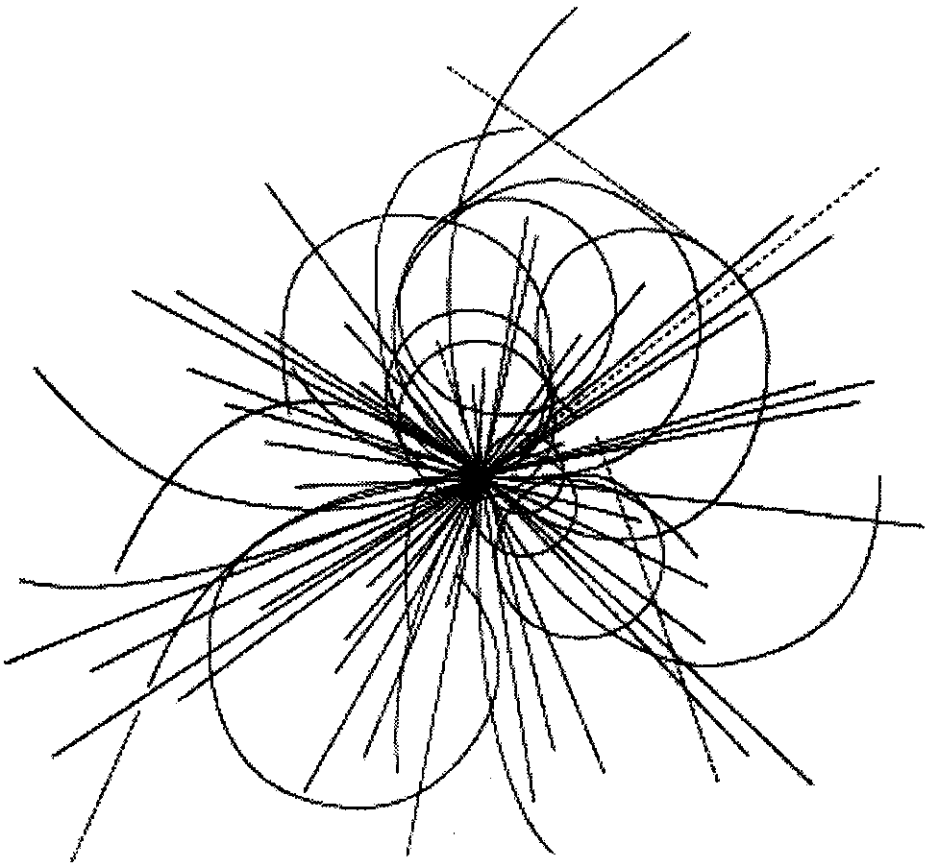
SSCL-N-850

SSCL-N-850
August 1993
Distribution Category: 400

R. Carcagno
W. Schiesser
A. Yücel

Prototype Helium Flow and Heat Transfer Model and Code



Superconducting Super Collider Laboratory

Disclaimer Notice

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government or any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Superconducting Super Collider Laboratory is an equal opportunity employer.

SSCL-N-850

Prototype Helium Flow and Heat Transfer Model and Code*

R. Carcagno, W. Schiesser, and A. Yücel

Superconducting Super Collider Laboratory
2550 Beckleymeade Ave.
Dallas, TX 75237 USA

August 1993

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

(1) Introduction

The equations of fluid flow and heat transfer discussed in this report are applied to problems in the transient flow of helium. The objective of this analysis is to develop a modeling and computational system that can be applied to a spectrum of problems in helium flow (or, more generally, fluid flow with heat transfer).

The approach is to first require the user to write down a descriptive set of equations for the problem system of interest. These equations will typically be ordinary differential equations (ODEs) and partial differential equations (PDEs) in time, i.e., dynamic ODE/PDEs; algebraic equations can also be included so that differential-algebraic equations (DAEs) can also be part of the model.

This equation specification is facilitated through the application of a generic system of ODE/PDEs that usually can be applied directly to the problem system of interest, or can directly be extended for application to the problem system. The generic equations are based on the three fundamental principles of conservation of mass (i.e., the continuity balance), momentum (Newton's second law applied to fluids) and energy; the state variables computed from these three types of equations are fluid density, mass flow rate and volumetric internal energy, respectively. Other variables can then be computed from the state variables, usually through an equation of state for helium, e.g., pressure, temperature and enthalpy.

Next, the user is required to code the model equations in a series of Fortran subroutines; these routines are then combined with a series of library routines and a library main program to make up a complete code that can be compiled and executed to compute numerical solutions to the model equations. This process of coding the equations for the problem system is facilitated by the availability of a framework that will accommodate most problems, and by the availability of two example programs that most likely can be applied rather directly to the problem system of interest (with possibly some straightforward modifications and extensions).

Finally, the user is required to execute the code and evaluate the numerical solutions that are produced. This is generally no less difficult than the two preceding steps of model formulation and coding; the evaluation of the model solutions generally requires thorough testing for spatial and time convergence, and critical analysis from the perspective of physical (engineering) principles. Clearly, a solution must make sense mathematically (computationally) and physically.

We start the discussion of the analysis system with the general PDEs, and then illustrate the use of the system in terms of some applications.

(2) Basic PDEs

We restrict the discussion of the model PDEs expressing conservation of mass, momentum and energy to the one-dimensional case.

(2.1) Continuity Equation

The mass (continuity) balance is

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho v)}{\partial t} = 0 \tag{1}$$

where

ρ	fluid density (kg/m^3)
v	fluid linear velocity (m/s)
t	time (s)
z	position along the system (m)

Because of the usual procedure of expressing helium flow in mass units, we use in place of v the helium mass flow rate W (kg/s-m^2)

$$W = \rho v \tag{2}$$

Eq. (1) written in terms of W is then

$$\frac{\partial \rho}{\partial t} + \frac{\partial W}{\partial t} = 0 \tag{3}$$

Eq. (3) is the basic continuity balance.

(2.2) Momentum Equation

The starting point for the momentum balance is

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial(\rho v v)}{\partial z} + \frac{\partial \tau_{zz}}{\partial t} + \frac{\partial P}{\partial z} - \rho g_z = 0 \tag{4}$$

where

τ_{zz}	component of the stress tensor giving the shear force in the z direction due to flow in the z direction (N/m^2)
P	fluid pressure (N/m^2)
g_z	gravitational acceleration in the z direction (m/s^2)

Eq. (4), which is a precursor of the Navier-Stokes equations, is just a statement of Newton's second law

$$\frac{dm}{dt} = \sum F \tag{5}$$

Thus, $\partial(\rho v)/\partial t$ in eq. (4) is the time rate of change of momentum (ρv is the volumetric momentum). $\partial(\rho vv)/\partial z$ is the force due to the net convection of momentum in the z direction. $\partial\tau_{zz}/\partial z$ is the frictional force. $\partial P/\partial z$ is the pressure force and $-\rho g_z$ is the gravitational force.

Eq. (4) written in terms of W is

$$\frac{\partial W}{\partial t} + \frac{\partial(W^2/\rho)}{\partial z} + \frac{\partial\tau_{zz}}{\partial t} + \frac{\partial P}{\partial z} - \rho g_z = 0 \tag{6}$$

where

f	friction factor (dimensionless)
D_c	channel diameter (m)

Eq. (6) is the basic momentum balance. A friction factor is used to compute the frictional force because a more fundamental approach is generally not available, particularly in the case of turbulent flow.

Before proceeding to the energy balance, we consider an important characteristic of eq. (4) (and thus, also, of eq. (6)). If the first two LHS terms of terms of eq. (4) are expanded (through differentiation of products), we get

$$\rho\frac{\partial v}{\partial t} + v\frac{\partial\rho}{\partial t} + v\frac{\partial(\rho v)}{\partial z}\ \rho v\frac{\partial v}{\partial z} + \frac{\partial\tau_{zz}}{\partial t} + \frac{\partial P}{\partial z} - \rho g_z = 0$$

or

$$\left\{\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial z}\right\} + (v/\rho)\left\{\frac{\partial\rho}{\partial t} + \frac{\partial(\rho v)}{\partial z}\right\} + (1/\rho)\frac{\partial\tau_{zz}}{\partial t} + (1/\rho)\frac{\partial P}{\partial z} - g_z = 0 \tag{7}$$

The second term in brackets is zero as a consequence of eq. (1). The group of terms in the first bracket

is the *nonlinear advection group*, which can produce front flattening and front sharpening in the numerical solution of eq. (4) or (6) [Schiesser (1994)]. We merely intend to point out this property of eq. (4) since it can have important consequences for calculating numerical solutions, particularly in the front sharpening case.

(2.3) Energy Equation

The energy equation is

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial(Wh)}{\partial z} + \dot{Q} - k\frac{\partial^2 T}{\partial z^2} = 0 \tag{8}$$

where

u	fluid mass internal energy (j/kg)
h	fluid mass enthalphy (j/kg)
$\dot{Q}$	volumetric heat transfer (j/m^3-s)
k	axial thermal conductivity (w-m/m^2-k)

Eqs. (3), (6) and (8) are the basic PDEs that must be integrated numerically in the analysis of flowing helium; they produce ρ, W and ρu as solutions (which are the dependent variables that appear in the temporal derivatives of eqs (3), (6) and (8)). Auxiliary equations are then added to these equations to complete the definition of the model as described next.

(2.4) Equation of State

The helium mass internal energy, u, is computed simply as

$$u = \rho u/u \tag{9}$$

where ρu and ρ come from eqs. (3) and (8) respectively. Then the helium thermodynamic equation of state is used to calculate the helium temperature, pressure and enthalpy

$$P = P(\rho,u) \tag{10}$$

$$T = T(\rho,u) \tag{11}$$

$$h = h(\rho, u) \tag{12}$$

Additionally, eqs. (3), (6) and (8) require initial and boundary conditions, which will be determined by the requirements of the parrticular problem system that is to be analyzed. Therefore, the discussion of these auxiliary conditions will be delayed until specific applications are considered next.

(3) Transients in a Uniform Channel

To illustrate the use of the preceding equations, and the associated Fortran coding, we consider the relatively simple problem of helium dynamics in a uniform channel, as illustrated in Figure 1a. Eqs. (3), (6) and (8) are applied to the helium in this channel. If the valves at the two ends of the channel are identical, and the transient conditions are the same on the external ends of the valves (away from the channel), then the solution to the model equations for the helium in the channel is

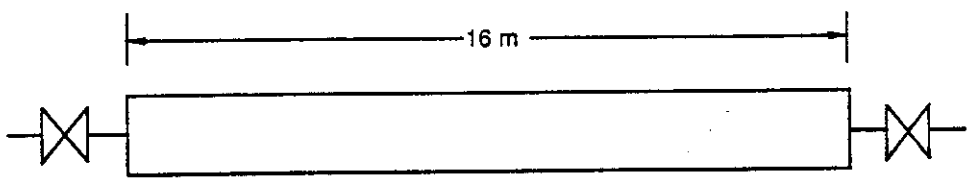


Figure 1a: Uniform Channel with Valve Terminations

symmetrical with respect to the centerpoint of the channel. Therefore, the solution to the model equations is required for only half the channel, as depicted in Figure 1b

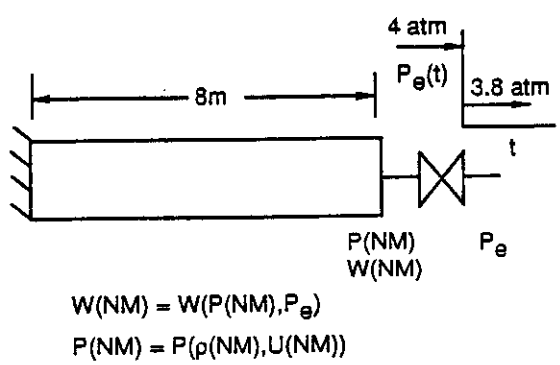


Figure 1b: Application of Symmetry to the Uniform Channel

(3.1) Momentum Equation

We now consider the solution of eq. (6) applied to the system of Figure 1b, starting with the boundary conditions. At the left end, the flow is zero (due to symmetry at the midpoint in Figure 1a)

(1) Left end, $z = z_l$:

$$W(z_l,t) = 0 \tag{13}$$

At the right end of the system of Figure 1b, the flow through the valve serves as the boundary condition for eq. (6)

(2) Right end, $z = z_r$:

First the pressure drop across the right valve in Figure 1b is calculated

$$\Delta P = P(z_r,t) - P_e(t) \tag{14}$$

where P_e is the right end pressure. This pressure drop then sets the flow out of the right end of the channel

$$\Delta P > 0,\ W(z_r,t) = W_{max}\ (1 - e^{-(t/\tau_v)})\sqrt{\Delta P} \tag{15}$$

$$\Delta P < 0,\ W(z_r,t) = -W_{max}\ (1 - e^{-(t/\tau_v)})\sqrt{|\Delta P|} \tag{16}$$

where

W_{max} maximum mass flow rate for a given pressure drop, ΔP, (kg/m^2-s)

τ_v time constant for the change in $P_e(t)$ (s)

Eq. (6) requires an initial condition

$$W(z,0) = 0 \tag{17}$$

i.e., the helium starts from rest. Then the helium is set in motion by a step change in the valve downstream pressure, P_e, from 4 atm to 3.8 atm at $t = 0$, as depicted in Figure 1b.

Of course, the solution of eq. (6) also requires the simultaneous solution of eq. (3) and eq. (8).

However, we consider first the coding of eq. (6), then go on to the final specification and coding of eqs. (3) and (8).

The temporal derivatives of eqs. (3), (6) and (8) are programmed in subroutine DERV, which is called as part of a method of lines (MOL) code. The initial conditions for these temporal derivatives are programmed in subroutine INITAL, and the output (numerical solution of eqs. (3), (6) and (8)) is printed and plotted in subroutine PRINT. We now consider the coding in subroutine DERV.

The state variables for the model (the dependent variables in the temporal derivatives of eqs. (3), (6) and (8)) are ρ, W and ρu. These state variables appear in COMMON/Y/ of the following COMMON block

```
C...
C...  DEFINE THE CONTINUITY, MOMENTUM AND ENERGY BALANCES ON A SPATIAL
C...  GRID OF NM POINTS.  UNITS ARE MKS (SI) EXCEPT FOR PRESSURE WHICH
C...  IS IN ATM.
C...
C...     COMMON/T/
C...
C...        T          INDEPENDENT VARIABLE, TIME (S)
C...
C...        NSTOP      STOPPING VARIABLE (SET TO A NONZERO VALUE WILL
C...                   TERMINATE THE CURRENT RUN)
C...
C...        NORUN      RUN COUNTER (SET TO 1 DURING THE FIRST RUN, 2
C...                   DURING THE SECOND RUN, ETC., AND CAN THEREFORE
C...                   BE USED TO CHANGE THE CODE FROM RUN TO RUN)
C...
C...     COMMON/Y/
C...
C...        WM         MASS FLOW RATE (KG/M**2-S)
C...
C...        RHOM       FLUID MASS DENSITY (KG/M**3)
C...
C...        UMG        FLUID INTERNAL ENERGY-DENSITY PRODUCT (J/M**3)
C...
C...     COMMON/F/
C...
C...        WMT        TEMPORAL DERIVATIVE OF WM, I.E., THE FLUID MOMEN-
C...                   TUM PER UNIT VOLUME (KG/M**2-S**2)
C...
C...        RHOMT      TEMPORAL DERIVATIVE OF RHO (KG/M**3-S)
C...
C...        UMGT       TEMPORAL DERIVATIVE OF UMG (J/M**3-S)
C...
C...     COMMON/V/
C...
```

```
C...        UM        FLUID INTERNAL ENEREGY PER UNIT MASS (J/KG)
C...
C...        PM        FLUID PRESSURE (ATM)
C...
C...        TM        FLUID TEMPERATURE (K)
C...
C...        HM        FLUID ENTHALPY PER UNIT MASS (J/KG)
C...
C...     COMMON/C/
C...
C...        ZL        LEFT HAND SPATIAL COORDINATE (M)
C...
C...        ZR        RIGHT HAND SPATIAL COORDINATE (M)
C...
C...        DZ        GRID SPACING (M)
C...
C...        Z         SPATIAL GRID (RUNNING FROM ZR TO ZL WITH SPACING
C...                  DZ) (M)
C...
C...        TMI       INITIAL FLUID TEMPERATURE (K)
C...
C...        PMI       INITIAL FLUID PRESSURE (ATM)
C...
C...        WMI       INITIAL FLUID.MASS FLOW RATE (KG/M**2-S)
C...
C...        QM        VOLUMETRIC HEAT INPUT TO THE FLUID ALONG THE
C...                  SPATIAL GRID (J/M**3-S)
C...
C...     COMMON/I/
C...
C...        IP        INTEGER COUNTER USED FOR PRINTING AND PLOTTING
C...                  IN SUBROUTINE PRINT
C...
C...     COMMON/IMODEL/
C...
C...        IMODEL    INTEGER VARIABLE TO SELECT THE THERMODYNAMIC
C...                  MODEL (ALLOWABLE VALUES ARE 1 TO 4)
C...
      PARAMETER (NM=33)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
```

The corresponding temporal derivatives of these state variables appear in COMMON/F/. Thus,

subroutine DERV computes the temporal derivative vector in COMMON/F/, and the ODE integrator (in this case, RKF45) returns the dependent variable vector in COMMON/Y/. Time is the first element in COMMON/T/ (and is provided by the ODE integrator as an input to DERV). Thus, the required COMMON areas for this MOL code are /T/, /Y/ and /F/. The other COMMON areas are optional; their purpose and contents are explained in the preceding comments.

