROBE
     CALL STORE('R',IPROBE,NSTORE,NPM,ISTORE,NSTOR1,NSTOR2)
     IFIN=2
     CALL OUTPUT
 90  CONTINUE
```

```
C
      STOP
      END
C
C----------- END  MAIN  PROGRAM ----------------------------------
C
C
C*********************************************************************
      SUBROUTINE STORE(CHAR,INDEX,NSTRE,NPRM,ISTORE,NSTR1,NSTR2)
C*********************************************************************
C
      DIMENSION ISTORE(NSTRE,NPRM)
      CHARACTER*1 CHAR
C
      COMMON IA1(9804)
      COMMON/COM2/IA2(107)
C
       N1P1=NSTR1+1
      IF(CHAR.EQ.'R') GOTO 1000
C----WRITE
      DO 100 I=1,NSTR1
100   ISTORE(I,INDEX)=IA1(I)
      DO 102 I=N1P1,NSTRE
102   ISTORE(I,INDEX)=IA2(I-NSTR1)
      RETURN
C
1000  CONTINUE
C----READ
      DO 101 I=1,NSTR1
101   IA1(I)=ISTORE(I,INDEX)
      DO 103 I=N1P1,NSTRE
I03   IA2(I-NSTR1)=ISTORE(I,INDEX)
      RETURN
      END
C
C
C *********************************************************************
      SUBROUTINE GRID
C *********************************************************************
C
      INCLUDE 'comp97.inc'
C
      IF(ISTEP.EQ.0) THEN
      NM1=N-1
      NM2=N-2
      ENDIF
C
      IF(IGRID.NE.1) GOTO 100
C-----UNIFORM GRID
      DZ1=ZDIM/FLOAT(NM2)
      DO 10 I=2,NM1
      DZCELL(I)=DZ1
10    CONTINUE
100   IF(IGRID.NE.2) GOTO 101
C-----EXPANDING GRID FROM LOW Z
      DZCELL(2)=ZDIM*(CEXPG-1.)/(CEXPG**NM2-1.)
      DO 11 I=3,NM1
      DZCELL(I)=CEXPG*DZCELL(I-1)
11    CONTINUE
```

```
101  IF(IGRID.NE.3) GOTO 102
C-----EXPANDING GRID FROM HIGH Z
   DZCELL(NM1)=ZDIM*(CEXPG-1.)/(CEXPG**NM2-1.)
   DO 12 I=NM2,2,-1
   DZCELL(I)=CEXPG*DZCELL(I+1)
12   CONTINUE
102  CONTINUE
C-----IGRID=4 INDICATES THAT DZCELL IS GIVEN IN CASE
C-----CALCULATE Z-VALUES
   Z(1)=0.
   Z(2)=0.5*DZCELL(2)
   DO 13 I=3,NM1
   Z(I)=Z(I-1)+0.5*(DZCELL(I-1)+DZCELL(I))
13   CONTINUE
   Z(N)=ZDIM
C-----CALCULATE OTHER CONTROL VOLUME PARAMETERS
   ZBOUND(1)=0.
   DO 14 I=2,NM1
   DZ(I)=Z(I+1)-Z(I-1)
   RECDZ(I)=1./DZ(I)
   ZBOUND(I)=ZBOUND(I-1)+DZCELL(I)
14   CONTINUE
   RETURN
   END
C
C
C***********************************************************************
   SUBROUTINE AREAD
C***********************************************************************
C
   INCLUDE 'comp97.inc'
C
   IF(INDARE.NE.1) GOTO 200
C-----UNIFORM AREA-DISTRIBUTION
C
   DO 10 I=1,N
   AREA(I)=AREAHZ
10   CONTINUE
   AREA(1)=0.
   RETURN
C
C-----LINEAR AND NON-LINEAR DISTRIBUTIONS
C
200  CONTINUE
   IF(INDARE.EQ.2) CEXPA=1.
   DO 20 I=2,NM1
   AREA(I)=(Z(I)/Z(N))**CEXPA*AREAHZ
20   CONTINUE
   AREA(1)=0.
   AREA(NM1)=AREAHZ
   AREA(N)=AREAHZ
   RETURN
   END
C
C
C***********************************************************************
   SUBROUTINE SURF
C***********************************************************************
C
```

```
      INCLUDE 'comp97.inc'
C
C
      LOGICAL STOGEO(NPM),FLAGDZ(NPM)
      DIMENSION UUZSR(NPM,NIM),UUDZR(NPM,NIM),UUZBR(NPM,NIM)
      DATA STOGEO/NPM*.TRUE./
C
C
      IF(ISTEP.EQ.0) THEN
       IF(STOGEO(IPROBE)) THEN
        DO 10 I=1,N
        UUZSR(IPROBE,I)=Z(I)
        UUDZR(IPROBE,I)=DZCELL(I)
        UUZBR(IPROBE,I)=ZBOUND(I)
   10    CONTINUE
        FLAGDZ(IPROBE)=.FALSE.
        STOGEO(IPROBE)=.FALSE.
       ENDIF
       DO 12 I=1,N
       ZSREF(I)=UUZSR(IPROBE,I)
       DZCREF(I)=UUDZR(IPROBE,I)
       ZBREF(I)=UUZBR(IPROBE,I)
   12  CONTINUE
C
C
      ZDIM=ZSSTRT
      DO 11 I=2,N-1
      IF(ZBREF(I).GE.ZDIM)THEN
       NTEST=I
       GOTO 13
      ENDIF
      NTEST=I+1
   11 CONTINUE
   13 CONTINUE
      IF(NTEST.GE.N)THEN
       WRITE(6,'(A,1P2E12.3)')'ZMAX, S.L. =',ZBREF(N-1),ZDIM
       STOP ' SURFACE TOO HIGH IN SURF'
      ENDIF
      N=NTEST+1
      NM1=N-1
      NM2=N-2
      DZCELL(NM1)=ZDIM-ZBREF(NM2)
      IF(FLAGDZ(IPROBE))THEN
       N=N-1
       NM1=N-1
       NM2=N-2
       DZCELL(NM1)=ZDIM-ZBREF(NM2)
      ENDIF
      ZBOUND(NM1)=ZDIM
      Z(N)=ZDIM
      Z(NM1)=ZDIM-0.5*DZCELL(NM1)
      RECDZ(NM1)=1./(Z(N)-Z(NM2))
      ENDIF
C-----END OF ISTEP=0
C-------CALCULATE MOVEMENT OF FREE SURFACE
C
      DELTAZ=QSURF*DT/AREA(NM1)
      ZNEW=ZDIM+DELTAZ
      IF(QSURF.LT.0.) GO TO 30
```

```
C------A RISING SURFACE
C
      ZLIMIT=ZBREF(NM1)+0.2*DZCREF(N)
      IF(ZNEW.LT.ZLIMIT) GO TO 40
C   -- CHANGE NUMBER OF ACTIVE CELLS.
      N=N+1
      NM1=N-1
      NM2=N-2
      DZCELL(NM1)=(QSURF*DT-(ZBREF(NM2)-ZBOUND(NM2))*AREA(NM2))
     1/AREA(NM1)
      ZDIM=ZBREF(NM2)+DZCELL(NM1)
      Z(NM2)=ZSREF(NM2)
      ZBOUND(NM2)=ZBREF(NM2)
      DZCELL(NM2)=DZCREF(NM2)
      Z(NM1)=ZDIM-0. 5*DZCELL(NM1)
      RECDZ(NM2)=1./(Z(NM1)-Z(N-3))
C   -- PROPERTIES FOR NEW ACTIVE CELL
      DO 20 JLOC=1,NFP2
      F(N,JLOC)=F(NM2,JLOC)
      F(NM1,JLOC)=F(NM2,JLOC)
   20 CONTINUE
      GO TO 50
   30 CONTINUE
C-------A SINKING SURFACE
C
      ZLIMIT=ZBREF(NM2)+0.2*DZCREF(NM1)
      IF(ZNEW.GT.ZLIMIT) GO TO 40
C   ---CHANGE NUMBER OF ACTIVE CELLS.
      VOL1=DZCELL(NM1)*AREA(NM1)
      VOL2=DZCELL(NM2)*AREA(NM2)
      N=N-1
      NM1=N-1
      NM2=N-2
      DZCELL(NM1)=DZCREF(NM1)+(ZDIM-ZBREF(NM1))
     1*AREA(N)/AREA(NM1)+QSURF*DT/AREA(NM1)
      ZDIM=ZBREF(NM2)+DZCELL(NM1)
      Z(N+1)=ZSREF(N+1)
      DZCELL(N)=DZCREF(N)
      ZBOUND(N)=ZBREF(N)
      RECDZ(N)=1./(ZSREF(N+1)-ZSREF(NM1))
C---  PROPERTIES FOR CELL NM1(MIXING IN PROPORTION TO VOLUMES)
      VOL3=-DT*QSURF+DZCELL(NM1)*AREA(NM1)
      DO 31 JLOC=1,NFP2
      F(NM1,JLOC)=(F(N,JLOC)*VOL1+F(NM1,JLOC)*VOL2)/VOL3
      F(N,JLOC)=F(NM1,JLOC)
      F(N+1,JLOC)=0.
   31 CONTINUE
      GO TO 50
   40 CONTINUE
C------NUMBER OF ACTIVE CELLS NOT CHANGED.
C
      ZDIM=ZNEW
      IF(DELTAZ.GT.0.) THEN
      DO 41 JLOC=1,NFP2
      F(NM1,JLOC)=(F(NM1,JLOC)*DZCELL(NM1)+
     1 F(NM2,JLOC)*DELTAZ)/(DZCELL(NM1)+DELTAZ)
   41 CONTINUE
      ENDIF
      DZCELL(NM1)=DZCELL(NM1)+DELTAZ
   50 CONTINUE
C-----CHANGES COMMON TO ALL SITUATIONS.
C
      ZBOUND(NM1)=ZDIM
      RECDZ(NM1)=1./(ZDIM-ZSREF(NM2))
      Z(N)=ZDIM
      Z(NM1)=ZDIM-0.5*DZCELL(NM1)
      IF(ISTEP.EQ.LSTEP-1)THEN
       IF(DZCELL(NM1).GT.DZCREF(NM1))THEN
        FLAGDZ(IPROBE)=.TRUE.
       ELSE
        FLAGDZ(IPROBE)=.FALSE.
       ENDIF
      ENDIF
      RETURN
      END
C
C
C
C*********************************************************************
      SUBROUTINE PEA
C ***********************************************************************
C
      INCLUDE 'comp97.inc'
C
      DIMENSION A(NIM),B(NIM),C(NIM),D(NIM)
C-----PEA-ALGORITM
C
      CALL BOUND(1,TLZ)
      J=JRHOV
      CALL BOUND(1,TLZ)
      CALL BOUND(N,THZ)
C  --A AND B
      DO 10 I=2,NM2
      A(I)=DIFREF(I)
      B(I+1)=A(I)
   10 CONTINUE
      NLIMIT=NM1
      IF(MOVE) NLIMIT=NM2
      DO 11 I=2,NLIMIT
      A(I)=A(I)+AMAX1(0.,-QZ(I))/AREA(I)
      B(I+1)=B(I+1)+AMAX1(0.,QZ(I))/AREA(I+1)
   11 CONTINUE
      B(2)=TLZ
      A(NM1)=THZ
C  --C AND D
      DO 12 I=2,NM1
      DCDT=DZCELL(I)/DT
      D(I)=A(I)+B(I)+DCDT
      C(I)=F(I,JRHOV)*DCDT-DZCELL(I)*DPDY(I)
   12 CONTINUE
      DO 13 I=2,NLIMIT
      D(I)=D(I)+(QZ(I)-QZ(I-1))/AREA(I)+QOUTFL(I)/AREA(I)
      C(I)=C(I)+PHIIN(I,J)*QINFL(I)/AREA(I)
   13 CONTINUE
      IF(IKBLZ(J).EQ.1) GOTO 100
      B(2)=0.
      C(2)=C(2)+FLUXLZ(J)
      D(2)=D(2)-TLZ
```