Thus, at the beginning of DERV, we can assume the dependent (state) variables in COMMON/Y/ are available for programming (again, they are provided by the ODE integrator). We first perform the algebra of eqs. (9) to (12)

```
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
C...  FOR THE MAGNETS
C...
C...  THE THERMODYNAMIC FUNCTIONS ARE
C...
C...     PRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               PRESSURE
C...
C...     TRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               TEMPERATURE
C...
C...     HRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               ENTHALPY
      DO 1 I=1,NM
         UM(I)=UMG(I)/RHOM(I)
         PM(I)=PRHOU(RHOM(I),UM(I))
         TM(I)=TRHOU(RHOM(I),UM(I))
         HM(I)=HRHOU(RHOM(I),UM(I))
1     CONTINUE
```

The four lines in DO loop 1 implement eqs. (9) to (12), respectively. PRHOU, TRHOU and HRHOU perform the thermodynamic flashes of eqs. (10) to (12), respectively, by calling the Air Products (APCI) model 3 thermodynamic correlation for helium. For example, function PRHOU is

```
      DOUBLE PRECISION FUNCTION PRHOU(RHO,U)
C...
C...  PRESSURE FROM DENSITY AND INTERNAL ENERGY
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=6
      kd=0
```

```
      kh=0
      ku=0
      ks=0
      kcp=0
C...
C...  CONVERT DENSITY IN KG/M**3 TO GM-MOL/LITER
      dt=RHO/4.0d0
C...
C...  CONVERT INTERNAL ENERGY IN J/KG TO J/GM-MOL
      ut=U*4.00d0/1000.0d0
      call thermo(      t,      p,    vof,  ifase, ifcode,
     +                 kd,     kh,     ku,     ks,    kcp,
     +                 dt,     dv,     dl,
     +                 ht,     hv,     hl,
     +                 ut,     uv,     ul,
     +                 st,     sv,     sl,
     +                        cpv,    cpl,
     +              imodel,  ipert,  itdbg,    ker)
      PRHOU=p
      RETURN
      END
```

Function PRHOU will not be considered in detail. The essential feature is the call to subroutine thermo with imodel = 3 (APCI model 3 for helium). Subroutine thermo has an extensive set of documentation comments describing its arguments. Independent testing of subroutine thermo indicated that it gives the properties of helium with sufficient accuracy for the dynamic models we are considering here. All of the functions called for the evaluation of thermodynamic properties are listed in Appendix 1f.

The heat input term of eq. (8), $\dot{Q}$, is programmed next in DERV (actually, it is included in DO loop 1 discussed previously, right after the programming of HM(I))

```
C...
C...  THE HEAT INPUT CAN BE SPECIFIED AS A FUNCTION OF POSITION Z (I.E.,
C...  GRID INDEX I)
         IF((I.GE.1).AND.(I.LE.NM-5))THEN
C...
C...        THE FOLLOWING Q IS A FUNCTION OF TIME, T, WITH A TIME
C...        CONSTANT OF 0.5 S AND THE TEMPERATURE DIFFERENCE (10 - TM).
C...        ANOTHER POSSIBILITY WOULD BE TO INCLUDE A HEAT CONDUCTION
C...        MODEL TO DEFINE Q
C...        QM(I)=1.0D+05*(1.0D0-DEXP(-T/0.5D0))*(10.0D0-TM(I))
            QM(I)=0.0D0
         ELSE IF(I.GT.NM-5)THEN
            QM(I)=0.0D0
         END IF
```

The comments illustrate how the heat input can be programmed as a function of z along the system of Figure 1b; specifically, $\dot{Q}$ is programmed as a function of t with a 0.5 s time constant and a temperature difference of (10 - T) for grid points 1 to NM - 5 where NM is the total number of grid points in z. However, when the program was executed, $\dot{Q}$ was set to zero for all of the grid points.

The boundary conditions for eq. (6), eqs. (13) to (16) are programmed next

```
C...
C...  BOUNDARY CONDITIONS
C...
C...  LEFT END, Z = ZL
         PLEND=4.0D0
C...
C...     PRESSURE DROP
         DP=PM(1)-PLEND
         IF(DP.LE.0.0D0)THEN
C...
C...        FLOW RATE FOR NEGATIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS POSITIVE, AND PROPORTIONAL TO THE SQUARE ROOT OF THE
C...        PRESSURE DROP.  THE FLOW RATE ALSO CHANGES DYNAMICALLY
C...        WITH A TIME CONSTANT OF 0.01 S
C...        WM(1)= 300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DABS(DP))
            WM(1)=0.0D0
         ELSE
C...
C...        FLOW RATE FOR POSITIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS NEGATIVE
C...        WM(1)=-300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DP)
            WM(1)=0.0D0
         END IF
C...
C...  RIGHT END, Z = ZR
         PREND=3.8D0
C...
C...     PRESSURE DROP
         DP=PM(NM)-PREND
         IF(DP.LE.0.0D0)THEN
C...
C...        FLOW RATE FOR NEGATIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS NEGATIVE, AND PROPORTIONAL TO THE SQUARE ROOT OF THE
C...        PRESSURE DROP.  THE FLOW RATE ALSO CHANGES DYNAMICALLY
C...        WITH A TIME CONSTANT OF 0.01 S
            WM(NM)=-300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DABS(DP))
         ELSE
C...
C...        FLOW RATE FOR POSITIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS POSITIVE
```

```
            WM(NM)= 300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DP)
         END IF
```

Note that for the left end of the system of Figure 1b, only the statements corresponding to zero flow (according to eq. (13)) are executed (the other statements are just comments to illustrate how a non-zero flow boundary condition might be programmed). For the right end, the flow, WM(NM), increases with time at a rate corresponding to a time constant of 0.01 s in accordance with eqs. (15) and (16). Note also, the necessity to define consistently the sign of the flow.

Finally, the programming of eq. (6) over the NM spatial grid points concludes subroutine DERV

```
      DO 11 I=1,NM
C...
C...  LEFT END
      IF(I.EQ.1)THEN
C...
C...     MOMENTUM BALANCE DOES NOT SET THE FLOW RATE AT THE LEFT END
C...     (THE FLOW RATE IS SET BY THE PRECEDING BOUNDARY CONDITION)
         WMT(I)= 0.0D0
C...
C...  RIGHT END
      ELSE IF(I.EQ.NM)THEN
C...
C...     MOMENTUM BALANCE DOES NOT SET THE FLOW RATE AT THE RIGHT END
C...     (THE FLOW RATE IS SET BY THE PRECEDING BOUNDARY CONDITION)
         WMT(I)= 0.0D0
C...
C...  INTERIOR POINTS
      ELSE
C...
C...     CONVECTIVE MOMENTUM FORCE
         GIM1=WM(I-1)**2/RHOM(I-1)
         GI  =WM(I  )**2/RHOM(I  )
         GIP1=WM(I+1)**2/RHOM(I+1)
         CMF=((GIP1+GI)/2.0D0-(GIM1+GI)/2.0D0)/DZ
C...
C...     FRICTIONAL FORCE
         FF=0.02D0
         DC=0.03D0
         FT=(FF/(2.0D0*DC))*DABS(WM(I))*WM(I)/RHOM(I)
C...
C...     PRESSURE FORCE
         DPDZ=1.01325D+05*((PM(I+1)+PM(I))/2.0D0
     +                    -(PM(I-1)+PM(I))/2.0D0)/DZ
C...
C...     LINEAR SPATIAL FILTERING
```

```
          EPS=1.0D+04
          DVDZ=EPS*(WM(I+1)/RHOM(I+1)-2.0D0*WM(I)/RHOM(I)
     +              +WM(I-1)/RHOM(I-1))/(DZ**2)
C...
C...      MOMENTUM BALANCE (DM/DT = SUM FORCES) IS CENTERED
          WMT(I)=-CMF-FT-DPDZ+DVDZ
C...
C...   STORE THE INDIVIDUAL TERMS IN THE MOMENTUM BALANCE AT A PART-
C...   ICULAR GRID POINT FOR PRINTING IN SUBROUTINE PRINT
          IF(I.EQ.NM-1)THEN
             TERM(1)=-CMF
             TERM(2)=-FT
             TERM(3)=-DPDZ
             TERM(4)=DVDZ
             TERM(5)=WMT(I)
          END IF
       END IF
C...
C...   NEXT INTERIOR GRID POINT
11     CONTINUE
```

An IF-THEN-ELSE determines how the temporal derivative of eq. (6), WMT(I), is computed. For I = 1 and NM, this derivative is zero since the flow at the boundaries is set by boundary conditions (13), (15) and (16) programmed previously. For the interior points, the individual terms in eq. (6) are computed; these terms are then summed in accordance with eq. (6).

Note that the momemtum equation (6) is approximated in the coding of DO loop 11 by *central differences.* The effectiveness of this central differencing is enhanced by the choice of the smoothing parameter $\epsilon = 10^4$ in the term $\epsilon\partial^2 v/\partial z^2$ (this value of ϵ was selected by trial and error as the minimum value to produce smooth solutions). Also, the individual terms in eq. (6) are stored at one grid point, in this case, I = NM-1 for printing in subroutine PRINT; this output gives a direct indication of the magnitude of each of the forces in eq. (6).

This completes the programming of the momentum equation, eq. (6). We now proceed to the continuity equation, eq. (3).

(3.2) Continuity Equation

The programming of continuity equation (3) is also contained in DO loop 11 within subroutine DERV (although the continuity equation programming was deleted in the preceding discussion of the momentum equation, and in the following discussion, we delete the programming of the momemtum equation to focus on the continuity equation). The approximation of the continuity equation at each

of the grid points in z is therefore programmed as

```
      DO 11 I=1,NM
C...
C...  LEFT END
      IF(I.EQ.1)THEN
C...
C...     CONTINUITY BALANCE IS UPWINDED
         RHOMT(I)=-(WM(I+1)-WM(I))/DZ
C...
C...  RIGHT END
      ELSE IF(I.EQ.NM)THEN
C...
C...     CONTINUITY BALANCE IS UPWINDED
         RHOMT(I)=-(WM(I)-WM(I-1))/DZ
C...
C...  INTERIOR POINTS
      ELSE
C...
C...  CONTINUITY BALANCE IS UPWINDED
C...
C...     NEGATIVE FLOW
         IF(WM(I).LT.0.0D0)THEN
         RHOMT(I)=-(WM(I+1)-WM(I))/DZ
C...
C...     POSITIVE FLOW
         ELSE IF(WM(I).GE.0.0D0)THEN
         RHOMT(I)=-(WM(I)-WM(I-1))/DZ
         END IF
      END IF
C...
C...  NEXT INTERIOR GRID POINT
11    CONTINUE
```

Note that at all grid points, the continuity balance is upwinded; at the interior points, a test is made for the direction of flow (the sign of WM(I)), and the choice of a two-point approximation for the spatial derivative in eq. (3), $\partial W/\partial z$, is made accordingly. Note also that a boundary condition is not specified for eq. (3), i.e., the specification of W at a boundary value of z. Rather, W is set by one of the boundary conditions for eq. (6), i.e., eq. (13) or (15) and (16).

(3.3) Energy Equation

The programming of energy equation (8) is also contained in DO loop 11 within subroutine DERV (although the energy equation programming was deleted in the preceding discussion of the momentum and continuity equations, and in the following discussion, we delete the programming of

the momentum and continuity equations to focus on the energy equation). The approximation of the energy equation at each of the grid points in z is therefore programmed as

```
      DO 11 I=1,NM
C...
C...  LEFT END
      IF(I.EQ.1)THEN
C...
C...     ENERGY BALANCE IS UPWINDED
         UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+QM(I)
C...
C...  RIGHT END
      ELSE IF(I.EQ.NM)THEN
C...
C...     ENERGY BALANCE IS UPWINDED
         UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+QM(I)
C...
C...  INTERIOR POINTS
      ELSE
C...
C...  ENERGY BALANCE IS UPWINDED
C...
C...     AXIAL CONDUCTION
         DIF=1.0D+03
         DTDZ=DIF*(TM(I+1)-2.0D0*TM(I)+TM(I-1))/(DZ**2)
C...
C...     NEGATIVE FLOW
         IF(WM(I).LT.0.0D0)THEN
         UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+DTDZ+QM(I)
C...
C...     POSITIVE FLOW
         ELSE IF(WM(I).GE.0.0D0)THEN
         UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+DTDZ+QM(I)
         END IF
      END IF
C...
C...  NEXT INTERIOR GRID POINT
11    CONTINUE
```

Note that at all grid points, the energy balance is upwinded; at the interior points, a test is made for the direction of flow (the sign of WM(I)), and the choice of a two-point approximation for the spatial derivative in eq. (8), $\partial(\mathrm{Wh})/\partial z$, is made accordingly. Note also that a boundary condition is not specified for eq. (8), i.e., the specification of Wh at a boundary value of z. Rather, W is set by one of the boundary conditions for eq. (6), i.e., eq. (13) or (15) and (16), and h is set through the function HRHOU discussed in the section on algebra. The value of the axial conductivity, DIF, was selected

arbitrarily just to test the coding.

An entering temperature could be specified as a boundary condition, which could then be converted to an enthalpy for inclusion in the preceding coding. However, this was not done for this test problem.

The preceding choice of numerical approximations, i.e., upwinding for the continuity and energy equations, and central approximations for the energy equation, is similar to the approach of Wong (1989)(1990a)(1990b)(1991) and Yeaw (1992). In particular the central approximations used in the momentum balance, eq. (6), offer the advantage that the individual terms in the approximating ODEs (defined by WMT(I) in DO loop 11) need not be selected according to the direction of flow. An alternative, however, would be to upwind the momentum balance as well; this would add to the complexity of the coding somewhat, but might provide better computational accuracy (e.g., less tendency for numerical oscillation, which is now surpressed by the linear filtering term containing the parameter ϵ). Other relevant papers for the modeling and analysis of compressible flow include those by Jang et al (1991) and Carcagno et al (1992).

This essentially completes the coding of the continuity, momentum and energy equations in DERV for this test problem (eqs. (3), (6) and (8)). A complete listing of DERV is given in Appendix 1a; note in particular that eqs. (3), (6) and (8) are all included in DO loop 11.

(3.4) Initial Conditions

We now consider the programming of the initial conditions for eqs. (3), (6) and (8) for the system of Figure 1b, which is done in subroutine INITAL. The beginning of this subroutine contains the same COMMON area as was discussed previously for DERV. Since INITAL is called only once during each run (execution of the code), parameters are also be set in this routine

```
C...
C...   SELECT THE THERMO MODEL
C...
C...       IMODEL = 3 - MODIFIED MARTIN EQUATION (VAN DER WAALS FAMILY)
C...
           IMODEL=3
C...
C...   LOWER AND UPPER LIMITS OF Z
       ZL=0.0D0
       ZR=8.0D0
C...
C...   GRID SPACING
       DZ=(ZR-ZL)/DFLOAT(NM-1)
```

```
C...
C...   AXIAL GRID
       DO 11 I=1,NM
          Z(I)=ZL+DFLOAT(I-1)*DZ
11     CONTINUE
C...
C...   INITIAL FLOW RATE
       WMI=0.0D0
C...
C...   INITIAL TEMPERATURE
       TMI=4.0D0
C...
C...   INITIAL PRESSURE
       PMI=4.0D0
```

This coding is largely self explanatory. Note in particular that the length of the system in Figure 1b is set to eight meters. Also, the initial mass flow rate, temperature and pressure are set to 0 kg/m^2-s, 4 k and 4 atm, respectively.

The initial conditions for eqs. (3), (6) and (8) are then set in DO loop 1

```
C...
C...   INITIAL CONDITIONS (FOR THE 3 X NM DEPENDENT VARIABLES IN
C...   COMMON/Y/).  RHOPT IS A FUNCTION FOR A PRESSURE-TEMPERATURE
C...   FLASH THAT RETURNS THE FLUID DENSITY.  UPT IS A FUNCTION FOR
C...   A PRESSURE-TEMPERATURE FLASH THAT RETURNS THE INTERNAL ENERGY.
C...   NOTE THAT DEPENDENT VARIABLE UMG IS THE PRODUCT OF THE FLUID
C...   INTERNAL ENERGY AND DENSITY (I.E., THE VOLUMETRIC INTERNAL
C...   ENERGY)
       DO 1 I=1,NM
          WM(I)=WMI
        RHOM(I)=RHOPT(PMI,TMI)
         UMG(I)=  UPT(PMI,TMI)*RHOM(I)
1      CONTINUE
```

These initial conditions are for W, ρ and ρu, as required by the temporal derivatives of eqs. (6), (3) and (8), respectively, i.e., the dependent variables in COMMON/Y/.

A call to DERV then computes the initial derivatives (in COMMON/F/)

```
C...
C...   INITIAL DERIVATIVES
       CALL DERV
C...
C...   INITIALIZE COUNTER FOR PLOTTED SOLUTION
       IP=0
```

Also, a counter, IP, is initialized for use in subroutine PRINT, which is considered next. A complete listing of INITAL is given in Appendix 1b.

(3.5) Output

The output from the model is produced by calls to subroutine PRINT. The solution for printing and plotting is transferred through COMMON, which is the same as in INITAL anmd DERV. The first set of executable statements: (a) opens a file TERMS to receive the individual terms of the momentum balance stored in array TERM in DERV, (b) increments the counter IP for subsequent use in printing and plotting (recall that IP was initialized to zero in INITAL), and (c) writes IP to the screen of the terminal used to execute the program (so that the progress of the solution can be monitored).

```
C...
C...   OPEN A FILE FOR PRINTING THE INDIVIDUAL TERMS IN THE MOMENTUM
C...   BALANCE
       OPEN(8,FILE='TERMS',STATUS='UNKNOWN')
C...
C...   INCREMENT THE COUNTER FOR THE PLOTTED OUTPUT
       IP=IP+1
C...
C...   MONITOR THE OUTPUT
       WRITE(*,*)IP
```

Next, the numerical solution is printed (during each call to PRINT)

```
C...
C...   WRITE THE NUMERICAL SOLUTION
       WRITE(NO,1)T,(Z(I),WM(I),PM(I),TM(I),RHOM(I),QM(I),I=1,NM)
1      FORMAT(//,' Time = ',F7.3,//,
     + '          z','          W','          P',
     + '          T','        rho',
     + '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
```

The terms stored in array TERM are then printed (to give an indication of the contribution of the individual terms in the momentum balance as the solution evolves)

```
C...
C...   WRITE THE INDIVIDUAL TERMS IN THE MOMENTUM BALANCE AT AN
C...   INTERIOR GRID POINT (SELECTED IN DERV)
       WRITE(8,2)T,(TERM(I),I=1,5)
2      FORMAT(   ' T = ',F5.3,/,
     +          '     -D(W**2/RHO)/DZ = ',F9.2,/,
```

```
+          '                    -FT = ',F9.2,/,
+          '                 -DP/DZ = ',F9.2,/,
+          '                  DV/DZ = ',F9.2,/,
+          '                 DWM/DT = ',F9.2,//)
```

The output from these write statements is rather extensive, and therefore plotting of the solutions (via Top Drawer) is also included by a call to subroutine PLOTTD to facilitate visualization of the solutions.

```
C...
C...  PLOT THE SOLUTION
      CALL PLOTTD
```

Subroutine PLOTTD uses the counter IP to write a series of Top Drawer commands and the numerical solution to a file for plotting. Subroutine PRINT is listed in Appendix 1c. Since subroutine PLOTTD is rather machine specific, it is not discussed here, but rather, only is listed in Appendix 1d. Also, the coding in PLOTTD is essentially self explanatory.

This essentially completes the programming of the model for the system of Figure 1b. A data file is also provided to control the integration of the ODEs which approximate eqs. (3), (6) and (8)

```
PROTOTYPE VENTING MODEL
0.        0.07      0.01
   99                        0.0001
END OF RUNS
```

This data file contains:

(1) Line 1: A documentation title which is merely read, then printed.

(2) Line 2: The initial, final and output intervalues of time. In this case, time runs from 0 to 0.07 s (70 ms), and subroutine PRINT is called every 0.01 s (every 10 ms).

(3) Line 3: The numbers of ODEs and the error tolerance. For this problem, we have 3 PDEs each approximated over a 33-point grid or 3 x 33 = 99 ODEs, to be integrated with an error tolerance of 0.0001.

The final line in the data set has the characters END OF RUNS to terminate program execution. This data file is read by a main program which then calls the ODE integrator, subroutine RKF45 [Forsythe,

et al (1977)] in this case, to integrate the 99 ODEs. The main program is listed in Appendix 1e.

Finally, to complete the discussion of the Fortran, functions UTP, RHOTP, PRHU, TRHOU and HRHOU, which are called in INITAL and DERV for the evaluation of thermdynamic properties, are listed in Appendix 1f. In each case, subroutine THERMO is called to execute the APCI model 3 for helium properties.

A sample of the output from PRINT is listed below

```
RUN NO. -   1  PROTOTYPE VENTING MODEL

INITIAL T -  0.000D+00

  FINAL T -  0.700D-01

  PRINT T -  0.100D-01

NUMBER OF DIFFERENTIAL EQUATIONS -  99

MAXIMUM INTEGRATION ERROR -  0.100D-03

Time =   0.000

        z          W          P          T        rho          Q
     0.00       0.00      4.000      4.000     139.79   0.000E+00
     0.25       0.00      4.000      4.000     139.79   0.000E+00
     0.50       0.00      4.000      4.000     139.79   0.000E+00
     0.75       0.00      4.000      4.000     139.79   0.000E+00
     1.00       0.00      4.000      4.000     139.79   0.000E+00
     1.25       0.00      4.000      4.000     139.79   0.000E+00
     1.50       0.00      4.000      4.000     139.79   0.000E+00
     1.75       0.00      4.000      4.000     139.79   0.000E+00
     2.00       0.00      4.000      4.000     139.79   0.000E+00
                  .                                .
                  .                                .
                  .                                .
     6.00       0.00      4.000      4.000     139.79   0.000E+00
     6.25       0.00      4.000      4.000     139.79   0.000E+00
     6.50       0.00      4.000      4.000     139.79   0.000E+00
     6.75       0.00      4.000      4.000     139.79   0.000E+00
     7.00       0.00      4.000      4.000     139.79   0.000E+00
     7.25       0.00      4.000      4.000     139.79   0.000E+00
     7.50       0.00      4.000      4.000     139.79   0.000E+00
     7.75       0.00      4.000      4.000     139.79   0.000E+00
     8.00       0.00      4.000      4.000     139.79   0.000E+00

Time =   0.010
```

z	W	P	T	rho	Q
0.00	0.00	4.000	4.000	139.79	0.000E+00
0.25	0.00	4.000	4.000	139.79	0.000E+00
0.50	0.00	4.000	4.000	139.79	0.000E+00
0.75	0.00	4.000	4.000	139.79	0.000E+00
1.00	0.00	4.000	4.000	139.79	0.000E+00
1.25	0.00	4.000	4.000	139.79	0.000E+00
1.50	0.00	4.000	4.000	139.79	0.000E+00
1.75	0.00	4.000	4.000	139.79	0.000E+00
2.00	0.00	4.000	4.000	139.79	0.000E+00
	.			.	
	.			.	
	.			.	
6.00	15.34	3.977	3.998	139.74	0.000E+00
6.25	21.06	3.967	3.997	139.72	0.000E+00
6.50	27.48	3.955	3.996	139.69	0.000E+00
6.75	34.11	3.941	3.994	139.66	0.000E+00
7.00	40.38	3.927	3.993	139.63	0.000E+00
7.25	45.77	3.913	3.992	139.60	0.000E+00
7.50	50.05	3.902	3.991	139.57	0.000E+00
7.75	53.28	3.893	3.990	139.55	0.000E+00
8.00	55.76	3.886	3.989	139.54	0.000E+00
	.			.	
	.			.	

Time = 0.020

.

.

Time = 0.030

.

.

Time = 0.040

.

.

Time = 0.050

.

.

Time = 0.060

.

.

Time = 0.070

z	W	P	T	rho	Q
0.00	0.00	3.714	3.973	139.15	0.000E+00
0.25	0.38	3.714	3.973	139.15	0.000E+00
0.50	0.78	3.715	3.973	139.15	0.000E+00
0.75	1.19	3.715	3.973	139.15	0.000E+00
1.00	1.66	3.716	3.973	139.16	0.000E+00
1.25	2.16	3.716	3.973	139.16	0.000E+00
1.50	2.74	3.717	3.973	139.16	0.000E+00
1.75	3.40	3.718	3.973	139.16	0.000E+00

2.00	4.15	3.720	3.973	139.17	0.000E+00
	.			.	
	.			.	
	.			.	
6.00	35.54	3.787	3.980	139.32	0.000E+00
6.25	38.18	3.793	3.980	139.33	0.000E+00
6.50	40.73	3.799	3.981	139.34	0.000E+00
6.75	43.18	3.805	3.982	139.36	0.000E+00
7.00	45.47	3.810	3.982	139.37	0.000E+00
7.25	47.59	3.816	3.983	139.38	0.000E+00
7.50	49.52	3.821	3.983	139.39	0.000E+00
7.75	51.22	3.826	3.984	139.40	0.000E+00
8.00	52.68	3.831	3.984	139.41	0.000E+00

We note that the preceding solution demonstrates the expected drop in pressure, and the increase in mass flow, particularly near the end at z = 8 m. Also, the solution is free of oscillation, principally due to the upwinding of the continuity and energy balances, and the linear filtering term with the parameter ϵ.

The individual terms stored in the array TERM (for the point NM - 1, or one point from the right end of Figure 1b) is summarized below

```
T = 0.000
   -D(W**2/RHO)/DZ =      0.00
                -FT =      0.00
            -DP/DZ =      0.00
             DV/DZ =      0.00
            DWM/DT =      0.00

T = 0.010
   -D(W**2/RHO)/DZ =     -8.67
                -FT =     -6.78
            -DP/DZ =   3098.20
             DV/DZ =   -861.52
            DWM/DT =   2221.23

T = 0.020
   -D(W**2/RHO)/DZ =     -2.47
                -FT =     -9.97
            -DP/DZ =    661.88
             DV/DZ =   -122.22
            DWM/DT =    527.23

T = 0.030

   -D(W**2/RHO)/DZ =     -0.82
                -FT =    -10.95
            -DP/DZ =    209.11
```

```
                DV/DZ =     -31.37
               DWM/DT =     165.98

T = 0.040
   -D(W**2/RHO)/DZ =       -0.30
                -FT =     -11.27
              -DP/DZ =     77.04
               DV/DZ =    -11.06
              DWM/DT =     54.41

T = 0.050
   -D(W**2/RHO)/DZ =       -0.32
                -FT =     -11.33
              -DP/DZ =    -11.10
               DV/DZ =    -24.36
              DWM/DT =    -47.10

T = 0.060
   -D(W**2/RHO)/DZ =       -2.24
                -FT =     -10.38
              -DP/DZ =   -543.90
               DV/DZ =   -152.94
              DWM/DT =   -709.46

T = 0.070
   -D(W**2/RHO)/DZ =       -4.63
                -FT =      -6.27
              -DP/DZ =  -2014.69
               DV/DZ =   -276.11
              DWM/DT =  -2301.71
```

We can conclude from this output that: (a) the relative contributions of the individual terms in the momentum equation change as the solution evolves, and (b) the pressure term at most times is relatively large, and often dominates the other terms.

The plotted solution produced by the call to PLOTTD in PRINT is in Figure 1c (i.e. from file P1.TOP produced by PLOTTD). Recall again that this solution is for a step change in the downstream pressure of Figure 1b from 4 atm to 3.8 atm with a time constant of 0.010 s ($\tau_V = 0.010$ in eqs. (15) and (16)).

Figure 1c demonstrates the propagation of a pressure wave from z = 8 m to z = 0, followed by a reflection of the wave. The corresponding plots for mass flow rate, temperature and density (from files W1.TOP, T1.TOP and R1.TOP produced by PLOTTD) are not included here, but they demonstrate similar traveling wave characteristics. The computer run time was less than one minute on the SSCL Vax.

The preceding calculation was repeated with $\tau_V = 0$ in eqs. (15) and (16)). The coding for this condition is

```
C...
C...  RIGHT END, Z = ZR
         PREND=3.8D0
C...
C...     PRESSURE DROP
         DP=PM(NM)-PREND
         IF(DP.LE.0.0D0)THEN
C...
C...        FLOW RATE FOR NEGATIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS NEGATIVE, AND PROPORTIONAL TO THE SQUARE ROOT OF THE
C...        PRESSURE DROP.  THE FLOW RATE ALSO CHANGES DYNAMICALLY
C...        WITH A TIME CONSTANT OF 0.01 S
            WM(NM)=-300.0D0*DSQRT(DABS(DP))
         ELSE
C...
C...        FLOW RATE FOR POSITIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS POSITIVE
            WM(NM)= 300.0D0*DSQRT(DP)
         END IF
```

Note that this coding is the same as for the previous case except that the exponential in time has been removed.

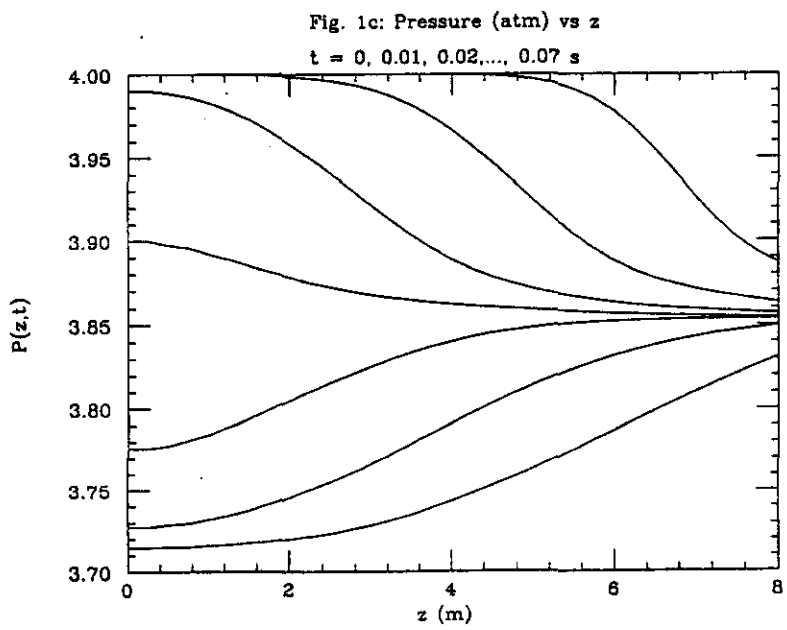


Figure 1c: Pressure Response of the System of Figure 1b to a -0.2 Step Change in Pressure with $\tau_V = 0.010$ s, 33 Grid Points, $\epsilon = 10^4$

We can note the following points about the solution in Figure 1d:

(1) A pressure wave again propagates right to left, as in Figure 1c, but the waves have a somewhat steeper front because the change in pressure at the right end is faster ($\tau_V = 0$ rather than $\tau_V = 0.010$ s); also the wave arrives at the left end about 0.010 s (10 ms) sooner.