```
100  IF(IKBHZ(J).EQ.1) GOTO 101
     A(NM1)=0.
     C(NM1)=C(NM1)-FLUXHZ(J)
     D(NM1)=D(NM1)-THZ
101  CONTINUE
     DO 14 I=2,NM1
     DAF=A(I)*F(I+1,JRHOV)+B(I)*F(I-1,JRHOV)+C(I)
     SI(I)=SI(I)+CORI/D(I)*DAF
     SIP(I)=-CORI**2*DZCELL(I)/D(I)
 14  CONTINUE
C
     J=JRHOU
C
     RETURN
     END
C
C
C************************************************************************
     SUBROUTINE DFAULT
C************************************************************************
C
     INCLUDE 'comp97.inc'
C
C ---DATA NOT TO BE ALTERED BY USERS
     NF=NJM
     NFP2=NJMP2
     IDIMF=NIM
     TU=0.
     ITEST=1
     IPROBE=1
     DO 1 IJK=1,NIM
     SI(IJK)=0.
     SIP(IJK)=0.
     DIF(IJK)=0.
     DIFREF(IJK)=0.
   1 CONTINUE
     TINY=1.E-15
     GREAT=1.E15
     PI=3.1416
     ISTEP=0
     IFIN=1
C**************************************************************
C*****GROUP 0. TYPE OF FLOW
C    ITYPEF=INDEX FOR TYPE OF FLOW
C         =1 GIVES 1-D TRANSIENT FLOW (DEFAULT)
C         =2 GIVES 2-D PARABOLIC FLOW
     ITYPEF=1
C**************************************************************
C*****GROUP 1. GRID IN SPACE AND TIME
C-----N=NUMBER OF GRID CELLS PLUS 2. MAXIMUM=NIM.
     N=NIM
     TIME=0.
     TLAST=1.E10
     LSTEP=10
C-----GRID DISTRIBUTION IN SPACE
C-----IGRID=INDEX FOR GRID
C        =1 GIVES UNIFORM GRID
C        =2 GIVES EXPANDING GRID FROM LOW Z
C        =3 GIVES EXPANDING GRID FROM HIGH Z
```

```
C         =4 INDICATES THAT THE GRID IS SPECIFIED IN CASE
C  ----SEE MANUAL FOR DETAILS OF THE EXPANDING GRID
      IGRID=1
      CEXPG=1.1
      DO 11 IJK=1,NIM
      DZCELL(IJK)=0.
   11 CONTINUE
C-----TIME STEP VARIATION
C       A VARIABLE TIME STEP IS SPECIFIED BY THE TFRAC FIELD
C       TFRAC/10.,1.,200.,2.,16*0./ GIVES A TIME STEP OF 1.0 S
C       THE FIRST 10 STEPS FOLLOWED BY 200 OF 2.0 S.
C       A CONSTANT TIME STEP IS OBTAINED BY SPECIFYING TFRAC(2)
C       IN CASE.
      DO 12 IJK=1,20
      TFRAC(IJK)=0.
   12 CONTINUE
      TFRAC(1)=1.E8
C*************************************************
C*****GROUP 2. PHYSICAL DIMENSIONS
      XDIM=1.E10
      YDIM=1.E10
      ZDIM=1.E10
C-----VERTICAL AREA DISTRIBUTION
C
C-----INDARE=INDEX FOR AREA-DISTRIBUTION
C-----      =1 INDICATES UNIFORM AREA
C-----      =2 INDICATES LINEAR DISTRIBUTION
C-----      =3 INDICATES NON-LINEAR DISTRB.,SEE MANUAL
C-----      =4 DISTR. SPECIFIED IN CASE
      INDARE=1
      AREAHZ=1.0
      CEXPA=2.
C*************************************************
C*****GROUP 3. DEPENDENT VARIABLES
C     F(I,JRHOU)=X-DIRECTION MOMENTUM
C     F(I,JRHOV)=Y-DIRECTION MOMENTUM
C     F(I,JH)=HEAT-ENERGY
C     F(I,JS)=SALINITY
C     F(I,JK)=TURBULENT KINETIC ENERGY
C     F(I,JD)=DISSIPATION OF TURBULENT KINETIC ENERGY
C     F(I,JC1)=CONCENTRATION NO.1
C     F(I,JC2)=CONCENTRATION NO.2
C     F(I,JC3)=CONCENTRATION NO.3
C     F(I,JC4)=CONCENTRATION NO.4
C     F(I,10+(NJM-10))=ADDITIONAL VARIABLES ACTIVATED FOR NJM>10.
C     F(I,JEMU)=DYNAMICAL EDDY VISCOSITY
C     F(I,JTE)=TEMPERATURE
      JRHOU=1
      JRHOV=2
      JH=3
      JS=4
      JK=5
      JD=6
      JC1=7
      JC2=8
      JC3=9
      JC4=10
      DO 31 IJK=1,NJM
      SOLVAR(IJK)=.FALSE.
```

```
  31 CONTINUE
     JEMU=NJMP1
     JTE=NJMP2
C*****************************************************
C*****GROUP 4. PROPERTIES
     CPHEAT=4190.
     RHOREF=1000.
     EMULAM=0.0013
     DO 41 IJK=1,NJM
     PRL(IJK)=1.
  41 CONTINUE
     PRL(3)=9.5
     PRL(4)=1000.
     AGRAV=9.81
C*******************************************************
C*****GROUP 5. EQUATION OF STATE
C-----RHO=RHOREF*(1.-C1RHO*(T-TREF)**2+C2RHO*S
C       +C3RHO*JC1+C4RHO*JC2+C5RHO*JC3+C6RHO*JC4)
     C1RHO=7.18E-6
     C2RHO=8.E-4
     TREF=3.98
     C3RHO=0.
     C4RHO=0.
     C5RHO=0.
     C6RHO=0.
C********************************************************
C*****GROUP 6. TURBULENCE MODEL
C-----ITURBM=INDEX FOR TURBULENCE MODEL
C-----      =1 GIVES CONSTANT VALUE (=EMUCON)
C-----      =2 GIVES K-E MODEL
C-----      =3 GIVES K-E MODEL WITH BUOYANCY EFFECTS
C-----      =4 INDICATES THAT F(I,JEMU) IS SPECIFIED IN CASE
C-------IPRSC=INDEX FOR TURBULENT PRANDTL/SCHMITH NUMBER
C            USED FOR HEAT,SALINITY AND CONCENTRATIONS
C           =1 INDICATES THAT CONSTANT VALUES ,GIVEN BELOW,
C            ARE USED.
C           =2 INDICATES THAT THE NUMBERS ARE AFFECTED
C            BY BUOYANCY.NOTE:SHOULD ONLY BE USED WITH
C            ITURBM EQUAL TO 2 OR 3.
     IPRSC=2
     ITURBM=3
     EMUCON=0.
     DO 61 IJK=1,NJM
     PRT(IJK)=1.
  61 CONTINUE
     PRT(5)=1.4
     PRT(6)=1.3
C-----CONSTANTS IN TURBULENCE MODEL. SHOULD NOT BE CHANGED.
     CD=0.09
     RTCD=0.3
     CD75=0.164
     C1=1.44
     C2=1.92
     C3=0.8
     C1PR=0.63
     C2PR=0.13
     C3PR=0.063
C****************************************************************
C*****GROUP 7. SOURCE TERMS
```

```
C
C----CORIOLIS PARAMETER
      CORI=1.E-4
C-----PRESSURE GRADIENTS
C     INDPX=INDEX FOR PRESSURE GRADIENTS IN X-DIRECTION
C          =1 GIVES PRESCRIBED CONSTANT PRESSURE
C             GRADIENTS ,DPDXP.
C          =2 GIVES PRESCRIBED MASSFLOW,RHOUP.ONLY
C             RELEVANT FOR STEADY STATE PROBLEMS.
C          =3 GIVES PRESSURE GRADIENT DEVELOPMENT ACCORDING TO
C             HORIZONTAL EXTENT OF WATERBODY.ONLY RELEVANT TO
C             LAKES AND RESERVOIRS.
C          =4 INDICATES THAT THE PRESSURE GRADIENTS ARE TO BE
C             READ FROM SEPARATE FILE AS A TIME SERIES.
C          =-1,-2,-3 OR -4 AS ABOVE,BUT WITH BUOYANCY DAMPING
C             OF PRESSURE GRADIENTS(EFFECT OF TILTED TERMOCLINE).
C     INDPY=SAME FOR Y-DIRECTION
      RHOUP=0.
      RHOVP=0.
      DPDXP=0.
      DPDYP=0.
      PFILT=1.
      INDPX=1
      INDPY=1
C-----IN- AND OUTFLOWS.
C-----SEE MANUAL FOR INSTRUCTIONS ON USE
      DO 71 IJK=1,NIM
      QZ(IJK)=0.
      QINFL(IJK)=0.
      QOUTFL(IJK)=0.
      DO 72 IKJ=1,NJM
      PHIIN(IJK,IKJ)=0.
   72 CONTINUE
   71 CONTINUE
C-----SHORT-WAVE RADIATION
C      ASSUMED TO PENETRATE THE WATER BODY.
C      FLXRAD=SHORT-WAVE RADIATION.
C      RADFRA=FRACTION ASSUMED TO BE A BOUNDARY FLUX
C      BETA=EXTINTION COEFFICIENT
      FLXRAD=0.0
      RADFRA=0.4
      BETA=0.1
C******************************************************************
C*****GROUP 8. INITIAL DATA
      DO 81 IJK=1,NIM
      DPDX(IJK)=0.
      DPDY(IJK)=0.
      FW(IJK)=0.
      DO 82 IKJ=1,NJMP2
      F(IJK,IKJ)=0.
   82 CONTINUE
   81 CONTINUE
C-----INITIALISE DEPENDENT VARIABLES
C     ISTPR=INDEX FOR STARTING PROFILES
C          =1 PROFILES ARE SPECIFIED WITH VST1(1-NJM)-ZST2(1-NJM)
C             SEE MANUAL.
C          =2 PROFILES ARE SPECIFIED IN CASE WITHOUT THE USE
C             OF VST1(1-NJM)-ZST2(1-NJM).
C     --NOTE:DEFAULT VALUE FOR ALL VARIABLES IS 0.0.
```