(2) If we estimate the time of arrival of the wave to be approximately 0.033 s (admittedly, this is estimated somewhat arbitrarily), then we estimate the speed of sound to be 8/0.033 = 242 m/s. The tabulated value of the speed of sound at 0.4 MPa and 3.9 k [Arp, et al (1989)] is 236.2 m/s.

(3) The right end pressure remains relatively constant at approximately 3.85 atm for 0.050 s until the reflected wave reaches the right end, then continues to decline to the final steady state pressure of 3.8 atm. This constant pressure at the right end is a result of the uniformly spaced pressure waves (uniformly spaced also in time) and the requirement of conserved mass imposed by eq. (3) [Carcagno (1993)].

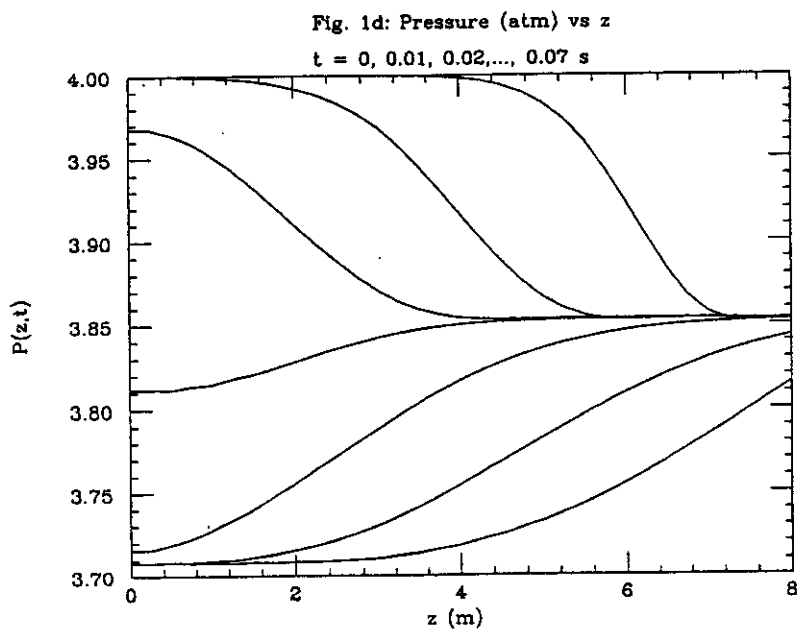


Figure 1d: Pressure Response of the System of Figure 1b
to a -0.2 Step Change in Pressure with $\tau_V = 0$, 33 Grid Points, $\epsilon = 10^4$

(4) During the transient phase of the solution, the pressure, particularly at the left end, drops below the external pressure of 3.8 atm. This lower pressure results from the complex interplay of the continuity and momentum equations, and the flow actually reverses at times (going from right to left) as a result of the reversed pressure gradient (but again, this does not happen instanteously because of the momentum effects of eq. (6)).

(5) The solution of Figure 1c was allowed to run to steady state, at which point the pressure assumed a constant value of 3.8 atm and the flow went to zero, as expected.

(6) The curves of Figure 1d display a roughness due to the spatial discretization (33 points over 8 m or a grid spacing of 8/(33 - 1) = 0.25 m. To demonstrate this effect, the code was executed with 51 grid point (so that the grid spacing is 8(51 - 1) = 0.16 m), corresponding to 3 x 51 = 153 ODEs (NM = 51 in the preceding subroutines and 153 ODEs specified in the data file rather than 99). The solution is given in Figure 1e. We note that the curves are somewhat smoother than in Figure 1d (as expected with more grid points), and the fronts are slightly steeper (this can be seen by overlaying

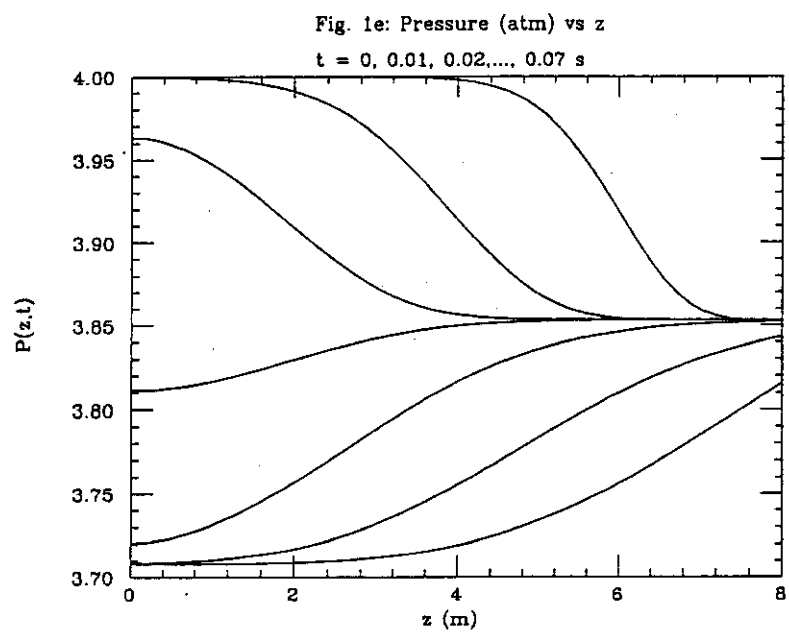


Figure 1e: Pressure Response of the System of Figure 1b
to a -0.2 atm Step Change with $\tau_V = 0$, 51 Grid Points, $\epsilon = 10^4$

Figures 1d and 1e) due to slightly less numerical diffusion resulting from the upwinding of eqs. (3) and (8) (again, resulting from more grid points). Thus, we can conclude that reasonable spatial

convergence has been achieved with 33 (or 51) grid points. The solutions of Figures 1c, 1d and 1e were computed with the smoothing parameter (in eq. (6)) $\epsilon = 10^4$ (which again, was set by trial and error to produce smooth solutions with a minimum effect on the shape of the solutions).

To conclude this prelimnary testing of the code, a run was also made for a total length of 90 m (which could represent, for example, a half cell of magnets) using 46 grid points (so that the grid spacing was (90/(46 - 1) = 2 m); this is a considerably coarser grid than used previously, and the first thing that was observed was the requirement to increase the smoothing parameter to produce solutions which did not have unrealistic oscillations. Thus we used $\epsilon = 10^5$ (with 46 x 3 = 138 ODEs). The solution is given in Figure 1f. We can note the following points about the solution in Figure 1f:

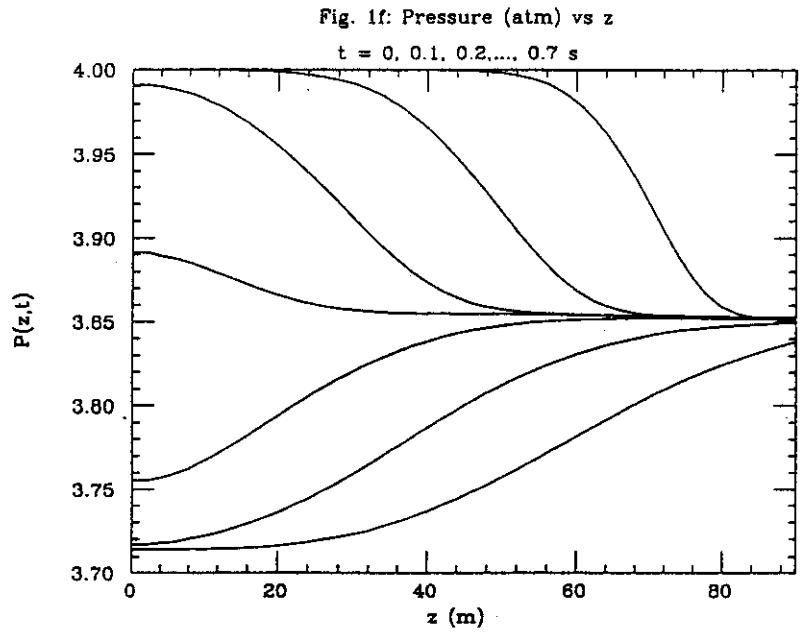


Figure 1f: Pressure Response of the System with a Length of 90 m to a -0.2 atm Step Change with $\tau_V = 0$, 46 Grid Points, $\epsilon = 10^5$

(1) The pressure waves are smooth, and arrive at the left end at an estimated time of 0.38 s (again, estimated somewhat arbitrarily). Thus, the estimated speed of sound is 90/0.38 = 236.8 m/s (the tabulated value is 236.2 m/s).

(2) The right end pressure again remains at about 3.85 atm until the reflected wave reduces it further.

To investigate the effect of the smoothing parameter, ϵ, we reexecuted the case of Figure 1f with $\epsilon = 0.5 \times 10^5$ (half its previous value). The solution is plotted in Figure 1g. We can note the

following points about this figure:

(1) Numerical oscillation has developed indicating that $\epsilon = 10^5$ (of Figure 1f) is about the mimimum value to suppress numerical oscillation.

(2) The fronts in Figure 1g are somewhat steeper than in Figure 1f due to the reduced numerical diffusion resulting from the reduced value of ϵ.

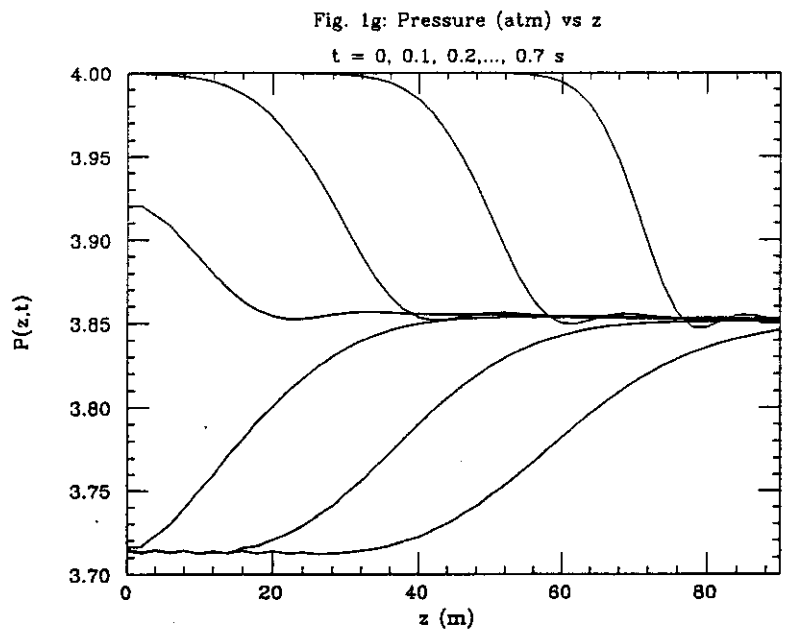


Figure 1g: Pressure Response of the System with a Length of 90 m to a -0.2 atm Step Change with $\tau_V = 0$, 46 Grid Points, $\epsilon = 0.5 \times 10^5$

Finally, to demonstrate that the smaller value of ϵ can be used if the spatial resolution is improved through an increased number of grid points, the preceding case of Figure 1g was executed with 91 grid points (so that the grid spacing was 90/(91 - 1) = 1m). The solution is plotted in Figure 1h, and we can conclude the increased number of grid points (91 rather than 46) eliminated the numerical oscillation of Figure 1g with little effect on the shape of the fronts.

More generally, we can conclude that a tradeoff is probably required in selecting a value of ϵ between minimizing numerical diffusion and minimizing numerical oscillation, and that the optimum value of ϵ will have to be selected by trial and error for each model. Hopefully, the effect of ϵ will not be detrimental with respect to the validity of the model and the conclusions that can be drawn from it.

Also, of course, the number of grid points is a factor in selecting ϵ.

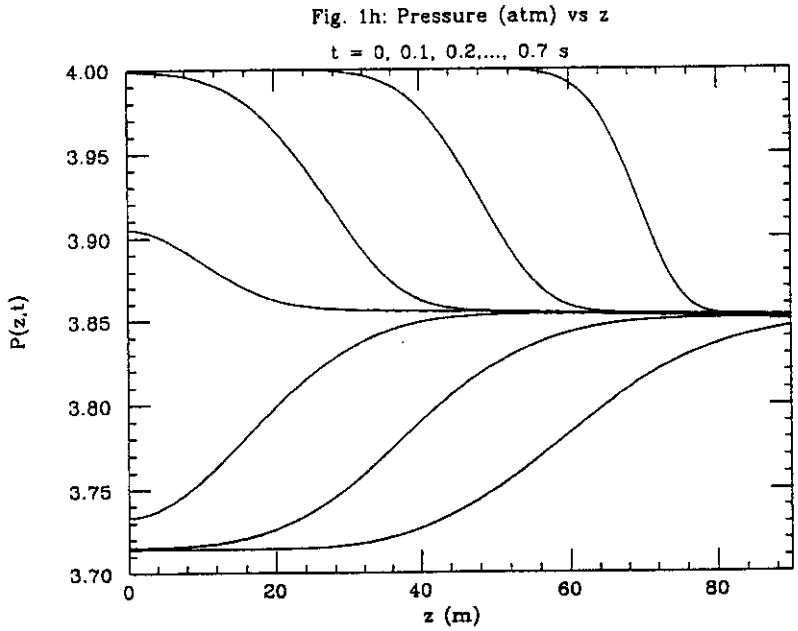


Figure 1h: Pressure Response of the System with a Length of 90 m to a -0.2 atm Step Change with $\tau_V = 0$, 90 Grid Points, $\epsilon = 0.5 \times 10^5$

(4) <u>Prototype Venting Model</u>

We now consider the application of the preceding ideas to a prototype helium venting system depicted in Figure 2a. This system consists of the following components:

(1) A string of magnets, 90 m in length; the Fortran variables for this string are: (a) ρ, RHOM, (b) W (mass flow rate), WM and (c) ρu, UMG. These state variables are defined over a spatial grid of NM points and therefore lead to 3 x NM ODEs as explained previously for the system of Figure 1b.

(2) A 20 k return line which runs in parallel with the string of magnets for which the state variables are: (a) ρ, RHOR, (b) W, WR_i and (c) ρu, URG. These state variables are defined over a spatial grid of NR points and therefore lead to 3 x NR ODEs as expalined previously for the system of Figure 1b.

(3) A dewar for which continuity and energy balances are written (in this case, for single phase helium, although these equations could also be written for two phases), leads to two ODEs.

(4) Valves in the places indicated in Figure 2a, which are modeled with algebraic equations that include a time constant in an exponential function of time to approximate the dynamics of the valves.

Thus, the total number of ODEs is 3 x (NM + NR) + 2. For the present discussion, we take NM = NR = 46 so the number of ODEs is 3 x (46 + 46) + 2 = 278.