```
      ISTPR=1
      DO 83 IJK=1,NJM
      VST1(IJK)=0.
      VST2(IJK)=0.
      ZST1(IJK)=0.
      ZST2(IJK)=0.
   83 CONTINUE
C*********************************************************************
C*****GROUP 9. BOUNDARY CONDITIONS
C
C-----ITYPEH=INDEX FOR TYPE OF BOUNDARY AT HIGH Z
C        =1 GIVES SOLID WALL(STATIONARY OR MOVING)
C        =2 GIVES SYMMETRY LINE
C     ITYPEL=SAME FOR LOW Z BOUNDARY
C
C-----IKBHZ(J)=INDEX FOR KIND OF BOUNDARY CONDITION FOR
C          VARIABLE J AT HIGH Z BOUNDARY
C          =1 GIVES PRESCRIBED VALUE
C          =2 GIVES PRESCRIBED FLUX
C     IKBLZ(J)=SAME FOR LOW Z BOUNDARY
C-----ITRHZ(J)=INDEX FOR TIMEDEPENDENCE OF BOUNDARY FOR
C          VARIABLE J
C          =1 GIVES STATIONARY CONDITIONS
C          =2 GIVES TRANSIENT CONDITIONS SPECIFIED FROM CASE-
C           SUBROUTINE.SEE MANUAL FOR INSTRUCTIONS ON USE.
C          =3 GIVES TRANSIENT CONDITIONS READ FROM FILE
C     ITRLZ(J)=SAME FOR LOW Z BOUNDARY
C-----IKBOT(J)=INDEX FOR KIND OF BEHAVIOR AT BOTTOM FOR VARIABLE J
C          ONLY RELEVANT FOR CASES WITH VERTICAL AREA-DISTRIB.
C          =1 GIVES "CONSERVATIVE" CONDITION.SEE MANUAL.
C          =2 GIVES "NON-CONSERVATIVE" CONDITION.SEE MANUAL.
C-----SPECIFICATION FOR STATIONARY BOUNDARY CONDITIONS
C
C-----SPECIFICATION FOR TRANSIENT CONDITIONS(ITRHZ OR ITRLZ=2).SEE MANUAL
C
C  ----SPECIFICATION OF WALL-FKN PARAMETERS.
C
      ITYPEH=1
      ITYPEL=1
      DO 91 IJK=1,NJM
      IKBHZ(IJK)=2
      IKBLZ(IJK)=2
      ITRHZ(IJK)=1
      ITRLZ(IJK)=1
      IKBOT(IJK)=1
      FLUXHZ(IJK)=0.
      FLUXLZ(IJK)=0.
      V1HZ(IJK)=0.
      V2HZ(IJK)=0.
      V3HZ(IJK)=0.
      V4HZ(IJK)=0.
      V5HZ(IJK)=0.
      V1LZ(IJK)=0.
      V2LZ(IJK)=0.
      V3LZ(IJK)=0.
      V4LZ(IJK)=0.
      V5LZ(IJK)=0.
      STANTN(IJK)=1.E-3
   91 CONTINUE
```

```
    IKBOT(1)=2
    IKBOT(2)=2
    IKBOT(5)=2
    IKBOT(6)=2
    STANTN(1)=1.
    STANTN(2)=1.
    STANTN(3)=0.05
    STANTN(5)=1.
    STANTN(6)=1.
    CAPPA=0.4
    C3B=9.
    ROULHZ=0.
    ROULLZ=0.
C***********************************************************************
C*****GROUP 10. LIMITS AND NUMBERS
    EMTMIN=1.E-6
    FKMIN=1.E-15
    FDMIN=1.E-15
    TAUMIN=1.E-3
    KINDAV=1
C***********************************************************************
C*****GROUP 11. PRINT OUT
C--------PRINT CONTROL
C  --SET ITPLOT=2 FOR CROSS-STREAM PLOT, =1 FOR NO PLOT
    ITPLOT=2
C  --SET NSTAT,NPROF,NPLOT TO NUMBER OF STEPS BETWEEN OUTPUT OF
C    STATION VALUES,PROFILES AND CROSS-STREAM PLOTS RESPECTIVELY
    NSTAT=10
    NPROF=50
    NPLOT=100
C  --SET INIOUT .FALSE.  FOR NO INITIAL OUTPUT
    INIOUT=.TRUE.
C
C---- SELECT PROFILES TO BE PRINTED AND PLOTTED.
C-----U,V,T,S,1C,2C,3C,4C,K,E,EMU,SIGM,DPDX,DPDY,W,PRSCN,RIF,N,UW,VW
C    1,2,3,4, 5, 6, 7, 8,9,10,11,  12,  13,  14,15,  16,17,18,19,20
C-----PRINTED
C-----PLOTTED
    DO 111 IJK=1,20
    PRPROF(IJK)=.FALSE.
    PLPROF(IJK)=.FALSE.
  111 CONTINUE
C
C-----PARTICLE TRACKING.SEE MANUAL.
C-----INDPT=INDEX FOR PARTICLE TRACKING
C        =0 GIVES NO TRACKING
C        =1-4 ONE TO FOUR PARTICLES ARE TRACKED
C
    INDPT=0
    ILEVEL(1)=0
    ILEVEL(2)=0
    ILEVEL(3)=0
    ILEVEL(4)=0
    IPSAVE=1000
C***************************************************************
C*****GROUP 12.LINKED RUNS.
    DO 121 IJK=1,NPM
    NSTPDT(IJK)=1
  121 CONTINUE
```

```
      NPROBE=1
C*****************************************************************
C*****GROUP 13. MOVING FREE SURFACE.
      MOVE=.FALSE.
      ZSSTRT=0.
      PREEVA=0.
C*****************************************************************
      RETURN
      END
C
C
C*****************************************************************
      SUBROUTINE PHYS
C*****************************************************************
C
      INCLUDE 'comp97.inc'
C
C-----------------------------------------------------------------------
CHAPTER A  A  A  A  EDDY VISCOSITY AND PRANDTL/SCHMIDT NUMBERS A A A
C
      IF(J.NE.0) GOTO 14
C
      IF(ITURBM.EQ.1.OR.ITURBM.EQ.4) GOTO 16
C ---EDDY VISCOSITY
      DO 10 I=2,NM1
      F(I,JEMU)=CD*RHO(I)*F(I,JK)*F(I,JK)/F(I,JD)+EMTMIN
 10   CONTINUE
C
C-----PRANDTL/SCHMIDT NUMBER
      IF(IPRSC.NE.2) GOTO 16
      DO 15 I=2,NM1
      DTDZ=(F(I+1,JTE)-F(I-1,JTE)+TINY)*RECDZ(I)
      DSDZ=(F(I+1,JS)-F(I-1,JS)+TINY)*RECDZ(I)
      DC1DZ=(F(I+1,JC1)-F(I-1,JC1)+TINY)*RECDZ(I)
      DC2DZ=(F(I+1,JC2)-F(I-1,JC2)+TINY)*RECDZ(I)
      DC3DZ=(F(I+1,JC3)-F(I-1,JC3)+TINY)*RECDZ(I)
      DC4DZ=(F(I+1,JC4)-F(I-1,JC4)+TINY)*RECDZ(I)
      BPR=-(F(I,JK)*F(I,JK))/(F(I,JD)*F(I,JD))*AGRAV*
     1 (-2.*C1RHO*(F(I,JTE)-TREF)*DTDZ
     2 +C2RHO*DSDZ+C3RHO*DC1DZ+C4RHO*DC2DZ
     3 +C5RHO*DC3DZ+C6RHO*DC4DZ)
      IF(BPR.LT.TINY) BPR=TINY
      PRSCNU(I)=(C1PR+C2PR*BPR)/(1.+C3PR*BPR)
 15   CONTINUE
 16   CONTINUE
C ---REFERENCE DIFF-VALUES AT CELL-BOUNDARIES
      DO 13 I=2,NM2
      IF(KINDAV.EQ.2) THEN
C ---HARMONIC MEAN
      EMU(I)=2.*(Z(I+1)-Z(I))/(DZCELL(I)/F(I,JEMU)
     1 +DZCELL(I+1)/F(I+1,JEMU))
      ELSE
C ---ARITHMETIC MEAN
      EMU(I)=0.5*(F(I+1,JEMU)*DZCELL(I)+F(I,JEMU)*DZCELL(I+1))
     1/(Z(I+1)-Z(I))+EMULAM
      ENDIF
      DIFREF(I)=EMU(I)/(0.5*(RHO(I)+RHO(I+1)))
     1/(Z(I+1)-Z(I))
 13   CONTINUE
```

```
      RETURN
 14   CONTINUE
C
C-------------------------------------------------------------------
CHAPTER B  B  B  B  B  B  CHOOSE VARIABLE B  B  B  B  B  B   B  B  B
C
      IF(J.EQ.JRHOU) GOTO 300
      IF(J.EQ.JRHOV) GOTO 400
      IF(J.EQ.JH.OR.J.EQ.JS) GOTO 500
      IF(J.EQ.JK) GOTO 600
      IF(J.EQ.JD) GOTO 700
      IF(J.GE.JC1) GOTO 500
C
C-------------------------------------------------------------------
CHAPTER C  C  C  C  C  C  U-MOMENTUM EQUATION C  C  C  C  C  C  C  C
C
 300  CONTINUE
C
      DO 30 I=2,NM1
      DIF(I)=DIFREF(I)
      SI(I)=0.
      SIP(I)=0.0
 30   CONTINUE
C
      IF(ABS(CORI).GT.TINY.AND.SOLVAR(JRHOV)) CALL PEA
360   CONTINUE
C-----PRESSURE GRADIENT
      IF(ABS(INDPX).NE.1) GOTO 310
      DO 311 I=2,NM1
311   DPDX(I)=DPDXP
      GOTO 340
310   CONTINUE
C-----CALCULATE MASS FLOW
      XMFL=0.0
      DO 31 I=2,NM1
      XMFL=XMFL+DZCELL(I)*F(I,JRHOU)
 31   CONTINUE
      IF(ABS(INDPX).NE.2) GOTO 320
      DO 32 I=2,NM1
      DPDX(I)=DPDX(I)+PFILT*(XMFL-RHOUP)
32    CONTINUE
      GOTO 340
320   IF(ABS(INDPX).NE.3) GOTO 330
      FACTPR=PI*PI*PFILT*PFILT*DT*XMFL*AGRAV/XDIM/XDIM
      DO 33 I=2,NM1
      DPDX(I)=DPDX(I)+FACTPR
33    CONTINUE
330   IF(ABS(INDPX).NE.4) GOTO 340
C-----READ DPDX FROM SEPARATE FILE
340   CONTINUE
      IF(INDPX.GT.0) GOTO 350
C  ---EFFECT OF STRATIFICATION
      DDIFF=AMAX1(0.05,-(RHO(NM1)-RHO(2)))
      DO 34 I=2,NM1
      DCORR=(-(RHO(I)-RHO(2))+0.05)/DDIFF
      IF(DCORR.GT.1.) DCORR=1.
      IF(DCORR.LT.0.01) DCORR=0.01
34    DPDX(I)=DPDX(I)*DCORR
350   CONTINUE
```

```
     DO 35 I=2,NM1
     SI(I)=SI(I)-DPDX(I)
35   CONTINUE
     CALL CASE(3)
     RETURN
C
C-----------------------------------------------------------------
CHAPTER D  D  D  D  D  D  V-MOMENTUM EQUATION D  D  D  D  D  D  D  D
C
 400  CONTINUE
     DO 40 I=2,NM1
     DIF(I)=DIFREF(I)
     SI(I)=-CORI*F(I,JRHOU)
     SIP(I)=0.0
 40  CONTINUE
C
C-----PRESSURE GRADIENT
     IF(ABS(INDPY).NE.1) GOTO 410
     DO 411 I=2,NM1
411  DPDY(I)=DPDYP
     GOTO 440
410  CONTINUE
C  --CALCULATE MASS FLOW
     YMFL=0.0
     DO 41 I=2,NM1
     YMFL=YMFL+DZCELL(I)*F(I,JRHOV)
  41  CONTINUE
     IF(ABS(INDPY).NE.2) GOTO 420
     DO 42 I=2,NM1
     DPDY(I)=DPDY(I)+PFILT*(YMFL-RHOVP)
42   CONTINUE
     GOTO 440
420  IF(ABS(INDPY).NE.3) GOTO 430
     FACTPR=PI*PI*PFILT*PFILT*DT*YMFL*AGRAV/YDIM/YDIM
     DO 43 I=2,NM1
     DPDY(I)=DPDY(I)+FACTPR
43   CONTINUE
430  IF(ABS(INDPY).NE.4) GOTO 440
C-----READ DPDY FROM SEPARATE FILE
440  CONTINUE
     IF(INDPY.GT.0) GOTO 450
C  ---EFFECT OF STRATIFICATION
     DDIFF=AMAX1(0.05,-(RHO(NM1)-RHO(2)))
     DO 44 I=2,NM1
     DCORR=(-(RHO(I)-RHO(2))+0.05)/DDIFF
     IF(DCORR.GT.1.) DCORR=1.
     IF(DCORR.LT.0.01) DCORR=0.01
44   DPDY(I)=DPDY(I)*DCORR
450  CONTINUE
     DO 45 I=2,NM1
     SI(I)=SI(I)-DPDY(I)
45   CONTINUE
     CALL CASE(3)
     RETURN
C
C-----------------------------------------------------------------
CHAPTER E  E  SOURCES AND DIFFUSION COEFFICIENTS FOR JH,JS,JC1-JC4
C
 500  CONTINUE
```

```
      DO 57 I=2,NM1
      SI(I)=0.
      SIP(I)=0.
 57   CONTINUE
C
C ---EFFECTIVE PRANDTL NUMBER
      DO 50 I=2,NM1
      PRTJ=PRT(J)
      IF(IPRSC.EQ.2) PRTJ=PRSCNU(I)
      PREF(I)=(F(I,JEMU)+EMULAM)/(F(I,JEMU)/PRTJ+EMULAM/PRL(J))
  50  CONTINUE
      PREF(NM1)=PREF(NM2)
C
C ---DIFFUSION VALUES
      DO 51 I=2,NM2
      PREFJ=0.5*(PREF(I+1)+PREF(I))
      DIF(I)=DIFREF(I)/PREFJ
  51  CONTINUE
C
      IF(J.NE.JH.OR.FLXRAD.GT.TINY) GOTO 56
C ---SHORT-WAVE RADIATION
      DO 52 I=2,NM1
      SHIG=EXP(-BETA*(ZDIM-ZBOUND(I)))
      SLOW=AREA(I-1)/AREA(I)*EXP(-BETA*(ZDIM-ZBOUND(I-1)))
      SI(I)=-(1.-RADFRA)*FLXRAD*(SHIG-SLOW)/DZCELL(I)
   52 CONTINUE
      SI(NM1)=SI(NM1)-RADFRA*FLXRAD/DZCELL(NM1)
  56  CONTINUE
C
      CALL CASE(3)
      RETURN
C
C----------------------------------------------------------------------
CHAPTER F  F  F  F  F  F  TURBULENT KINETIC ENERGY F  F  F  F  F  F
C
 600 CONTINUE
      DO 60 I=3,NM2
      BUO(I)=0.
      FJK=F(I,JK)
      FJD=F(I,JD)
      DUDZ=(F(I+1,JRHOU)-F(I-1,JRHOU)+TINY)*RECDZ(I)/RHO(I)
       IF (ITYPEF.EQ.1) THEN
       DVDZ=(F(I+1,JRHOV)-F(I-1,JRHOV)+TINY)*RECDZ(I)/RHO(I)
      GRADSQ(I)=DUDZ*DUDZ+DVDZ*DVDZ
       ELSE
       DWDZ=(FW(I)-FW(I-1))/DZCELL(I)
       GRADSQ(I)=DUDZ*DUDZ+2.*DWDZ*DWDZ
       ENDIF
C
C  --BUOYANCY PARAMETERS
      DTDZ=(F(I+1,JTE)-F(I-1,JTE)+TINY)*RECDZ(I)
      DSDZ=(F(I+1,JS)-F(I-1,JS)+TINY)*RECDZ(I)
      DC1DZ=(F(I+1,JC1)-F(I-1,JC1)+TINY)*RECDZ(I)
      DC2DZ=(F(I+1,JC2)-F(I-1,JC2)+TINY)*RECDZ(I)
      DC3DZ=(F(I+1,JC3)-F(I-1,JC3)+TINY)*RECDZ(I)
      DC4DZ=(F(I+1,JC4)-F(I-1,JC4)+TINY)*RECDZ(I)
      IF(IPRSC.EQ.2) GOTO 602
      BUO(I)=AGRAV*(-2.*C1RHO*(F(I,JTE)-TREF)
     1 *DTDZ/PRT(JH)+C2RHO*DSDZ/PRT(JS)+C3RHO*DC1DZ/PRT(JC1)
```

```
      SI(I)=F(I,JEMU)*(C1*GRADSQ(I)+C3*BUO(I))*FJD/FJK/RHO(I)
      SIP(I)=-C2*FJD/FJK
      DIF(I)=DIFREF(I)/PRT(JD)
 70   CONTINUE
C
C-----D IS PRESCRIBED NEAR BOUNDARIES
C   --AT LOW Z
      DIF(2)=DIFREF(2)/PRT(JD)
      SI(2)=FACTLZ**1.5*CD75/(CAPPA*Z(2))*GREAT
      IF(F(2,JK).LT.(FKMIN+TINY))SI(2)=FDMIN*GREAT
      IF(ITYPEL.EQ.2) SI(2)=F(3,JD)*GREAT
      SIP(2)=-GREAT
C
C   --AT HIGH Z
      ZREF=ZDIM-Z(NM1)
      SI(NM1)=FACTHZ**1.5*CD75/(CAPPA*ZREF)*GREAT
      IF(F(NM1,JK).LT.(FKMIN+TINY))SI(NM1)=FDMIN*GREAT
      IF(ITYPEH.EQ.2)   SI(NM1)=F(NM2,JD)*GREAT
      SIP(NM1)=-GREAT
      CALL CASE(3)
      RETURN
C ----------------------------------------------------------------------
      END
C
C
C*********************************************************************
      SUBROUTINE COMP
C*********************************************************************
C
      INCLUDE 'comp97.inc'
C
C
      DIMENSION A(NIM),B(NIM),C(NIM),D(NIM),F1D(NJP2NI)
      REAL CUP,CDOWN,TERM,DCDT
C
      EQUIVALENCE(F1D(1),F(1,1))
C
      DOUBLE PRECISION A,B,C,D
C
C---------------------------------------------------------------------
CHAPTER  A  A  A  A  CALCULATE URUP, URUPUP AND FW   A  A  A  A  A  A
C
      J=0
      CALL PHYS
C
      IF (ITYPEF.EQ.2) THEN
C     STORE X-DIRECTION MOMENTUM ONE AND TWO INTEGRATION STEPS UP
C     ONLY RELEVANT FOR 2-D PARABOLIC FLOW
      DO I=2,NM1
      URUPUP(I)=URUP(I)
      URUP(I)=F1D(I)
      IF (ISTEP.EQ.0) URUPUP(I)=URUP(I)
      ENDDO
      ENDIF
C
      DO 480 J=1,NF
C
      IF (ITYPEF.EQ.2.AND.J.EQ.2) THEN
C     CALCULATE VERTICAL VELOCITY FROM CONTINUITY EQUATION
```