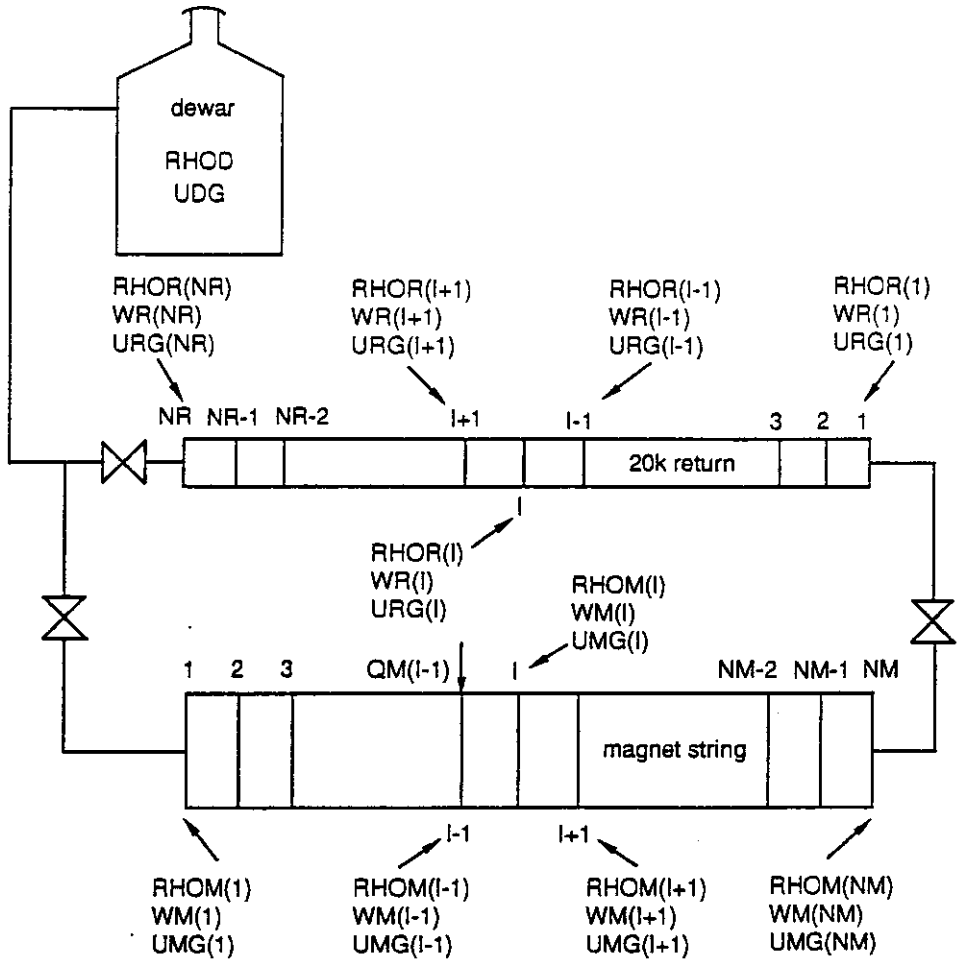


Figure 2a: Prototype Venting System Consisting of a 90 m String of Magnets, a Parallel 20 k Return Line and a Dewar

(4.1) Derivatives (DERV)

Subroutine DERV is structured the same way as subroutine DERV for the system of Figure 1b (and which is listed in Appendix 1a). The COMMON area now has state variables for the string of magnets, the return line and the dewar

```
      SUBROUTINE DERV
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=46,NR=46)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),   UMG(NM),
     +                          RHOD,        UDG,
     +             WR(NR),  RHOR(NR),   URG(NR)
     2      /F/    WMT(NM), RHOMT(NM),  UMGT(NM),
     +                         RHODT,       UDGT,
     +            WRT(NR), RHORT(NR),  URGT(NR)
     3      /V/     UM(NM),    PM(NM),     TM(NM),     HM(NM),
     +                 UD,        PD,         TD,         HD,
     +             UR(NR),    PR(NR),     TR(NR),     HR(NR)
     4      /C/         ZL,        ZR,         DZ,      Z(NM),
     5       R,        TMI,       PMI,        WMI,     QM(NM),
     +      QD,        TDI,       PDI,         VD,         AC,
     +                 TRI,       PRI,        WRI,     QR(NR)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
```

We can note the following points about this COMMON area:

(1) The first, second and third lines of COMMON/Y/ have the state variables for the magnets, the dewar and the return, respectively. The corresponding temporal derivatives are in the first, second and third lines of COMMON/F/ (the naming convention for the temporal derivatives is to add a "T" to the end of the state variable name, e.g., the magnet density, RHOM(NM), has the temporal derivative *RHOMT(NM)*).

(2) COMMON/V/ contains other time dependent variables. Thus, the first line of COMMON/V/ has the internal energy, UM, the pressure, PM, the temperature, TM and the enthalpy, HM, of the helium in the magnets. Similarly, the second line of COMMON/V/ has the same properties for the dewar, and the third line has these properties for the return.

(3) COMMON/C/ has various model constants which are explained subsequently, e.g., ZL is the axial coordinate at the left end of the magnet string.

The first executable code in DERV is for the model algebra as in the previous DERV. Do loop 1 generally computes the thermodynamic properties for the helium in the magnets

```
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
```

```
      DO 1 I=1,NM
         UM(I)=UMG(I)/RHOM(I)
         PM(I)=PRHOU(RHOM(I),UM(I))
         TM(I)=TRHOU(RHOM(I),UM(I))
         HM(I)=HRHOU(RHOM(I),UM(I))
         IF((I.GE.1).AND.(I.LE.NM-5))THEN
           QM(I)=1.0D+05*(1.0D0-DEXP(-T/0.5D0))*(10.0D0-TM(I))
         ELSE IF(I.GT.NM-5)THEN
           QM(I)=0.0D0
         END IF
1     CONTINUE
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
C...  FOR THE RETURN
      DO 2 I=1,NR
         UR(I)=URG(I)/RHOR(I)
         PR(I)=PRHOU(RHOR(I),UR(I))
         TR(I)=TRHOU(RHOR(I),UR(I))
         HR(I)=HRHOU(RHOR(I),UR(I))
         QR(I)=0.0D0
2     CONTINUE
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, HEAT INPUT FOR THE DEWAR
      UD=UDG/RHOD
      PD=PRHOU(RHOD,UD)
      TD=TRHOU(RHOD,UD)
      QD=0.0D0
```

Then the algebra for the return is done in DO loop 2, followed by the algebra for the dewar. Note also that the heat input to each of these components, is defined in this code, e.g., $\dot{Q}$ in eq. (8) for the magnets is computed as

```
QM(I)=1.0D+05*(1.0D0-DEXP(-T/0.5D0))*(10.0D0-TM(I))
```

which is an exponentially increasing function of time, with a time constant of 0.5 s, and is driven by a temperature difference (10 - TM). This heat input is used at grid points I to NM - 5. Beyond grid point NM - 5 (to NM), the heat input is zero.

The flow rates into the three units (magnets, dewar and return) are then set by boundary conditions in much the same way as in the preceding DERV

```
C...
C...  BOUNDARY CONDITIONS
C...
```

```
C...   MAGNETS, Z = ZL
          DP=PM(1)-PD
          IF(DP.LE.0.0D0)THEN
             WM(1)=0.0D0
          ELSE
             WM(1)=-300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
          END IF
C...
C...   MAGNETS, Z = ZR
          DP=PM(NM)-PR(1)
          IF(DP.LE.0.0D0)THEN
             WM(NM)=0.0D0
          ELSE
             WM(NM)= 300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
          END IF
C...
C...   RETURN, Z = ZL
          DP=PR(NR)-PD
          IF(DP.LE.0.0D0)THEN
             WR(NR)=0.0D0
          ELSE
             WR(NR)= 300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
          END IF
C...
C...   RETURN, Z = ZR
          DP=PM(NM)-PR(1)
          IF(DP.LE.0.0D0)THEN
             WR(1)=0.0D0
          ELSE
             WR(1)=WM(NM)
          END IF
```

These boundary conditions generally reflect the equality of mass flow rates between the three units. We should note the following points about this coding:

(1) The grid for the magnets, defined by I = 1 to NM, runs left to right in Figure 2a. The grid for the return, defined by I = 1 to NR, runs right to left. This choice of grid definitions is arbitrary, and other arrangements could have been used. However, once the grids are defined, the consistent use of flows is essential, including the correct signs of the flows.

(2) The flows for the magnets and the return are set by pressure drop relationships as before (but the external pressure of the preceding system of Figure 1b is now the dewar pressure, PD). The flow rates change according to a time constant of 0.1 s, which could represent, for example, the dynamics of the valves.

(3) *The one exception for (2) is the right end flow of the return which is equated to the corresponding* right end flow of the magnets,

```
WR(1)=WM(NM)
```

Again, alternative formulations are possible, and this process of writing equations does not lead to necessarily unique results.

The PDEs for the magnets are then programmed in DO loop 11, much the same way as in the preceding example of Figure 1b (i.e., as in DERV of Appendix 1a)

```
C...
C...   PDES FOR THE MAGNETS
       DO 11 I=1,NM
C...
C...     RIGHT END
         IF(I.EQ.NM)THEN
             WMT(I)=0.0D0
           RHOMT(I)=-(WM(I)-WM(I-1))/DZ
            UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+QM(I)
C...
C...     LEFT END
         ELSE IF(I.EQ.1)THEN
             WMT(I)=0.0D0
           RHOMT(I)=-(WM(I+1)-WM(I))/DZ
            UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+QM(I)
C...
C...     INTERIOR POINTS
         ELSE
C...
C...        MOMENTUM BALANCE
            GIM1=WM(I-1)**2/RHOM(I-1)
            GI  =WM(I  )**2/RHOM(I  )
            GIP1=WM(I+1)**2/RHOM(I+1)
            DVDZ=(WM(I+1)/RHOM(I+1)
     +           -2.0D0*WM(I)/RHOM(I)
     +           +WM(I-1)/RHOM(I-1))/(DZ**2)
            EPS=1.0D+05
            DPDZ=1.013D+05*((PM(I+1)+PM(I))/2.0D0
     +                     -(PM(I-1)+PM(I))/2.0D0)/DZ
            WMT(I)=-((GIP1+GI)/2.0D0
     +              -(GIM1+GI)/2.0D0)/DZ
     +              -DPDZ+EPS*DVDZ
C...
C...        MASS BALANCE
```

```
            IF(WM(I).LT.0.0D0)THEN
               RHOMT(I)=-(WM(I+1)-WM(I))/DZ
            ELSE IF(WM(I).GE.0.0D0)THEN
               RHOMT(I)=-(WM(I)-WM(I-1))/DZ
            END IF
C...
C...        ENERGY BALANCE
            DTDZ=(TM(I+1)-2.0D0*TM(I)+TM(I-1))/(DZ**2)
            DIF=1.0D+03
            IF(WM(I).LT.0.0D0)THEN
               UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ
     +                 +QM(I)+DIF*DTDZ
            ELSE IF(WM(I).GE.0.0D0)THEN
               UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ
     +                 +QM(I)+DIF*DTDZ
            END IF
         END IF
11     CONTINUE
```

The three temporal derivatives of eqs. (3), (6) and (8), RHOMT(I), WMT(I) and UMGT(I), respectively (which are in COMMON/Y/) are first programmed at grid point I = NM (i.e., the right end of the magnet string). Then these derivatives are programmed at I = 1 (the left end of the magnet string). Finally, the temporal derivatives are programmed at the interior points, I = 2 to NM - 1.

Note again that the momentum balance is approximated with ceneterd differences while the mass and energy balances are upwinded. The smoothing parameter, ϵ, has a value of 10^5, as suggested by the earlier value for the 91 point simulation of the system in Figure 1b (i.e., the value used to produce Figure 1h).

The programming then goes on to the PDEs for the return line in DO loop 12

```
C...
C...   PDES FOR THE RETURN
       DO 12 I=1,NR
C...
C...     LEFT END
         IF(I.EQ.NR)THEN
             WRT(I)=0.0D0
           RHORT(I)=-(WR(I)-WR(I-1))/DZ
            URGT(I)=-(WR(I)*HR(I)-WR(I-1)*HR(I-1))/DZ+QR(I)
C...
C...     RIGHT END
         ELSE IF(I.EQ.1)THEN
             WRT(I)=0.0D0
           RHORT(I)=-(WR(I+1)-WR(I))/DZ
            URGT(I)=-(WR(I+1)*HR(I+1)-WR(I)*HR(I))/DZ+QR(I)
C...
```

```
C...     INTERIOR POINTS
         ELSE
C...
C...        MOMENTUM BALANCE
            GIM1=WR(I-1)**2/RHOR(I-1)
            GI  =WR(I  )**2/RHOR(I  )
            GIP1=WR(I+1)**2/RHOR(I+1)
            DVDZ=(WR(I+1)/RHOR(I+1)
     +          -2.0D0*WR(I)/RHOR(I)
     +          +WR(I-1)/RHOR(I-1))/(DZ**2)
            EPS=1.0D+05
            DPDZ=1.013D+05*((PR(I+1)+PR(I))/2.0D0
     +                     -(PR(I-1)+PR(I))/2.0D0)/DZ
            WRT(I)=-((GIP1+GI)/2.0D0
     +              -(GIM1+GI)/2.0D0)/DZ
     +              -DPDZ+EPS*DVDZ
C...
C...        MASS BALANCE
            IF(WR(I).LT.0.0D0)THEN
               RHORT(I)=-(WR(I+1)-WR(I))/DZ
            ELSE IF(WR(I).GE.0.0D0)THEN
               RHORT(I)=-(WR(I)-WR(I-1))/DZ
            END IF
C...
C...        ENERGY BALANCE
            DTDZ=(TR(I+1)-2.0D0*TR(I)+TR(I-1))/(DZ**2)
            DIF=1.0D+03
            IF(WR(I).LT.0.0D0)THEN
               URGT(I)=-(WR(I+1)*HR(I+1)-WR(I)*HR(I))/DZ
     +                 +QR(I)+DIF*DTDZ
            ELSE IF(WR(I).GE.0.0D0)THEN
               URGT(I)=-(WR(I)*HR(I)-WR(I-1)*HR(I-1))/DZ
     +                 +QR(I)+DIF*DTDZ
            END IF
         END IF
12     CONTINUE
```

The coding of the PDEs for the return line (in DO loop 12) is essentially the same as for the magnet string (in DO loop 11). The only basic difference is in the definition of the grid (running left to right for the magnets, and right to left for the return line).

We assume that the dewar is perfectly mixed so that it is modeled by ODEs in t rather than by PDEs. The coding follows

```
C...
C...   ODES FOR THE DEWAR
       RHODT=-(AC/VD)*WM( 1)
     +       +(AC/VD)*WR(NR)
```

```
      UDGT=-(AC/VD)*WM( 1)*HM( 1)
     +        +(AC/VD)*WR(NR)*HR(NR)
     +        +QD/VD
      RETURN
      END
```

The first equation is the mass balance with time derivative RHODT. The signs of the RHS terms are determined by the gridding of the magnet and return line PDEs. For example, flow out of the left end of the magnets is negative (in the negative z direction; see Figure 2a), and therefore constitutes flow into the dewar. Generally, the vector character of the mass flow rate, W, in eqs. (3), (6) and (8) must be taken into account.

The second equation is the energy balance for the dewar with time derivative UDGT. Note that the RHS terms involve the enthalpies of the flow into or out of the left ends of the magnets and return line, HM(1) and HR(NR), as expected.

This completes the programming of the model temporal derivatives in COMMON/F/ (the complete DERV is listed in Appendix 2a). Note that a total of 3 x 46 (DO loop 11) + 3 x 46 (DO loop 12) + 2 (dewar) = 278 ODEs have been programmed. The initial conditions for these ODEs must then be programmed in INITAL, which is discussed next.

(4.2) Initial Conditions (INITAL)

The COMMON area is the same in INITAL as in DERV. The programming then closely resembles the previous INITAL (in Appendix 1b)

```
C...
C...  SELECT THE THERMO MODEL
C...
C...     IMODEL = 3 - MODIFIED MARTIN EQUATION (VAN DER WAALS FAMILY)
C...
         IMODEL=3
C...
C...  LOWER AND UPPER LIMITS OF Z
      ZL=0.0D0
      ZR=90.0D0
C...
C...  MAGNET GRID SPACING
      DZ=(ZR-ZL)/DFLOAT(NM-1)
C...
C...  MAGNET AXIAL GRID
      DO 11 I=1,NM
         Z(I)=ZL+DFLOAT(I-1)*DZ
11    CONTINUE
```

The thermodynamic correlation is first selected (APCI Model 3). The grid for the magnets is then defined in array Z. Since the grid for the return line is the same, it is not programmed separately.