```
     2 +C4RHO*DC2DZ/PRT(JC2)+C5RHO*DC3DZ/PRT(JC3)
     3 +C6RHO*DC4DZ/PRT(JC4))
      GOTO 601
602   BUO(I)=AGRAV/PRSCNU(I)*(-2.*C1RHO*(F(I,JTE)-TREF)
     1 *DTDZ+C2RHO*DSDZ+C3RHO*DC1DZ
     2 +C4RHO*DC2DZ+C5RHO*DC3DZ+C6RHO*DC4DZ)
601   CONTINUE
C
C ---SOURCE TERMS AND DIFFUSION
      SI(I)=F(I,JEMU)*(GRADSQ(I)+BUO(I))/RHO(I)
      SIP(I)=-FJD/FJK
      DIF(I)=DIFREF(I)/PRT(JK)
 60   CONTINUE
C
C-----K IS PRESCRIBED NEAR BOUNDARIES
C  --AT LOW Z
      ZREF=Z(2)
      FACTST=(ABS(FLUXLZ(JRHOU)+TINY)**1.5
     1 +ABS(FLUXLZ(JRHOV)+TINY)**1.5)/RHOREF**1.5/CD75
      FACTBU=CAPPA*ZREF*AGRAV/CD75*(
     1 FLUXLZ(JH)*2.*C1RHO*(F(2,JTE)-TREF)/RHOREF/CPHEAT
     2 -FLUXLZ(JS)*C2RHO-FLUXLZ(JC1)*C3RHO
     3 -FLUXLZ(JC2)*C4RHO-FLUXLZ(JC3)*C5RHO-FLUXLZ(JC4)*C6RHO)
      IF(FACTBU.LT.TINY**2) FACTBU=TINY**2
      FACTLZ=(FACTST+FACTBU)**.67
      IF(FACTLZ.LT.FKMIN)FACTLZ=FKMIN
      SIP(2)=-GREAT
      SI(2)=FACTLZ*GREAT
      IF(ITYPEL.EQ.2) SI(2)=F(3,JK)*GREAT
      DIF(2)=DIFREF(2)/PRT(JK)
C
C  --AT HIGH Z
      ZREF=ZDIM-Z(NM1)
      COEFKS=CD75
      FACTST=(ABS(FLUXHZ(JRHOU)+TINY)**1.5
     1 +ABS(FLUXHZ(JRHOV)+TINY)**1.5)/RHOREF**1.5/COEFKS
      FACTBU=CAPPA*ZREF*AGRAV/CD75*(
     1 FLUXHZ(JH)*2.*C1RHO*(F(NM1,JTE)-TREF)/RHOREF/CPHEAT
     2 -FLUXHZ(JS)*C2RHO-FLUXHZ(JC1)*C3RHO
     3 -FLUXHZ(JC2)*C4RHO-FLUXHZ(JC3)*C5RHO-FLUXHZ(JC4)*C6RHO)
      IF(FACTBU.LT.TINY**2) FACTBU=TINY**2
      FACTHZ=(FACTST+FACTBU)**.67
      IF(FACTHZ.LT.FKMIN)FACTHZ=FKMIN
      SI(NM1)=FACTHZ*GREAT
      IF(ITYPEH.EQ.2)  SI(NM1)=F(NM2,JK)*GREAT
      SIP(NM1)=-GREAT
      CALL CASE(3)
      RETURN
C
C------------------------------------------------------------------
CHAPTER G  G  G  G  G  G  DISSIPATION OF TURBULENCE G  G  G  G  G  G
C
 700  CONTINUE
      DO 70 I=3,NM2
      FJK=F(I,JK)
      IF(FJK.LT.FKMIN) FJK=FKMIN
      FJD=F(I,JD)
C
C  ---SOURCE TERMS AND DIFFUSION
```

```
C    ONLY RELEVANT FOR 2-D PARABOLIC FLOW
C    PRESCRIBED ZERO VERTICAL VELOCITY AT THE LOWER BOUNDARY
C    FW(1) IS THE VERTICAL VELOCITY AT THE LOWER BOUNDARY
C    FW(I) IS THE VERTICAL VELOCITY AT THE UPPER WALL OF CELL I
      FW(1)=0.
      DO I=2,NM1
      FW(I)=FW(I-1)+(URUP(I)-F1D(I))*DZCELL(I)/(DT*RHO(I))
      ENDDO
      ENDIF
C
C-----------------------------------------------------------------
CHAPTER B  B  B  B  B  1-D ARRAY B  B  B  B  B  B  B  B  B  B  B  B
C
      IF(.NOT.SOLVAR(J)) GOTO 480
      IDJ=IDIMF*(J-1)
      I1J=1+IDJ
      I2J=2+IDJ
      INM1J=NM1+IDJ
      INJ=N+IDJ
C
C-----------------------------------------------------------------
CHAPTER C  C  C  C  C  PHYSICS C  C  C  C  C  C  C  C  C  C  C  C  C
C
      CALL PHYS
      THZ=0.0
      TLZ=0.0
      IF(ITYPEH.NE.2) CALL BOUND(N,THZ)
      IF(ITYPEL.NE.2) CALL BOUND(1,TLZ)
      IF(ITEST.EQ.1) GOTO 450
      WRITE(6,451) J,(DIF(I),I=2,NM1)
      WRITE(6,452) (SI(I),I=2,NM1)
      WRITE(6,453) (SIP(I),I=2,NM1)
 451  FORMAT(24H COMP SOLVE TESTS FOR J=,I3/8H DIF(I)=/(3X,1P6E11.3))
 452  FORMAT(7H SI(I)=/(3X,1P6E11.3))
 453  FORMAT(8H SIP(I)=/(3X,1P6E11.3))
 450  CONTINUE
C
C-----------------------------------------------------------------
CHAPTER D  D  D  D  D  COEFFICIENTS D  D  D  D  D  D  D  D  D  D  D
C
      IF (ITYPEF.EQ.1) THEN
C     1-D TRANSIENT FLOW
C-----A'S AND B'S
      DO I=2,NM2
      A(I)=DIF(I)
      B(I+1)=A(I)
      IF(IKBOT(J).EQ.1) B(I+1)=A(I)*AREA(I)/AREA(I+1)
      ENDDO
      NLIMIT=NM1
      IF(MOVE) NLIMIT=NM2
      DO 339 I=2,NLIMIT
      A(I)=A(I)+AMAX1(0.,-QZ(I))/AREA(I)
      B(I+1)=B(I+1)+AMAX1(0.,QZ(I))/AREA(I+1)
339   CONTINUE
      B(2)=TLZ
      A(NM1)=THZ
C-----C'S AND D'S
      DO 485 I=2,NM1
      IJ=I+IDJ
```

```
   DCDT=DZCELL(I)/DT
   D(I)=A(I)+B(I)+DCDT-DZCELL(I)*SIP(I)
 485 C(I)=F1D(IJ)*DCDT+DZCELL(I)*SI(I)
   DO 487  I=2,NLIMIT
   D(I)=D(I)+(QZ(I)-QZ(I-1))/AREA(I)+QOUTFL(I)/AREA(I)
   C(I)=C(I)+PHIIN(I,J)*QINFL(I)/AREA(I)
 487 CONTINUE
   IF(IKBLZ(J).EQ.1) GOTO 486
   B(2)=0.0
   C(2)=C(2)+FLUXLZ(J)
   D(2)=D(2)-TLZ
 486 IF(IKBHZ(J).EQ.1) GOTO 491
   A(NM1)=0.0
   C(NM1)=C(NM1)-FLUXHZ(J)
   D(NM1)=D(NM1)-THZ
 491  CONTINUE
C
   ELSEIF (ITYPEF.EQ.2) THEN
C    2-D PARABOLIC FLOW
C-----A'S AND B'S
   DO I=2,NM2
C    POWER-LAW SCHEME
   A(I)=MAX(0.,DIF(I)*(1.-0.1*ABS(FW(I)/DIF(I)))**5.)
   1+MAX(0.,-FW(I))
   B(I+1)=A(I)+FW(I)
   ENDDO
C-----B(2) AND A(NM1)
   B(2)=TLZ
   A(NM1)=THZ
   IF (IKBLZ(J).EQ.2) B(2)=0.
   IF (IKBHZ(J).EQ.2) A(NM1)=0.
C-----C'S AND D'S
   DO I=2,NM1
   IJ=I+IDJ
   CDOWN=F1D(I)/RHO(I)
   CUP=URUP(I)/RHO(I)
   IF (J.EQ.JRHOU) CUP=URUPUP(I)/RHO(I)
   DCDT=DZCELL(I)/DT
   TERM=FW(I)-FW(I-1)+A(I)+B(I)
   D(I)=DCDT*CDOWN+TERM-DZCELL(I)*SIP(I)
   C(I)=DCDT*CUP*F1D(IJ)+DZCELL(I)*SI(I)
   ENDDO
   IF(IKBLZ(J).EQ.2) C(2)=C(2)+FLUXLZ(J)
   IF(IKBHZ(J).EQ.2) C(2)=C(2)-FLUXHZ(J)
   ENDIF
C
   IF(ITEST.EQ.1) GOTO 464
   WRITE(6,405) (A(I),I=2,NM1)
   WRITE(6,406) (B(I),I=2,NM1)
   WRITE(6,407) (C(I),I=2,NM1)
   WRITE(6,408) (D(I),I=2,NM1)
 405  FORMAT(6H A(I)=/(3X,1P6E11.3))
 406  FORMAT(6H B(I)=/(3X,1P6E11.3))
 407  FORMAT(6H C(I)=/(3X,1P6E11.3))
 408  FORMAT(6H D(I)=/(3X,1P6E11.3))
C
C-----------------------------------------------------------------------
CHAPTER E  E  E  E  E  SOLVE FOR NEW F'S E  E  E  E  E  E  E  E  E
C
```