A series of parameters for the model is then defined (and passed through COMMON to the other routines which require these parameters)

```
C...
C...   DEWAR VOLUME
       VD=5.0D0
C...
C...   LINE CROSS SECTIONAL AREA
       AC=0.04D0
C...
C...   INITIAL MAGNET FLOW RATE
       WMI=1.0D0
C...
C...   INITIAL MAGNET TEMPERATURE
       TMI=4.0D0
C...
C...   INITIAL MAGNET PRESSURE
       PMI=4.0D0
C...
C...   INITIAL DEWAR TEMPERATURE
       TDI=4.5D0
C...
C...   INITIAL DEWAR PRESSURE
       PDI=4.0D0
C...
C...   INITIAL RETURN FLOW RATE
       WRI=1.0D0
C...
C...   INITIAL RETURN TEMPERATURE
       TRI=4.0D0
C...
C...   INITIAL RETURN PRESSURE
       PRI=2.0D0
```

Note in particular that the initial magnet and return line pressures are 4 and 2 atm., respectively, which provide a pressure driving force for venting into the return line. The initial flow rates were set arbitrarily at 1 kg/m^2-s (these could just as well have been zero, for example).

The initial conditions for the 278 ODEs are then set, which is the principal requirement of INITAL

```
C...
```

```
C...   MAGNET INITIAL CONDITIONS
       DO 1 I=1,NM
          WM(I)=WMI
        RHOM(I)=RHOPT(PMI,TMI)
         UMG(I)=  UPT(PMI,TMI)*RHOM(I)
1      CONTINUE
C...
C...   DEWAR INITIAL CONDITIONS
       RHOD=RHOPT(PDI,TDI)
        UDG=  UPT(PDI,TDI)*RHOD
C...
C...   RETURN INITIAL CONDITIONS
       DO 2 I=1,NR
          WR(I)=WRI
        RHOR(I)=RHOPT(PRI,TRI)
         URG(I)=  UPT(PRI,TRI)*RHOR(I)
2      CONTINUE
```

By the end of the execution of DO loop 2, all 278 initial conditions have been set (i.e., all of the state or dependent variable vectors in COMMON/Y have initial values). Therefore, the initial derivative vector can be computed by a call to DERV (the initial derivative vectors in COMMON/F/ can be computed)

```
C...
C...   INITIAL DERIVATIVES
       CALL DERV
C...
C...   INITIALIZE COUNTER FOR PLOTTED SOLUTION
       IP=0
       RETURN
       END
```

This completes subroutine INITAL (a complete listing of INITAL is given in Appendix 2b). Subroutine PRINT merely prints tabular output (plotting could be added as in PRINT of Appendx 1c; this would be particularly worthwhile to better visualize the large number of numbers from subroutine PRINT).

(4.3) Output (PRINT)

The COMMON area of PRINT is the same as in INITAL and DERV. Then a series of write statements are used to give the numerical solution in some detail

```
C...
C...   INCREMENT THE COUNTER FOR THE PLOTTED OUTPUT
```

```
      IP=IP+1
C...
C...  MONITOR THE OUTPUT
      WRITE(*,*)IP
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE MAGNETS
      WRITE(NO,1)T,(Z(I),WM(I),PM(I),TM(I),RHOM(I),QM(I),I=1,NM)
1     FORMAT(//,' Magnets, time = ',F7.3,//,
     +  '          z','          W','          P',
     +  '          T','        rho',
     +  '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE DEWAR
      WRITE(NO,2)PD,TD,RHOD,WM(1),QD
2     FORMAT(/,    ' Dewar',
     +  /,'  PD = ',F7.2,'  TD = ',F7.3,'  RHOD = ',F7.3,
     +     '  WM = ',F7.2,'  QD = ',F7.1)
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE RETURN
      WRITE(NO,3)T,(Z(I),WR(I),PR(I),TR(I),RHOR(I),QR(I),I=1,NR)
3     FORMAT(/,' Return, time = ',F7.3,/,
     +  '          z','          W','          P',
     +  '          T','        rho',
     +  '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
      RETURN
      END
```

Subroutine PRINT is listed in Appendix 2c.

Finally, a data file is required to control the numerical integration of the 278 ODEs

```
PROTOTYPE VENTING MODEL
0.        2.0        0.1
  278                      0.0001
END OF RUNS
```

Time covers the interval $0 \leq t \leq 2$ s with output every 0.1 s. The integration is again by RKF45 (the main program is the same as in the preceding example, and is listed in Appendix 1e). Clearly PRINT produces many numbers (the flow rate, pressure, density, and heat input at 278 points for each of the 21 points in time, t = 0, 0.1, 0.2, . . ., 2 s). We therefore present only some of the highlights of this output.

The output at t = 0 merely reflects the initial conditions set in INITAL

```
RUN NO. -    1   PROTOTYPE VENTING MODEL

INITIAL T -   0.000D+00
```

FINAL T - 0.200D+01

PRINT T - 0.100D+00

NUMBER OF DIFFERENTIAL EQUATIONS - 278

MAXIMUM INTEGRATION ERROR - 0.100D-03

Magnets, time = 0.000

z	W	P	T	rho	Q
0.00	0.00	4.000	4.000	139.79	0.000E+00
2.00	1.00	4.000	4.000	139.79	0.000E+00
4.00	1.00	4.000	4.000	139.79	0.000E+00
6.00	1.00	4.000	4.000	139.79	0.000E+00
8.00	1.00	4.000	4.000	139.79	0.000E+00
10.00	1.00	4.000	4.000	139.79	0.000E+00
	.			.	
	.			.	
	.			.	
80.00	1.00	4.000	4.000	139.79	0.000E+00
82.00	1.00	4.000	4.000	139.79	0.000E+00
84.00	1.00	4.000	4.000	139.79	0.000E+00
86.00	1.00	4.000	4.000	139.79	0.000E+00
88.00	1.00	4.000	4.000	139.79	0.000E+00
90.00	0.00	4.000	4.000	139.79	0.000E+00

Dewar

PD = 4.00 TD = 4.500 RHOD = 131.567 WM = 0.00 QD = 0.0

Return, time = 0.000

z	W	P	T	rho	Q
0.00	0.00	2.000	4.000	131.13	0.000E+00
2.00	1.00	2.000	4.000	131.13	0.000E+00
4.00	1.00	2.000	4.000	131.13	0.000E+00
6.00	1.00	2.000	4.000	131.13	0.000E+00
8.00	1.00	2.000	4.000	131.13	0.000E+00
10.00	1.00	2.000	4.000	131.13	0.000E+00
	.			.	
	.			.	
	.			.	
80.00	1.00	2.000	4.000	131.13	0.000E+00
82.00	1.00	2.000	4.000	131.13	0.000E+00
84.00	1.00	2.000	4.000	131.13	0.000E+00
86.00	1.00	2.000	4.000	131.13	0.000E+00
88.00	1.00	2.000	4.000	131.13	0.000E+00
90.00	0.00	2.000	4.000	131.13	0.000E+00

Although this printing of the initial conditions gives expected and obvious output, the careful examination of this output is worthwhile to ensure that all of the initial conditions (for the 278 state variables), and all of the auxiliary variables (computed by algebra in DERV) are correct and reasonable.

The response of the system of Figure 2a to the heat input, $\dot{Q}$, in eq. (8) and the pressure difference between the magnet string and the return line, begins to evolve from t = 0. For example, the response at t = 0.1 (abbreviated) is

Magnets, time = 0.100

z	W	P	T	rho	Q
0.00	-29.62	4.028	4.015	139.67	0.108E+06
2.00	-25.28	4.036	4.016	139.69	0.108E+06
4.00	-21.25	4.044	4.017	139.70	0.108E+06
6.00	-17.53	4.051	4.017	139.72	0.108E+06
8.00	-14.17	4.057	4.018	139.73	0.108E+06
10.00	-11.18	4.062	4.019	139.74	0.108E+06
12.00	-8.59	4.067	4.019	139.75	0.108E+06
14.00	-6.41	4.071	4.019	139.76	0.108E+06
16.00	-4.60	4.074	4.020	139.77	0.108E+06
18.00	-3.16	4.076	4.020	139.77	0.108E+06
20.00	-2.05	4.078	4.020	139.78	0.108E+06
22.00	-1.21	4.079	4.020	139.78	0.108E+06
24.00	-0.59	4.080	4.020	139.78	0.108E+06
26.00	-0.15	4.080	4.020	139.78	0.108E+06
28.00	0.16	4.081	4.020	139.78	0.108E+06
30.00	0.40	4.081	4.020	139.79	0.108E+06
32.00	0.58	4.082	4.020	139.79	0.108E+06
34.00	0.72	4.082	4.020	139.79	0.108E+06
36.00	0.82	4.082	4.020	139.79	0.108E+06
38.00	0.89	4.082	4.020	139.79	0.108E+06
40.00	0.95	4.082	4.020	139.79	0.108E+06
	.			.	
	.			.	
	.			.	
80.00	157.98	3.800	3.994	139.16	0.109E+06
82.00	173.10	3.720	3.974	139.16	0.000E+00
84.00	184.47	3.701	3.972	139.12	0.000E+00
86.00	195.25	3.654	3.967	139.01	0.000E+00
88.00	203.39	3.629	3.965	138.96	0.000E+00
90.00	210.35	3.602	3.962	138.89	0.000E+00

Dewar

PD = 4.00 TD = 4.500 RHOD = 131.575 WM = -29.62 QD = 0.0

Return, time = 0.100

z	W	P	T	rho	Q
0.00	210.35	2.371	4.046	132.15	0.000E+00
2.00	199.03	2.371	4.046	132.15	0.000E+00
4.00	185.44	2.315	4.040	131.99	0.000E+00
6.00	169.56	2.305	4.038	131.97	0.000E+00
8.00	153.49	2.260	4.033	131.84	0.000E+00
10.00	135.73	2.232	4.029	131.77	0.000E+00
	.			.	
	.			.	
	.			.	
80.00	0.07	2.002	4.000	131.14	0.000E+00
82.00	0.03	2.002	4.000	131.14	0.000E+00
84.00	0.01	2.002	4.000	131.14	0.000E+00
86.00	0.00	2.002	4.000	131.14	0.000E+00
88.00	0.00	2.002	4.000	131.14	0.000E+00
90.00	0.00	2.002	4.000	131.14	0.000E+00

We can note the following points about this output at t = 0.1:

(1) Helium flows out of both ends of the magnet string; note in particular the split in the flow at z = 26 and 28.

26.00	-0.15	4.080	4.020	139.78	0.108E+06
28.00	0.16	4.081	4.020	139.78	0.108E+06

(2) Helium flows out of the right end of the magnet string into the right end of the return line, as expected, since the pressure of the magnet string is higher.

(3) Helium does not flow out of the left end of the return line into the dewar since the required pressure drop has not occurred.

(4) The temperature and pressure at the left end of the magnet string have started to increase due to the heat input.

(5) The temperature and pressure at the right end of the magnet string have started to decrease since there is no heat input and the helium is expanding as it flows into the return line.

(6) The temperature and pressure of the helium in the return line have started to increase since helium is flowing into the return line from the magnet string.

(7) The density changes along the magnet string and return line are small because the helium is relatively incompressible at the existing conditions (approximately 4 atm and 4 k).

These trends continue as the transient proceeds. Thus, at t = 1 s, the output (abbreviated) is

Magnets, time = 1.000

z	W	P	T	rho	Q
0.00	-293.75	5.470	4.797	133.03	0.450E+06
2.00	-276.51	5.476	4.798	133.05	0.450E+06
4.00	-259.21	5.479	4.798	133.06	0.450E+06
6.00	-241.89	5.485	4.798	133.08	0.450E+06
8.00	-224.50	5.487	4.799	133.09	0.450E+06
10.00	-207.09	5.493	4.799	133.11	0.450E+06
12.00	-189.62	5.495	4.799	133.11	0.450E+06
14.00	-172.12	5.500	4.799	133.13	0.450E+06
16.00	-154.58	5.501	4.799	133.13	0.450E+06
18.00	-137.00	5.506	4.800	133.15	0.450E+06
20.00	-119.37	5.506	4.800	133.15	0.450E+06
22.00	-101.72	5.511	4.800	133.17	0.450E+06
24.00	-83.99	5.511	4.800	133.17	0.450E+06
26.00	-66.23	5.515	4.800	133.18	0.450E+06
28.00	-48.45	5.515	4.800	133.18	0.450E+06
30.00	-30.56	5.519	4.801	133.19	0.450E+06
32.00	-12.69	5.517	4.800	133.19	0.450E+06
34.00	5.31	5.522	4.801	133.20	0.450E+06
36.00	23.44	5.520	4.801	133.20	0.450E+06
38.00	41.55	5.524	4.801	133.21	0.450E+06
40.00	59.79	5.523	4.801	133.20	0.450E+06
	.			.	
	.			.	
	.			.	
80.00	434.04	5.539	4.806	133.20	0.449E+06
82.00	444.18	5.359	4.449	138.17	0.000E+00
84.00	444.50	5.512	4.234	141.68	0.000E+00
86.00	439.96	5.399	4.167	142.19	0.000E+00
88.00	432.75	5.495	4.133	142.92	0.000E+00
90.00	425.23	5.423	4.136	142.67	0.000E+00

Dewar

PD = 4.51 TD = 4.560 RHOD = 132.878 WM = -293.75 QD = 0.0

Return, time = 1.000

z	W	P	T	rho	Q
0.00	425.23	3.414	4.168	134.84	0.000E+00
2.00	409.36	3.414	4.168	134.84	0.000E+00

4.00	393.41	3.420	4.169	134.84	0.000E+00
6.00	377.73	3.423	4.170	134.85	0.000E+00
8.00	362.15	3.427	4.171	134.85	0.000E+00
10.00	346.90	3.431	4.171	134.87	0.000E+00
	.			.	
	.			.	
	.			.	
80.00	27.73	3.510	4.178	135.08	0.000E+00
82.00	22.13	3.510	4.179	135.08	0.000E+00
84.00	16.58	3.511	4.179	135.08	0.000E+00
86.00	11.04	3.511	4.179	135.08	0.000E+00
88.00	5.52	3.512	4.179	135.08	0.000E+00
90.00	0.00	3.512	4.179	135.08	0.000E+00

We can note the following points about the output at t = 1 s:

(1) The split in the flow in the magnet string now occurs between z = 32 and 34 m. Thus, the point at which this split occurs is traveling left to right in the magnet string.

32.00	-12.69	5.517	4.800	133.19	0.450E+06
34.00	5.31	5.522	4.801	133.20	0.450E+06

More generally, the traneisnt response consists of traveling wave of pressure, flow, temperature and density which can best be visualized through graphical output.

(2) The pressure and temperature in the magnet string are above their initial values, due primarily to the continuing heat input.

(3) The pressure and temperature in the return line also have increased above their initial values, due primarily to the flow from the magnet string.

The transient response continues to t = 2 (the final value of time set in the data file). The abbreviated output for t = 2 is

Magnets, time = 2.000

z	W	P	T	rho	Q
0.00	-364.26	7.256	5.749	123.73	0.417E+06
2.00	-345.54	7.257	5.748	123.74	0.417E+06
4.00	-326.82	7.257	5.748	123.75	0.417E+06
6.00	-308.09	7.259	5.748	123.76	0.417E+06
8.00	-289.36	7.259	5.748	123.76	0.417E+06
10.00	-270.64	7.261	5.748	123.77	0.417E+06

	.			.	
30.00	-84.47	7.271	5.747	123.84	0.417E+06
32.00	-66.03	7.269	5.747	123.83	0.417E+06
34.00	-47.62	7.273	5.748	123.85	0.417E+06
36.00	-29.29	7.271	5.747	123.84	0.417E+06
38.00	-10.96	7.276	5.748	123.86	0.417E+06
40.00	7.28	7.273	5.748	123.84	0.417E+06
42.00	25.42	7.278	5.748	123.86	0.417E+06
44.00	43.56	7.276	5.748	123.85	0.417E+06
46.00	61.61	7.280	5.748	123.87	0.417E+06
48.00	79.64	7.279	5.748	123.86	0.417E+06
50.00	97.61	7.283	5.749	123.88	0.417E+06
	.			.	
	.			.	
	.			.	
80.00	363.04	7.305	5.753	123.91	0.417E+06
82.00	378.65	7.126	5.327	131.16	0.000E+00
84.00	389.55	7.281	4.909	138.42	0.000E+00
86.00	394.63	7.167	4.616	141.99	0.000E+00
88.00	394.87	7.271	4.416	144.72	0.000E+00
90.00	392.03	7.190	4.335	145.41	0.000E+00

Dewar

PD = 5.78 TD = 4.720 RHOD = 135.639 WM = -364.26 QD = 0.0

Return, time = 2.000

z	W	P	T	rho	Q
0.00	392.03	5.482	4.379	139.62	0.000E+00
2.00	381.77	5.482	4.379	139.62	0.000E+00
4.00	371.53	5.486	4.379	139.63	0.000E+00
6.00	361.32	5.486	4.380	139.62	0.000E+00
8.00	351.15	5.489	4.380	139.63	0.000E+00
10.00	341.05	5.489	4.381	139.62	0.000E+00
	.			.	
	.			.	
	.			.	
80.00	36.61	5.443	4.375	139.53	0.000E+00
82.00	29.25	5.442	4.375	139.53	0.000E+00
84.00	21.92	5.442	4.375	139.53	0.000E+00
86.00	14.61	5.442	4.375	139.53	0.000E+00
88.00	7.30	5.441	4.375	139.53	0.000E+00
90.00	0.00	5.441	4.375	139.53	0.000E+00

The split in the flow in the magnet string continues

38.00	-10.96	7.276	5.748	123.86	0.417E+06
40.00	7.28	7.273	5.748	123.84	0.417E+06

The continuing increase in the pressure and temperature in the magnet string, return line and dewar due to the continuing heat input is apparent. Also, as long as the heat input continues, the transient response will continue (the heat input will eventually be moderated by the temperature difference (10 - TM(I)) programmed in DO loop 1 of DERV).

(5) Summary and Conclusions

The two preceding examples (for the systems of Figures 1a, 1b and 2a) are intended to illustrate the programming requirements for the application of eqs. (3), (6) and (8) to other problem systems. The requirements are generally:

(1) The problem system ODE/PDEs must be formulated, including initial and boundary conditions. Generally these equations will have as state variables (a) ρ (computed from the continuity equation (3)), (b) W (computed from the momentum equation (6)) and (c) ρu (computed from the energy equation (8)). If a component of the problem system can be considered perfectly mixed, e.g., the dewar of Figure 2a, then ODEs in t replace the PDEs.

(2) The use of boundary conditions for the momentum eq. (6) which define the flows at the ends of the component are recommended rather than attempting to use the momentum equation to compute the terminal flows.

(3) The combination of upwinding the continuity and energy balances (eqs. (3) and (8)), and centering the momentum balance (eq. (6)) has been effective in computing a solution to the three PDEs for the problems considered so far. An alternative would be to also upwind the momentum balance, which would add additional smoothing (but also, probably additional numerical diffusion). Also, higher order approximations to the spatial derivatives (in z) in eqs. (3), (6) and (8) could be considered (to possibly better resolve moving fronts).

(4) The formulation of (3) requires the choice of the smoothing parameter ϵ, which is done by trial and error. Ideally, a value of ϵ is selected which will smooth the solutions (to eliminate unrealistic oscillations), but will not significantly alter the form or shape of the solution; the resolution of moving fronts is an important consideration in the choice of ϵ. Another important parameter is the number of grid points used to achieve adequate spatial resolution. The interplay of ϵ and the grid spacing is demonstrated by the plots of Figure 1c to 1h.

In summary, the user is required to write systems of ODE/PDEs and code them, using the guidelines presented in the preceding discussion. These requirements in general are not easy to implement, and our hope is that the preceding discussion, particularly the examples, will provide a general approach and methodology for the analysis of new problem systems. This approach is fundamental, however, (it has as the starting point the conservation principles embodied in eqs. (3), (6) and (8)), and broadly applicable to a spectrum of problem systems.

Also, unfortunately, writing systems of equations and coding their numerical solution does not guarantee in advance that solutions can be computed. Unexpected numerical problems will almost certainly develop, and experimentation with the code will generally be required before acceptable solutions are computed; thus the success of this approach to the analysis of convective systems with heat transfer is dependent to some extent on the background and experience of the analyst(s). We have attempted here to provide only a starting point, and to present an approach which, at least to this point in time, represents an effective method for computing solutions to problems in fluid transients with significant energy effects.

Also, we recommend an "incremental approach" to the writing of codes for new problem systems, e.g., start with a simplified version of the problem such as using ideal gas properties in place of real gas properties and isothermal conditions (by using only continuity and momentum balances). Once the simplified problem is solved satisfactorily, the code can be extended incrementally (e.g., add real gas properties, then add the energy equation with no heat inputs, then add heat inputs, etc.). Our experience has been that this incremental approach is much more likely to lead to a final code that will analyze the problem system of interest than to start with the most general problem, and include in the initial code all conceivably important effects.

Finally, we offer the opinion that this process of modeling and simulation can probably not be fully "automated" in the sense of producing a computer code that requires only some simple component connection and parameter information to simulate a new problem system. Our experience has indicated that each new problem system requires as the starting point the formulation of a mathematical model, e.g., a system of ODEs, PDEs and algebraic equations, followed by the detailed coding of the model, testing of the code with simplified (special case) problems, and then finally the use of the code to analyze the problem system of interest. We would be interested in the experience of others concerning this matter of "computerizing" the analysis of new problem systems.

To conclude, additional information on numerical methods for problems with moving fronts is given in the extensive surveys of Anderson et al (1984) and Finlayson (1992). We anticipate that the development of numerical algorithms for the solution of equations which model convection and heat transfer will continue as an active area of research; one important element of this continuing research will be the impact of new computrer architectures which should facilitate large scale scientific computation.

(6) References

Anderson, D. A., J. C. Tannehill and R. H. Pletcher, (1984), Computational Fluid Mechanics and Heat Transfer, Hemisphere Pub. Corp., NY

Arp, V. D., and R. D. McCarty, (1989), Thermophysical Properties of Helium-4 From 0.8 to 1500 K with Pressures to 2000 MPa, NIST Technical Note PB90-183351, November, 1989

Carcagno, R. H., and W. E. Schiesser, (1992), "Helium Venting Computer Simulation During an SSC Dipole Quench", *Advances in Cryogenic Engineering*, R. W. Fast (ed.), v. 37, Part A, pp 709-718, Plenum Press, NY

Carcagno, R. H., (1993), private communication

Finlayson, B. A., (1992), Numerical Methods for Problems with Moving Fronts, Ravenna Park Publishing, Seattle, WA

Forsythe, G. E., M. A. Malcolm and C. B. Moler, (1977), "Computer Methods for Mathematical Computations", Prentice Hall, Englewood Cliffs, NJ

Jang, J. H., A. Faghri, and W. S. Chang, (1991), "Analysis of the One-dimensional Transient Compressible Vapor Flow in Heat Pipes", *Int. J. Heat and Mass Trans.*, v. 38, no. 8, pp. 2029-2037

Schiesser, W. E., (1994), Computational Mathematics in Engineering and Applied Science, CRC Press, Boca Raton, FL

Wong, R. L., (1989), "Program CICC, Flow and Heat Transfer in Cable In Conduit Conductors - Equations and Verification", *LLNL Report UCID*-21733, May 22, 1989

Wong, R. L., (1990a), "Program CICC, Flow and Heat Transfer in Cable In Conduit Conductors - User's Manual", *LLNL Report UCID*-21941, February 14, 1990

Wong, R. L., (1990b), "Computations of Quenching and Stability in a CICC Conductor", *LLNL Report UCRL-JC*-103598, September 21, 1990

Wong, R. L., C. T. Yeaw, S. S. Shen and J. R. Miller, (1991), "Numerical Formulation of the Stability in a Cable-in-Conduit Conductor Developed for Fusion-Magnet Applications", *LLNL Report UCRL-JC*-107258, September 25, 1991

Yeaw, C. T., and R. L. Wong, (1992), "Numerical Simulation of the Stability in Long Cable-in-Conduit Conductors for Fusion Magnets", *LLNL Report UCRL-JC*-109337, March 5, 1992

Appendix 1a

Subroutine DERV for the System of Figure 1b

```
      SUBROUTINE DERV
C...
C...  SUBROUTINE DERV COMPUTES THE TEMPORAL DERIVATIVES OF THE MODEL
C...  ODE/PDES
C...
C...  THIS CODE FOR THE HELIUM DYNAMICS IN AN 8 M LINE IS CONSTRUCTED
C...  FROM THE FOLLOWING SUBROUTINES
C...
C...     INITAL      SETS THE INITIAL CONDITIONS FOR THE MODEL ORDINARY
C...                 DIFFERENTIAL EQUATIONS (ODES) AND PARTIAL DIFFER-
C...                 ENTIAL EQUATIONS (PDES).
C...
C...     DERV        COMPUTES THE DERIVATIVES OF THE MODEL ODE/PDES.
C...                 ALGEBRA IS ALSO PERFORMED IN THIS SUBROUTINE.
C...
C...     PRINT       PRINTS AND PLOTS THE NUMERICAL SOLUTION.
C...
C...  THESE THREE SUBROUTINES ALSO CALL SUBORDINATE ROUTINES, E.G., FOR
C...  THE EVALUATION OF THERMODYNAMIC PROPERTIES, PLOTTING OF THE SOLU-
C...  TIONS.  LINKAGE BETWEEN ROUTINES IS BY COMMON (DESCRIBED BELOW)
C...  AND ARGUMENTS.
C...
C...  IN ADDITION TO THESE ROUTINES WHICH ARE SPECIFIC TO THE PROBLEM,
C...  LIBRARY ROUTINES ARE ALSO USED, E.G., SUBROUTINE RKF45 FOR THE
C...  RUNGE KUTTA INTEGRATION OF THE MODEL ODES.  A GENERIC MAIN PRO-
C...  GRAM CALLS THE VARIOUS SUBROUTINES.
C...
C...  DEFINE THE CONTINUITY, MOMENTUM AND ENERGY BALANCES ON A SPATIAL
C...  GRID OF NM POINTS.  UNITS ARE MKS (SI) EXCEPT FOR PRESSURE WHICH
C...  IS IN ATM.
C...
C...     COMMON/T/
C...
C...        T         INDEPENDENT VARIABLE, TIME (S)
C...
C...        NSTOP     STOPPING VARIABLE (SET TO A NONZERO VALUE WILL
C...                  TERMINATE THE CURRENT RUN)
C...
C...        NORUN     RUN COUNTER (SET TO 1 DURING THE FIRST RUN, 2
C...                  DURING THE SECOND RUN, ETC., AND CAN THEREFORE
C...                  BE USED TO CHANGE THE CODE FROM RUN TO RUN)
C...
C...     COMMON/Y/
C...
C...        WM        MASS FLOW RATE (KG/M**2-S)
```

```
C...
C...        RHOM      FLUID MASS DENSITY (KG/M**3)
C...
C...        UMG       FLUID INTERNAL ENERGY-DENSITY PRODUCT (J/M**3)
C...
C...     COMMON/F/
C...
C...        WMT       TEMPORAL DERIVATIVE OF WM, I.E., THE FLUID MOMEN-
C...                  TUM PER UNIT VOLUME (KG/M**2-S**2)
C...
C...        RHOMT     TEMPORAL DERIVATIVE OF RHO (KG/M**3-S)
C...
C...        UMGT      TEMPORAL DERIVATIVE OF UMG (J/M**3-S)
C...
C...     COMMON/V/
C...
C...        UM        FLUID INTERNAL ENEREGY PER UNIT MASS (J/KG)
C...
C...        PM        FLUID PRESSURE (ATM)
C...
C...        TM        FLUID TEMPERATURE (K)
C...
C...        HM        FLUID ENTHALPY PER UNIT MASS (J/KG)
C...
C...     COMMON/C/
C...
C...        ZL        LEFT HAND SPATIAL COORDINATE (M)
C...
C...        ZR        RIGHT HAND SPATIAL COORDINATE (M)
C...
C...        DZ        GRID SPACING (M)
C...
C...        Z         SPATIAL GRID (RUNNING FROM ZR TO ZL WITH SPACING
C...                  DZ) (M)
C...
C...        TMI       INITIAL FLUID TEMPERATURE (K)
C...
C...        PMI       INITIAL FLUID PRESSURE (ATM)
C...
C...        WMI       INITIAL FLUID MASS FLOW RATE (KG/M**2-S)
C...
C...        QM        VOLUMETRIC HEAT INPUT TO THE FLUID ALONG THE
C...                  SPATIAL GRID (J/M**3-S)
C...
C...     COMMON/I/
C...
C...        IP        INTEGER COUNTER USED FOR PRINTING AND PLOTTING
C...                  IN SUBROUTINE PRINT
C...
C...     COMMON/IMODEL/
C...
```

```
C...          IMODEL   INTEGER VARIABLE TO SELECT THE THERMODYNAMIC
C...                   MODEL (ALLOWABLE VALUES ARE 1 TO 4)
C...
C...  DOUBLE PRECISION CODING IS USED
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=33)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM),  TERM(5)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
C...  FOR THE MAGNETS
C...
C...  THE THERMODYNAMIC FUNCTIONS ARE
C...
C...     PRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               PRESSURE
C...
C...     TRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               TEMPERATURE
C...
C...     HRHOU     DENSITY-INTERNAL ENERGY FLASH TO COMPUTE THE FLUID
C...               ENTHALPY
      DO 1 I=1,NM
         UM(I)=UMG(I)/RHOM(I)
         PM(I)=PRHOU(RHOM(I),UM(I))
         TM(I)=TRHOU(RHOM(I),UM(I))
         HM(I)=HRHOU(RHOM(I),UM(I))
C...
C...  THE HEAT INPUT CAN BE SPECIFIED AS A FUNCTION OF POSITION Z (I.E.,
C...  GRID INDEX I)
         IF((I.GE.1).AND.(I.LE.NM-5))THEN
C...
C...        THE FOLLOWING Q IS A FUNCTION OF TIME, T, WITH A TIME
C...        CONSTANT OF 0.5 S AND THE TEMPERATURE DIFFERENCE (10 - TM).
C...        ANOTHER POSSIBILITY WOULD BE TO INCLUDE A HEAT CONDUCTION
C...        MODEL TO DEFINE Q
C...        QM(I)=1.0D+05*(1.0D0-DEXP(-T/0.5D0))*(10.0D0-TM(I))
            QM(I)=0.0D0
         ELSE IF(I.GT.NM-5)THEN
            QM(I)=0.0D0
         END IF
1     CONTINUE
C...
C...  BOUNDARY CONDITIONS
C...
```

```
C...  LEFT END, Z = ZL
         PLEND=4.0D0
C...
C...     PRESSURE DROP
         DP=PM(1)-PLEND
         IF(DP.LE.0.0D0)THEN
C...
C...        FLOW RATE FOR NEGATIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS POSITIVE, AND PROPORTIONAL TO THE SQUARE ROOT OF THE
C...        PRESSURE DROP.  THE FLOW RATE ALSO CHANGES DYNAMICALLY
C...        WITH A TIME CONSTANT OF 0.01 S
C...        WM(1)= 300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DABS(DP))
            WM(1)=0.0D0
         ELSE
C...
C...        FLOW RATE FOR POSITIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS NEGATIVE
C...        WM(1)=-300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DP)
            WM(1)=0.0D0
         END IF
C...
C...  RIGHT END, Z = ZR
         PREND=3.8D0
C...
C...     PRESSURE DROP
         DP=PM(NM)-PREND
         IF(DP.LE.0.0D0)THEN
C...
C...        FLOW RATE FOR NEGATIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS NEGATIVE, AND PROPORTIONAL TO THE SQUARE ROOT OF THE
C...        PRESSURE DROP.  THE FLOW RATE ALSO CHANGES DYNAMICALLY
C...        WITH A TIME CONSTANT OF 0.01 S
            WM(NM)=-300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DABS(DP))
         ELSE
C...
C...        FLOW RATE FOR POSITIVE PRESSURE DROP.  NOTE THAT THE FLOW
C...        IS POSITIVE
            WM(NM)= 300.0D0*(1.0-DEXP(-T/0.01D0))*DSQRT(DP)
         END IF
C...
C...  PDES FOR THE MAGNETS
      DO 11 I=1,NM
C...
C...  LEFT END
      IF(I.EQ.1)THEN
C...
C...     MOMENTUM BALANCE DOES NOT SET THE FLOW RATE AT THE LEFT END
C...     (THE FLOW RATE IS SET BY THE PRECEDING BOUNDARY CONDITION)
         WMT(I)= 0.0D0
C...
C...     CONTINUITY BALANCE IS UPWINDED
```

```
         RHOMT(I)=-(WM(I+1)-WM(I))/DZ
C...
C...     ENERGY BALANCE IS UPWINDED
         UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+QM(I)
C...
C...  RIGHT END
      ELSE IF(I.EQ.NM)THEN
C...
C...     MOMENTUM BALANCE DOES NOT SET THE FLOW RATE AT THE RIGHT END
C...     (THE FLOW RATE IS SET BY THE PRECEDING BOUNDARY CONDITION)
         WMT(I)= 0.0D0
C...
C...     CONTINUITY BALANCE IS UPWINDED
         RHOMT(I)=-(WM(I)-WM(I-1))/DZ
C...
C...     ENERGY BALANCE IS UPWINDED
         UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+QM(I)
C...
C...  INTERIOR POINTS
      ELSE
C...
C...     CONVECTIVE MOMENTUM FORCE
         GIM1=WM(I-1)**2/RHOM(I-1)
         GI  =WM(I  )**2/RHOM(I  )
         GIP1=WM(I+1)**2/RHOM(I+1)
         CMF=((GIP1+GI)/2.0D0-(GIM1+GI)/2.0D0)/DZ
C...
C...     FRICTIONAL FORCE
         FF=0.02D0
         DC=0.03D0
         FT=(FF/(2.0D0*DC))*DABS(WM(I))*WM(I)/RHOM(I)
C...
C...     PRESSURE FORCE
         DPDZ=1.01325D+05*((PM(I+1)+PM(I))/2.0D0
     +                    -(PM(I-1)+PM(I))/2.0D0)/DZ
C...
C...     LINEAR SPATIAL FILTERING
         EPS=1.0D+04
         DVDZ=EPS*(WM(I+1)/RHOM(I+1)-2.0D0*WM(I)/RHOM(I)
     +            +WM(I-1)/RHOM(I-1))/(DZ**2)
C...
C...     MOMENTUM BALANCE (DM/DT = SUM FORCES) IS CENTERED
         WMT(I)=-CMF-FT-DPDZ+DVDZ
C...
C...  STORE THE INDIVIDUAL TERMS IN THE MOMENTUM BALANCE AT A PART-
C...  ICULAR GRID POINT FOR PRINTING IN SUBROUTINE PRINT
         IF(I.EQ.NM-1)THEN
            TERM(1)=-CMF
            TERM(2)=-FT
            TERM(3)=-DPDZ
            TERM(4)=DVDZ
```

```
            TERM(5)=WMT(I)
         END IF
C...
C...  CONTINUITY BALANCE IS UPWINDED
C...
C...     NEGATIVE FLOW
         IF(WM(I).LT.0.0D0)THEN
         RHOMT(I)=-(WM(I+1)-WM(I))/DZ
C...
C...     POSITIVE FLOW
         ELSE IF(WM(I).GE.0.0D0)THEN
         RHOMT(I)=-(WM(I)-WM(I-1))/DZ
         END IF
C...
C...  ENERGY BALANCE IS UPWINDED
C...
C...     AXIAL CONDUCTION
         DIF=1.0D+03
         DTDZ=DIF*(TM(I+1)-2.0D0*TM(I)+TM(I-1))/(DZ**2)
C...
C...     NEGATIVE FLOW
         IF(WM(I).LT.0.0D0)THEN
         UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+DTDZ+QM(I)
C...
C...     POSITIVE FLOW
         ELSE IF(WM(I).GE.0.0D0)THEN
         UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+DTDZ+QM(I)
         END IF
      END IF
C...
C...  NEXT INTERIOR GRID POINT
11    CONTINUE
      RETURN
      END
```