```
 464  C(2)=(B(2)*F1D(I1J)+C(2))/D(2)
      D(2)=A(2)/D(2)
      DO 465 I=3,NM1
      T=1./(D(I)-B(I)*D(I-1))
      D(I)=A(I)*T
 465  C(I)=(B(I)*C(I-1)+C(I))*T
      DO 466 IDASH=1,NM2
      I=N-IDASH
      IJ=I+IDJ
 466  F1D(IJ)=D(I)*F1D(IJ+1)+C(I)
C
C------------------------------------------------------------------
CHAPTER F  F  F  F  F  ADJUST F(1,J),F(N,J) F  F  F  F  F  F  F  F
C
      IF(IKBLZ(J).EQ.1) GOTO 468
      F1D(I1J)=F1D(I2J)+FLUXLZ(J)/(TLZ+TINY)
      GOTO 460
  468 FLUXLZ(J)=TLZ*(F1D(I1J)-F1D(I2J))
  460 IF(IKBHZ(J).EQ.1) GOTO 472
      F1D(INJ)=F1D(INM1J)-FLUXHZ(J)/(THZ+TINY)
      GOTO 470
  472 FLUXHZ(J)=THZ*(F1D(INM1J)-F1D(INJ))
C
 470  IF(ITEST.EQ.1) GOTO 480
      WRITE(6,476) J,(F1D(I+IDJ),I=1,N)
 476  FORMAT(6H  F(I,,I2,1H)/(3X,1P6E11.3))
 480  CONTINUE
C------------------------------------------------------------------
      RETURN
      END
C
C
C**********************************************************************
      SUBROUTINE BOUND(I1,OUT)
C**********************************************************************
C
      INCLUDE 'comp97.inc'
C
      DIMENSION S1(2),S2(2),S3(2),S4(2),S5(2)
C
C------------------------------------------------------------------
CHAPTER A  A  A  A  A  A  PRELIMINARIES A  A  A  A  A  A  A  A  A  A
C
      KWALL=2-1/I1
      I2=I1+3-2*KWALL
C
      FACTOR=FLOAT(I1/N)
      ZREF=Z(2)+(Z(N)-Z(NM1)-Z(2))*FACTOR
      SQRTK=SQRT(ABS(F(2,JK)+TINY))
      IF(KWALL.EQ.2) SQRTK=SQRT(ABS(F(NM1,JK)+TINY))
      ZPLUS=ZREF*SQRTK/(EMULAM/RHOREF)
      S3(1)=ROULLZ
      S3(2)=ROULHZ
      S4(1)=ABS(FLUXLZ(JRHOU))
      S4(2)=ABS(FLUXHZ(JRHOU))
      S5(1)=ABS(FLUXLZ(JRHOV))
      S5(2)=ABS(FLUXHZ(JRHOV))
      IF(ABS(FLUXLZ(JRHOU)).LE.TAUMIN)S4(1)=TAUMIN
      IF(ABS(FLUXHZ(JRHOU)).LE.TAUMIN)S4(2)=TAUMIN
      IF(ABS(FLUXLZ(JRHOV)).LE.TAUMIN)S5(1)=TAUMIN
      IF(ABS(FLUXHZ(JRHOV)).LE.TAUMIN)S5(2)=TAUMIN
C
C ---CALCULATE SURFACE ROUGHNESS
      TAUB=SQRT(S4(KWALL)**2+S5(KWALL)**2+TINY)
      FRIVEL=SQRT(TAUB/RHOREF+TINY)
      S1(KWALL)=FRIVEL
      IF(S3(KWALL).GT.TINY) GOTO 20
      ZROUGH=EMULAM/RHOREF/C3B/S1(KWALL)
      CCLOG=ALOG(ZREF/ZROUGH)
      S2(KWALL)=CCLOG
      GOTO 21
 20   CONTINUE
      S2(KWALL)=ALOG(ZREF/S3(KWALL))
 21   CONTINUE
      IF(J.NE.JRHOU) GOTO 200
C
C-----------------------------------------------------------------
CHAPTER B  B  B  B  B  B  VELOCITIES  B  B  B  B  B   B  B  B  B  B
C ---TLZ AND THZ FOR X-MOMENTUM
      FRIX=SQRT(ABS(S4(KWALL))/RHOREF+TINY)
      OUT=CAPPA*FRIX/S2(KWALL)
      GOTO 400
 200  IF(J.NE.JRHOV) GOTO 300
C ---TLZ AND THZ FOR Y-MOMENTUM
      FRIY=SQRT(ABS(S5(KWALL))/RHOREF+TINY)
      OUT=CAPPA*FRIY/S2(KWALL)
      GOTO 400
C
C-----------------------------------------------------------------
CHAPTER C  C  C  C  C  C  OTHER DEPENDENT VARIABLES C  C  C  C  C  C
C
 300  OUT=S1(KWALL)/(1./STANTN(J)+S2(KWALL)/CAPPA)
C
C-----------------------------------------------------------------
 400  IF(ZPLUS.LT.11.5) OUT=EMULAM/RHOREF/ZREF/PRL(J)
      IF(ITURBM.EQ.1) OUT=(EMUCON+EMULAM)/RHOREF/ZREF/PRT(J)
      IF(ITEST.EQ.1) GOTO 401
      WRITE(6,4000) J,I1,OUT
 4000 FORMAT(11H WALL TESTS,3H J=,I3,4H I1=,I3,6H OUT= ,E10.3)
 401  CONTINUE
      RETURN
      END
C
C
C*********************************************************************
      SUBROUTINE OUTPUT
C*********************************************************************
C
      INCLUDE 'comp97.inc'
C
      DIMENSION LAB1(10,NPM),OUT(13),LAB2(10,NPM),LABI(13),
     1 XLPLOT(500),YLPLOT(500,10),IOUT(11),LABEL(20),
     2 XTPLOT(NIM),YTAXIS(6),YTPLOT(NIM,6),OUTALL(NIM,20)
      DIMENSION XP(5),YP(5),LABP(10),TIMEP(500),XPART(500,4),
     F YPART(500,4),YPAXIS(4),OUT1(NIM,10),OUT2(NIM,10)
     F,PLABEL(20),KOUT1(NPM),KOUT2(NPM),HCONTI(NPM),SCONTI(NPM)
     F,BFLUXH(NPM),BFLUXS(NPM)
C-----------------------------------------------------------------
```

```
CHAPTER 1  1  1  1  1  1  INITIAL DATA FOR PRINTOUT 1  1  1  1  1  1
C
C ----CROSS-STREAM OUTPUT(PROFILE) DATA
C  --ASSIGN KOUT1/2=NO. OF VARIABLES AND OUTPUT LABELS LAB(K)
     DATA (LABEL(K),K=1,20)/4HUVEL,4HVVEL,4HTEMP,3HSAL,2H1C,
    1 2H2C,2H3C,2H4C,
    2 2HKE,3HDKE,3HEMU,4HSIGM,4HDPDX,4HDPDY,1HW,
    3 4HPRSC,3HRIF,1HN,2HUW,2HVW/
     DATA (PLABEL(K),K=1,20)/4HUVEL,4HVVEL,4HTEMP,3HSAL,2H1C,
    1 2H2C,2H3C,2H4C,
    2 2HKE,3HDKE,3HEMU,4HSIGM,4HDPDX,4HDPDY,1HW,
    3 4HPRSC,3HRIF,1HN,2HUW,2HVW/
     DATA (LAB1(K),K=1,13)/1HZ,4HAREA,4HDZCL,4HUVEL,4HVVEL,4HTEMP,
    1 3HSAL,3HTKE,3HDKE,2HC1,2HC2,2HC3,2HC4/
C
C-----TRAVERSE(CROSS-STREAM) PLOT DATA
C  --ASSIGN NYT=NO. OF VARIABLES TO BE PLOTTED
C  --INSERT DIMENSIONS,ENSURE THAT ITDIM.GE.N AND JTDIM.GE.NYT
     DATA NYT/5/,ITDIM,JTDIM/40,6/
C  --ASSIGN LABELS FOR PLOT AXIS
     DATA XTAXIS/4HZ(I)/
C
     DATA (LABP(K),K=1,9)/4HTIME,2HX1,2HY1,2HX2,2HY2,
    F 2HX3,2HY3,2HX4,2HY4/
     DATA XPAXIS/2H X/
     DATA (YPAXIS(K),K=1,4)/4*1HX/
     DATA ILDIM,JLDIM/500,10/
C-------------------------------------------------------------------
C
C--------NOTE,IN THIS SUBROUTINE X AND Y ARE USED AS COORDINATES
C      FOR THE PLOT-ROUTINES.
C
C-------------------------------------------------------------------
CHAPTER 2  2  2  INITIAL OUTPUT AND CALCULATIONS  2  2 2 2 2 2
C
     IF(ISTEP.NE.0) GOTO 100
     IF(INIOUT)THEN
     WRITE(6,1000)
     WRITE(6,1008)('*',K=1,19)
     IF(NPROBE.GT.1) WRITE(6,1011)IPROBE
     WRITE(6,1001)
     VOLUME=0.
     DO 15 I=2,NM1
     VOLUME=VOLUME+AREA(I)*DZCELL(I)
  15 CONTINUE
     DO 10 J=1,NF
     IOUT(1)=J
     IOUT(2)=0
     IF(SOLVAR(J)) IOUT(2)=1
     IOUT(3)=IKBLZ(J)
     IOUT(4)=IKBHZ(J)
     IOUT(5)=ITRLZ(J)
     IOUT(6)=ITRHZ(J)
     IOUT(7)=IKBOT(J)
     OUT(1)=PRL(J)
     OUT(2)=PRT(J)
     OUT(3)=STANTN(J)
 10  WRITE(6,1003)(IOUT(K),K=1,7),(OUT(K),K=1,3)
     WRITE(6,1005)N,ZDIM,IGRID,ITURBM,INDPX,ITYPEH,LSTEP,
    1 XDIM,INDARE,IPRSC,INDPY,ITYPEL,TLAST,YDIM,VOLUME
    WRITE(6,1010) CPHEAT,BETA,PFILT,RHOREF,RADFRA,
    1 ROULHZ,EMULAM,CORI,ROULLZ
C-----INITIAL PROFILES
C
    WRITE(6,1002)
    WRITE(6,1009)('*',K=1,16)
    WRITE(6,1006)(LAB1(K),K=1,9)
    DO 11 I=N,1,-1
    OUT(1)=Z(I)
    OUT(2)=AREA(I)
    OUT(3)=DZCELL(I)
    OUT(4)=F(I,JRHOU)/RHOREF
    OUT(5)=F(I,JRHOV)/RHOREF
    OUT(6)=F(I,JH)/RHOREF/CPHEAT
    OUT(7)=F(I,JS)
    OUT(8)=F(I,JK)
    OUT(9)=F(I,JD)
  11  WRITE(6,1007)I,(OUT(K),K=1,9)
    WRITE(6,1012)(LAB1(K),K=10,13)
    DO 16 I=N,1,-1
    OUT(10)=F(I,JC1)
    OUT(11)=F(I,JC2)
    OUT(12)=F(I,JC3)
    OUT(13)=F(I,JC4)
16   WRITE(6,1013)I,(OUT(K),K=10,13)
     ENDIF
C-----CALCULATE INITIAL HEAT AND SALINITY CONTENTS
    HCONTI(IPROBE)=0.
    SCONTI(IPROBE)=0.
    BFLUXH(IPROBE)=0.
    BFLUXS(IPROBE)=0.
    DO 12 I=2,NM1
    HCONTI(IPROBE)=HCONTI(IPROBE)+F(I,JH)*DZCELL(I)*AREA(I)
    SCONTI(IPROBE)=SCONTI(IPROBE)+F(I,JS)*DZCELL(I)*AREA(I)
12   CONTINUE
C-----PRELIMINARY CALCULATIONS FOR OUTPUT
    NUMBPR=0
    DO 13 J=1,20
    IF(.NOT.PRPROF(J)) GOTO 14
    NUMBPR=NUMBPR+1
    IF(NUMBPR.LE.10) LAB1(NUMBPR,IPROBE)=LABEL(J)
    IF(NUMBPR.GT.10) LAB2(NUMBPR-10,IPROBE)=LABEL(J)
  14 CONTINUE
  13 CONTINUE
    KOUT1(IPROBE)=MIN(10,NUMBPR)
    KOUT2(IPROBE)=MIN(10,NUMBPR-10)
1000  FORMAT(1H1,19HPRINCIPAL DATA USED)
1011  FORMAT(1X,'IPROBE=',I3)
1001  FORMAT(1H0,5X,'PHI',2X,'SOLVAR',3X,'IKBLZ',3X,'IKBHZ',
    1 3X,'ITRLZ',3X,'ITRHZ',3X,'IKBOT',
    2 4X,'PRL',8X,'PRT',5X,'STANTN')
1002  FORMAT(1H1,18H  INITIAL PROFILES)
1003  FORMAT(1X,I7,6I8,1P5E11.3)
1005  FORMAT(/1X,4H**  ,2HN=,I5,9X,5HZDIM=,1PE10.3,5X,6HIGRID=,I2,3X,
    1 7HITURBM=,I2,3X,6HINDPX=,I2,3X,7HITYPEH=,I2/5X,6HLSTEP=,I5,
    2 5X,5HXDIM=,1PE10.3,5X,7HINDARE=,I2,2X,6HIPRSC=,I2,4X,
    3 6HINDPY=,I2,3X,7HITYPEL=,I2/5X,6HTLAST=,1PE8.1,2X,
    4 5HYDIM=,1PE10.3,5X,7HVOLUME=,1PE10.3)
```

```
1006 FORMAT(1H0,2X,2HI ,A8,8A10)
1012 FORMAT(1X/3X,2HI ,4A9)
1007 FORMAT(1X,I3,1P9E10.2)
1013 FORMAT(1X,I3,1P4E10.2)
1009 FORMAT(1X,2X,16A1)
1008 FORMAT(1X,19A1)
1010 FORMAT(/5X,7HCPHEAT=,1PE10.3,5X,5HBETA=,1PE10.3,
   1 4X,6HPFILT=,1PE10.3/5X,7HRHOREF=,1PE10.3,5X,7HRADFRA=,
   2 1PE10.3,2X,7HROULHZ=,1PE10.3/
   3 5X,7HEMULAM=,1PE10.3,5X,5HCORI=,
   4 1PE10.3,4X,7HROULLZ=,1PE10.3)
    RETURN
100  CONTINUE
C
C-----------------------------------------------------------------------
CHAPTER 3  3  3  3  3  3  COMPUT OUTPUT REQUIRED AT EACH STEP 3  3  3
C
C-----INTEGRATE BOUNDARY FLUXES FOR HEAT AND SALINITY
C ---IN/OUT-FLOWS
    DELH=0.
    DELS=0.
    DO 23 I=2,NM2
    DELH=DELH+QINFL(I)*PHIIN(I,JH)-QOUTFL(I)*F(I,JH)
    DELS=DELS+QINFL(I)*PHIIN(I,JS)-QOUTFL(I)*F(I,JS)
23  CONTINUE
    BFLUXH(IPROBE)=BFLUXH(IPROBE)+NSTPDT(IPROBE)*DT*(FLUXLZ(JH)
   1      *AREA(2)-AREA(NM1)*FLXRAD-FLUXHZ(JH)*AREA(NM1)+DELH)
    BFLUXS(IPROBE)=BFLUXS(IPROBE)+NSTPDT(IPROBE)*DT*(FLUXLZ(JS)
   1      *AREA(2)-FLUXHZ(JS)*AREA(NM1)+DELS)
    IF(INDPT.EQ.0) GOTO 202
C-----PARTICLE TRACKING
C  -- PRELIMINARIES
C
    IF(ISTEP.NE.1) GOTO 200
    DO 20 J=1,INDPT
    XP(J)=0.
    YP(J)=0.
    XPART(1,J)=0.
    YPART(1,J)=0.
20  CONTINUE
200  CONTINUE
C
C  -- NEW COORDINATES
C
    DO 21 J=1,INDPT
    IP=ILEVEL(J)
    XP(J)=XP(J)+NSTPDT(IPROBE)*DT*F(IP,JRHOU)/RHOREF
    YP(J)=YP(J)+NSTPDT(IPROBE)*DT*F(IP,JRHOV)/RHOREF
21  CONTINUE
C
C --- SAVE COORDINATES
C
    IF(MOD(ISTEP,IPSAVE).NE.0) GOTO 201
    ILP=ISTEP/IPSAVE
    TIMEP(ILP)=TIME
    DO 22 J=1,INDPT
    XPART(ILP,J)=XP(J)
    YPART(ILP,J)=YP(J)
22  CONTINUE
```

```
201  C●NTINUE
C
202  CONTINUE
C
C-----TESTS FOR PRINTOUT
C     ---IPRINT=1 GIVES SINGLE(STATION) VARIABLES
C     ---IPRINT=2 ADDS THE ARRAY(PROFILE) VARIABLES
C     ---IPRINT=3 ADDS CROSS-STREAM PLOTS
      IPRINT=0
      IF(MOD(ISTEP,NSTAT).EQ.0) IPRINT=1
      IF(MOD(ISTEP,NPROF).EQ.0) IPRINT=2
      IF(ISTEP.EQ.0.OR.ITPLOT.EQ.1) GOTO 1020
      IF(MOD(ISTEP,NPLOT).EQ.0.AND.ISTEP.NE.0.OR.ITEST.NE.1
     1 .OR.IFIN.NE.1) IPRINT=3
C
 1020 IF(IPRINT.EQ.0) RETURN
C
C-----------------------------------------------------------------------
CHAPTER 4  4  4  4  4  4  STATION VARIABLES 4  4  4  4  4  4  4  4  4
C
C-----CALCULATE HEAT AND SALINITY CONTENTS
      HCONT=0.0
      SCONT=0.0
      VOLUME=0.0
C
      DO 30 I=2,NM1
      VOLUME=VOLUME+AREA(I)*DZCELL(I)
      HCONT=HCONT+F(I,JH)*DZCELL(I)*AREA(I)
      SCONT=SCONT+F(I,JS)*DZCELL(I)*AREA(I)
   30 CONTINUE
      WRITE(6,3000) ISTEP,DPDX(NM1),FLXRAD,DT,TIME,DPDY(NM1)
      IF(MOVE) WRITE(6,3008) ZDIM,VOLUME
      IF(NPROBE.GT.1) WRITE(6,3007)IPROBE
      WRITE(6,3004) HCONTI(IPROBE),BFLUXH(IPROBE),HCONT,
     1SCONTI(IPROBE),BFLUXS(IPROBE),SCONT
      WRITE(6,3001)
      WRITE(6,3002)(FLUXHZ(K),K=1,6)
      WRITE(6,3003)(FLUXLZ(K),K=1,6)
      WRITE(6,3009)
      WRITE(6,3010)(FLUXHZ(K),K=7,NJM)
      WRITE(6,3011)(FLUXLZ(K),K=7,NJM)
3000  FORMAT(//1HX,4H**  ,7H ISTEP=,I5,9X,10HDPDX(NM1)=,1PE10.3,5X,
     1 7HFLXRAD=,1PE10.3,5X,3HDT=,1PE10.3/6X,5HTIME=,1PE10.3,
     2 5X,10HDPDY(NM1)=,1PE10.3)
3001  FORMAT(14X,'XMOM',7X,'YMOM',7X,'HEAT',7X,'SALT',
     1 8X,'TKE',8X,'DKE')
3002  FORMAT(1X,5X,'FLUXHZ',1P6E11.3)
3003  FORMAT(1X,5X,'FLUXLZ',1P6E11.3)
3009  FORMAT(1X/14X,'CONC.1',5X,'CONC.2',5X,'CONC.3',5X,'CONC.4')
3010  FORMAT(1X,5X,'FLUXHZ',1P4E11.3)
3011  FORMAT(1X,5X,'FLUXLZ',1P4E11.3)
3004  FORMAT(1H0,5X,20HINTEGRAL CHECKS      ,
     1 17HINIT. HEAT-CONT.=,1PE11.3,20HINTEGR. BOUND. FLUX=,1PE11.3,
     2 16HPRESENT H-CONT.=,1PE11.3,/
     3 1X,5X,20H***************     ,
     4 17HINIT. SALT-CONT.=,1PE11.3,20HINTEGR. BOUND. FLUX=,1PE11.3,
     5 16HPRESENT S-CONT.=,1PE11.3/1X)
3005  FORMAT(1H0,5X,'BOUNDARY FLUXES')
3007  FORMAT(1X,5X,'IPROBE=',I3)
```

```
3008  FORMAT(1X,5X,'ZDIM=',1PE10.3,5X,'VOLUME=',1PE10.3)
C
C-------------------------------------------------------------------
CHAPTER 5  5  5  5  5  5  CROSS-SRTEAM PROFILES 5  5  5  5  5  5  5  5
C
      IF(IPRINT.EQ.1) GOTO 1050
C
C-----CALCULATE ALL OUTPUT
      DO 40 I=1,N
C  ---VERTICAL VELOCITIES
      WQ=0.
      IF(ABS(QZ(I)).LT.TINY) GOTO 45
      WQ=QZ(I)/(AREA(I)+TINY)
 45   CONTINUE
      OUTALL(I,1)=F(I,JRHOU)/RHOREF
      OUTALL(I,2)=F(I,JRHOV)/RHOREF
      OUTALL(I,3)=F(I,JTE)
      OUTALL(I,4)=F(I,JS)
      OUTALL(I,5)=F(I,JC1)
      OUTALL(I,6)=F(I,JC2)
      OUTALL(I,7)=F(I,JC3)
      OUTALL(I,8)=F(I,JC4)
      OUTALL(I,9)=F(I,JK)
      OUTALL(I,10)=F(I,JD)
      IF(I.EQ.1.OR.I.EQ.N) GOTO 401
      OUTALL(I,11)=F(I,JEMU)+EMULAM
      OUTALL(I,12)=RHO(I)-1000.
      OUTALL(I,13)=DPDX(I)
      OUTALL(I,14)=DPDY(I)
      OUTALL(I,15)=WQ
      OUTALL(I,16)=PRSCNU(I)
      IF(I.LE.2.OR.I.GE.NM1) GOTO 401
      DRHODZ=(RHO(I+1)-RHO(I-1))*RECDZ(I)
      DUDZ=(F(I+1,JRHOU)-F(I-1,JRHOU))*RECDZ(I)/RHO(I)
      DVDZ=(F(I+1,JRHOV)-F(I-1,JRHOV))*RECDZ(I)/RHO(I)
      OUTALL(I,17)=-BUO(I)/(GRADSQ(I)+TINY)
      OUTALL(I,18)=SQRT(AMAX1(TINY,-AGRAV/RHOREF*DRHODZ))
      OUTALL(I,19)=-F(I,JEMU)*DUDZ/RHO(I)
      OUTALL(I,20)=-F(I,JEMU)*DVDZ/RHO(I)
401   CONTINUE
40    CONTINUE
C
C  ---MODIFY OUTPUT ACCORDING TO BOUNDARY CONDITIONS.
      OUTALL(N,15)=0.
      IF(ITYPEH.NE.2) OUTALL(N,9)=0.
      IF(ITYPEH.NE.2) OUTALL(N,10)=0.
      IF(ITYPEL.NE.2) OUTALL(1,9)=0.
      IF(ITYPEL.NE.2) OUTALL(1,10)=0.
      IF(ITYPEH.EQ.2) OUTALL(N,11)=OUTALL(N-1,11)
      IF(ITYPEL.EQ.2) OUTALL(1,11)=OUTALL(2,11)
      IF(ITYPEH.NE.2) OUTALL(N,11)=0.
      IF(ITYPEL.NE.2) OUTALL(1,11)=0.
C
      DO 41 I=1,N
      NUMBPR=0
      DO 42 J=1,20
      IF(.NOT.PRPROF(J)) GOTO 49
      NUMBPR=NUMBPR+1
      IF(NUMBPR.LE.10) OUT1(I,NUMBPR)=OUTALL(I,J)
```

```
     IF(NUMBPR.GT.10) OUT2(I,NUMBPR-10)=OUTALL(I,J)
 49  CONTINUE
 42  CONTINUE
 41  CONTINUE
C
     WRITE(6,1099) (LAB1(K,IPROBE),K=1,KOUT1(IPROBE))
     DO 46 I=N,1,-1
 46  WRITE(6,1098) I,Z(I),(OUT1(I,K),K=1,KOUTI(IPROBE))
     IF(NUMBPR.LE.10) GOTO 47
     WRITE(6,1099) (LAB2(K,IPROBE),K=1,KOUT2(IPROBE))
     DO 48 I=N,1,-1
 48  WRITE(6,1098) I,Z(I),(OUT2(I,K),K=1,KOUT2(IPROBE))
 47  CONTINUE
C
     IF(IPRINT.LT.3.OR.ITPLOT.EQ.1) GOTO 1050
C-----ASSIGN CROSS-STREAM PLOTS
     DO 402 I=1,N
402  XTPLOT(I)=Z(I)
     NUMBPR=0
     DO 44 J=1,20
     IF(.NOT.PLPROF(J)) GOTO 400
     NUMBPR=NUMBPR+I
     DO 43 I=1,N
43   YTPLOT(I,NUMBPR)=OUTALL(I,J)
     YTAXIS(NUMBPR)=PLABEL(J)
400  CONTINUE
44   CONTINUE
     NYT=NUMBPR
C  --CROSS-STREAM PLOT OUTPUT
     WRITE(6,1096) TIME,ISTEP
 1096 FORMAT(18H1CROSS-STREAM PLOT,
    1 6H TIME=,1PE10.3,7H ISTEP=,I4)
     CALL PLOTLP(XTPLOT,ITDIM,N,XTAXIS,YTPLOT,JTDIM,NYT,YTAXIS)
C
C-----------------------------------------------------------------
CHAPTER 6 6 6 6 6 6 RETURN OR TERMINATE 6 6 6 6 6 6 6 6
C
 1050 IF(IFIN.EQ.1) RETURN
C-----PARTICLE TRACKING OUTPUT
     IF(INDPT.EQ.0) RETURN
     WRITE(6,500)(LABP(K),K=1,9)
     DO 50 I=1,ILP
     OUT(1)=TIMEP(I)
     OUT(2)=XPART(I,1)
     OUT(3)=YPART(I,1)
     OUT(4)=XPART(I,2)
     OUT(5)=YPART(I,2)
     OUT(6)=XPART(I,3)
     OUT(7)=YPART(I,3)
     OUT(8)=XPART(I,4)
     OUT(9)=YPART(I,4)
50   WRITE(6,501)I,(OUT(K),K=1,9)
500  FORMAT(1H0,2X,2HI ,9A11)
501  FORMAT(1X,I3,1P9E11.3)
     DO 51 J=1,INDPT
     DO 52 I=1,ILP
     XLPLOT(I)=XPART(I,J)
     YLPLOT(I,1)=YPART(I,J)
 52  CONTINUE
```

```
      WRITE(6,502) J
502   FORMAT(12HPARTICLE NR.,I3)
      CALL PLOTLP(XLPLOT,ILDIM,ILP,XPAXIS,YLPLOT,JLDIM,1,YPAXIS)
51    CONTINUE
      RETURN
 1098 FORMAT(1X,I3,1P11E11.3)
 1099 FORMAT(1H0,2X,2HI ,6X,1HZ,10A11)
C
      END
C
C
C*****************************************************************************
      SUBROUTINE PLOTLP(X,IDIME,IMAX,XAXIS,Y,JDIME,JMAX,YAXIS)
C******************************************************************************
C
C  SUBROUTINE FOR PLOTTING J CURVES OF Y(I,J) AGAINST X(I).
C
C  X AND Y ARE SCALED TO THE RANGE 0. TO 1., FOR PLOTTING AS
C    (Y-YMIN)/(YMAX-YMIN), THE MAXIMUM AND MINIMUM VALUES ARE PRINTED
C    N.B. THE X AND Y ARRAYS MUST BE REDEFINED BEFORE EACH CALL PLOTS.
C  IDIME IS THE VARIABLE DIMENSION FOR X.
C  IMAX IS THE NUMBER OF X VALUES.
C  XAXIS STORES THE NAME OF THE X-AXIS.
C  JDIME IS THE VARIABLE DIMENSION FOR Y.
C  JMAX IS THE NUMBER OF CURVES TO BE PLOTTED, (UP TO 30).
C  THE ARRAY YAXIS(J) STORES THE NAMES OF THE CURVES,
C    THE FIRST CHARACTER OF EACH CURVE-NAME IS USED FOR PLOTTING.
C  XSIZE ALTERS THE X-PLOT SIZE BY A FACTOR OF .2 TO 1., IN STEPS OF
C  YSIZE IS THE Y-PLOT SIZE FACTOR OF .2 UPWARDS IN STEPS OF .2
C    XSIZE=1., YSIZE=1. GIVES NORMAL SIZE PLOT.
C
      DIMENSION X(IDIME),Y(IDIME,JDIME),YAXIS(JDIME),
     1 A(101),YMAX(30),YMIN(30),DIGIT(11)
      EQUIVALENCE (YMAX(1),A(1)),(YMIN(1),A(31))
      DATA  DOT,CROSS,BLANK/1H.,1H+,1H /
     1,DIGIT/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H1/
C***** SET PLOT SIZE FACTORS
      XSIZE=0.6
      YSIZE=0.6
C***** SCALING X-ARRAY TO RANGE 0 TO 100*XSIZE
      XR=100.*XSIZE
      XMAX=-1.E30
      XMIN=+1.E30
      IM=IMAX
      DO 1 I=1,IM
      XMAX=AMAX1(XMAX,X(I))
    1 XMIN=AMIN1(XMIN,X(I))
      S=XR/(XMAX-XMIN+1.E-30)
      DO 2 I=1,IM
    2 X(I)=(X(I)-XMIN)*S
C***** SCALING Y-ARRAY TO RANGE 0 TO 50*YSIZE
      YR=50.*YSIZE
      JM=JMAX
      DO 4 J=1,JM
      YMAX(J)=-1.E30
      YMIN(J)=+1.E30
      DO 4 I=1,IM
      YMIN(J)=AMIN1(YMIN(J),Y(I,J))
    4 YMAX(J)=AMAX1(YMAX(J),Y(I,J))
```

```
C   COMP97.INC
C
     PARAMETER (NIM=100,NJM=30,NPM=30)
     PARAMETER (NSTORE=9911)
     PARAMETER (NJMP1=NJM+1,NJMP2=NJM+2,NJMM4=NJM-4,
    A NJMM6=NJM-6,NJP2NI=NIM*(NJM+2),NIMNJM=NIM*NJM,
    B NSTOR1=9804,NSTOR2=107)
C
     COMMON
c    COMMON/COM1/
C-----ARRAYS
C
    A AREA(NIM),BUO(NIM),DIF(NIM),DIFREF(NIM),DPDX(NIM),DPDY(NIM),
    D DZ(NIM),DZCELL(NIM),DZCREF(NIM),EMU(NIM),PREF(NIM),
    D F(NIM,NJMP2),FLUXLZ(NJM),FLUXHZ(NJM),FW(NIM),GRADSQ(NIM),
    I  IKBLZ(NJM),IKBHZ(NJM),ITRLZ(NJM),ITRHZ(NJM),IKBOT(NJM),
    N ILEVEL(4),NSTPDT(NPM),PHIIN(NIM,NJM),PRPROF(20),PLPROF(20),
    P PRSCNU(NIM),PRL(NJM),PRT(NJM),PHIQLZ(NJM),PHIQHZ(NJM),
    Q QINFL(NIM),QOUTFL(NIM),QZ(NIM),RECDZ(NIM),
    R RHO(NIM),SI(NIM),SIP(NIM),SOLVAR(NJM),STANTN(NJM),
    T TFRAC(20),URUP(NIM),URUPUP(NIM),V1LZ(NJM),V2LZ(NJM),
    T V3LZ(NJM),V4LZ(NJM),V5LZ(NJM),V1HZ(NJM),V2HZ(NJM),V3HZ(NJM),
    V V4HZ(NJM),V5HZ(NJM),VST1(NJM),VST2(NJM),Z(NIM),ZBOUND(NIM),
    Z ZST1(NJM),ZST2(NJM),ZBREF(NIM),ZSREF(NIM)
C
    COMMON/COM2/
C-----VARIABLES
    A AGRAV,AREAHZ,BETA,CAPPA,CORI,CPHEAT,
    C CD,CD75,C1,C2,C3,C1PR,C2PR,C3PR,CEXPG,CEXPA,C3B,
    C CKSURF,C1RHO,C2RHO,C3RHO,C4RHO,C5RHO,C6RHO,
    D DT,DPDXP,DPDYP,DQ1,DQ2,EMTMIN,EMUCON,EMULAM,FLXRAD,
    F FACTHZ,FACTLZ,FKMIN,FDMIN,GREAT,ITYPEF,ITYPEH,ITYPEL,
    I IDIMF,IFIN,ILPLOT,IPROBE,ISTEP,ITEST,ITPLOT,ITURBM,INDPX,
    I INDPY,IPRSC,IGRID,INDARE,INDPT,IPSAVE,ISTPR,INIOUT,
    J J,JRHOU,JRHOV,JH,JS,JK,JD,JC1,JC2,JC3,JC4,JEMU,JTE,KINDAV,
    L LSTEP,N,NM1,NM2,NF,NFP2,NSTAT,NPROF,NPLOT,NPROBE,MOVE,
    P PI,PFILT,PREEVA,QSURF,RADFRA,RHOREF,RTCD,RHOUP,RHOVP,
    P ROULLZ,ROULHZ,SRAD,TAUMIN,TU,TINY,TREF,TLAST,TQ1,TQ2,TIME,
    X XDIM,YDIM,ZDIM,ZSSTRT
C
    LOGICAL SOLVAR,PLPROF,PRPROF,MOVE,INIOUT
```

SMHI

Sveriges meteorologiska och hydrologiska institut
601 76 Norrköping
Tel 011-495 80 00. Fax 011-495 80 01

ISSN 0283-1112