Appendix 1b

Subroutine INITAL for the System of Figure 1b

```
      SUBROUTINE INITAL
C...
C...  SUBROUTINE INITAL SETS THE INITIAL CONDITIONS FOR THE MODEL
C...  ODE/PDES
C...
C...  THIS COMMON AREA IS THE SAME AS IN SUBROUTINE DERV, AND IS
C...  USED IN EACH OF THE MODEL ROUTINES
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=33)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  SELECT THE THERMO MODEL
C...
C...     IMODEL = 3 - MODIFIED MARTIN EQUATION (VAN DER WAALS FAMILY)
C...
         IMODEL=3
C...
C...  LOWER AND UPPER LIMITS OF Z
      ZL=0.0D0
      ZR=8.0D0
C...
C...  GRID SPACING
      DZ=(ZR-ZL)/DFLOAT(NM-1)
C...
C...  AXIAL GRID
      DO 11 I=1,NM
         Z(I)=ZL+DFLOAT(I-1)*DZ
11    CONTINUE
C...
C...  INITIAL FLOW RATE
      WMI=0.0D0
C...
C...  INITIAL TEMPERATURE
      TMI=4.0D0
C...
C...  INITIAL PRESSURE
      PMI=4.0D0
C...
C...  INITIAL CONDITIONS (FOR THE 3 X NM DEPENDENT VARIABLES IN
```

```
C...  COMMON/Y/).  RHOPT IS A FUNCTION FOR A PRESSURE-TEMPERATURE
C...  FLASH THAT RETURNS THE FLUID DENSITY.  UPT IS A FUNCTION FOR
C...  A PRESSURE-TEMPERATURE FLASH THAT RETURNS THE INTERNAL ENERGY.
C...  NOTE THAT DEPENDENT VARIABLE UMG IS THE PRODUCT OF THE FLUID
C...  INTERNAL ENERGY AND DENSITY (I.E., THE VOLUMETRIC INTERNAL
C...  ENERGY)
      DO 1 I=1,NM
         WM(I)=WMI
       RHOM(I)=RHOPT(PMI,TMI)
        UMG(I)=  UPT(PMI,TMI)*RHOM(I)
1     CONTINUE
C...
C...  INITIAL DERIVATIVES
      CALL DERV
C...
C...  INITIALIZE COUNTER FOR PLOTTED SOLUTION
      IP=0
      RETURN
      END
```

Appendix 1c

Subroutine PRINT for the System of Figure 1b

```
      SUBROUTINE PRINT(NI,NO)
C...
C...  SUBROUTINE PRINT PRINTS THE NUMERICAL SOLUTION AND PLOTS THE
C...  SOLUTION BY CALLS TO PLOTTD
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=33)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM),   TERM(5)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  OPEN A FILE FOR PRINTING THE INDIVIDUAL TERMS IN THE MOMENTUM
C...  BALANCE
      OPEN(8,FILE='TERMS',STATUS='UNKNOWN')
C...
C...  INCREMENT THE COUNTER FOR THE PLOTTED OUTPUT
      IP=IP+1
C...
C...  MONITOR THE OUTPUT
      WRITE(*,*)IP
C...
C...  WRITE THE NUMERICAL SOLUTION
      WRITE(NO,1)T,(Z(I),WM(I),PM(I),TM(I),RHOM(I),QM(I),I=1,NM)
1     FORMAT(//,' Time = ',F7.3,//,
     +  '          z','          W','          P',
     +  '          T','        rho',
     +  '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
C...
C...  WRITE THE INDIVIDUAL TERMS IN THE MOMENTUM BALANCE AT AN
C...  INTERIOR GRID POINT (SELECTED IN DERV)
      WRITE(8,2)T,(TERM(I),I=1,5)
2     FORMAT(   ' T = ',F5.3,/,
     +          '     -D(W**2/RHO)/DZ = ',F9.2,/,
     +          '                  -FT = ',F9.2,/,
     +          '               -DP/DZ = ',F9.2,/,
     +          '                DV/DZ = ',F9.2,/,
     +          '               DWM/DT = ',F9.2,//)
C...
C...  PLOT THE SOLUTION
      CALL PLOTTD
      RETURN
      END
```

Appendix 1d

Subroutine PLOTTD for Plotting the Solution from INITAL, DERV and PRINT

```
      SUBROUTINE PLOTTD
C...
C...  SUBROUTINE PLOTTD WRITES A FILE WITH THE NUMERICAL SOLUTION
C...  WHICH IS THEN READ BY TOP DRAWER FOR PLOTTING
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=33)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  OPEN A SERIES OF FILES FOR TOP DRAWER PLOTTING
C...
C...     W1.TOP      FLOW RATE AS A FUNCTION OF Z WITH T AS A PARAMETER
C...
C...     P1.TOP      PRESSURE AS A FUNCTION OF Z WITH T AS A PARAMETER
C...
C...     T1.TOP      TEMPERATURE AS A FUNCTION OF Z WITH T AS A PARAMETER
C...
C...     R1.TOP      DENSITY AS A FUNCTION OF Z WITH T AS A PARAMETER
C...
      IF(IP.EQ.1)THEN
         OPEN(1,FILE='W1.TOP',STATUS='UNKNOWN')
         OPEN(2,FILE='P1.TOP',STATUS='UNKNOWN')
         OPEN(3,FILE='T1.TOP',STATUS='UNKNOWN')
         OPEN(4,FILE='R1.TOP',STATUS='UNKNOWN')
      END IF
C...
C...  SCALE THE AXES FOR EACH PLOT
      IF(IP.EQ.1)THEN
C...
C...     FLOW RATE
         WRITE(1,1)
1        FORMAT(' SET LIMITS X FROM 0 TO 8 Y FROM 0 TO 75',/,
     +          ' SET FONT DUPLEX')
C...
C...     PRESSURE
         WRITE(2,2)
2        FORMAT(' SET LIMITS X FROM 0 TO 8 Y FROM 3.7 TO 4',/,
     +          ' SET FONT DUPLEX')
C...
C...     TEMPERATURE
```

```
         WRITE(3,3)
3        FORMAT(' SET LIMITS X FROM 0 TO 8 Y FROM 3.95 TO 4',/,
     +           ' SET FONT DUPLEX')
C...
C...     DENSITY
         WRITE(4,4)
4        FORMAT(' SET LIMITS X FROM 0 TO 8 Y FROM 139 TO 140',/,
     +           ' SET FONT DUPLEX')
      END IF
C...
C...  WRITE THE NUMERICAL SOLUTION FOR SUBSEQUENT TOP DRAWER PLOTTING
      DO 5 I=1,NM
C...
C...     FLOW RATE
         WRITE(1,6)Z(I),WM(I)
C...
C...     PRESSURE
         WRITE(2,6)Z(I),PM(I)
C...
C...     TEMPERATURE
         WRITE(3,6)Z(I),TM(I)
C...
C...     DENSITY
         WRITE(4,6)Z(I),RHOM(I)
6        FORMAT(F10.3,F10.5)
5     CONTINUE
C...
C...  CONNECT POINTS
         WRITE(1,7)
         WRITE(2,7)
         WRITE(3,7)
         WRITE(4,7)
7        FORMAT(' JOIN 1')
C...
C...  LEGEND, AXES LABELS FOR EACH PLOT WITH T = 0, 0.01,..., 0.07
      IF(IP.EQ.8)THEN
C...
C...     FLOW RATE
         WRITE(1,8)
8        FORMAT(
     +      ' TITLE 4.75 9.75 "Fig. 2c: Mass flow (kg/m**2-s) vs z"'
     +   ,/,' TITLE 5.5  9.25 "t = 0, 0.01, 0.02,..., 0.07 s"'
     +   ,/,' TITLE LEFT "       W(z,t)"'
     +   ,/,' TITLE BOTTOM "z (m)"')
C...
C...     PRESSURE
         WRITE(2,9)
9        FORMAT(
     +      ' TITLE 5.25 9.75 "Fig. 1c: Pressure (atm) vs z"'
     +   ,/,' TITLE 5.25 9.25 "t = 0, 0.01, 0.02,..., 0.07 s"'
     +   ,/,' TITLE LEFT "       P(z,t)"'
```

```
     +    ,/,' TITLE BOTTOM "z (m)"')
C...
C...      TEMPERATURE
          WRITE(3,10)
10        FORMAT(
     +       ' TITLE 5    9.75 "Fig. 3c: Temperature (k) vs z"'
     +    ,/,' TITLE 5    9.25 "t = 0, 0.01, 0.02,..., 0.07 s"'
     +    ,/,' TITLE LEFT "        T(z,t)"'
     +    ,/,' TITLE BOTTOM "z (m)"')
C...
C...      DENSITY
          WRITE(4,11)
11        FORMAT(
     +       ' TITLE 4.75 9.75 "Fig. 4c: Density (kg/m**3) vs z"'
     +    ,/,' TITLE 5    9.25 "t = 0, 0.01, 0.02,..., 0.07 s"'
     +    ,/,' TITLE LEFT "      Rho(z,t)"'
     +    ,/,' TITLE BOTTOM "z (m)"')
      END IF
      RETURN
      END
```

Appendix 1e

Main Program (and Subordinate Routines) that calls Subroutine RKF45

```
      PROGRAM HELIUM
C...
C...  PROGRAM HELIUM CALLS: (1) SUBROUTINE INITAL TO DEFINE THE ODE
C...  INITAL CONDITIONS, (2) SUBROUTINE RKF45 TO INTEGRATE THE ODES,
C...  AND (3) SUBROUTINE PRINT TO PRINT THE SOLUTION.
C...
C...  THE FOLLOWING CODING IS FOR 1000 ODES.  IF MORE ODES ARE TO BE INTE-
C...  GRATED, ALL OF THE 1000'S SHOULD BE CHANGED TO THE REQUIRED NUMBER
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/T/          T,      NSTOP,      NORUN
     1       /Y/    Y(1000)
     2       /F/    F(1000)
C...
C...  THE NUMBER OF DIFFERENTIAL EQUATIONS IS IN COMMON/N/ FOR USE IN
C...  SUBROUTINE FCN
      COMMON/N/       NEQN
C...
C...  COMMON AREA TO PROVIDE THE INPUT/OUTPUT UNIT NUMBERS TO OTHER
C...  SUBROUTINES
      COMMON/IO/        NI,         NO
C...
C...  ABSOLUTE DIMENSIONING OF THE ARRAYS REQUIRED BY RKF45
      DIMENSION YV(1000), WORK(7000), IWORK(5)
C...
C...  EXTERNAL THE DERIVATIVE ROUTINE CALLED BY RKF45
      EXTERNAL FCN
C...
C...  ARRAY FOR THE TITLE (FIRST LINE OF DATA), CHARACTERS  END OF RUNS
      CHARACTER TITLE(20)*4, ENDRUN(3)*4
C...
C...  DEFINE THE CHARACTERS  END OF RUNS
      DATA ENDRUN/'END ','OF R','UNS '/
C...
C...  DEFINE THE INPUT/OUTPUT UNIT NUMBERS
      NI=5
      NO=6
C...
C...  OPEN INPUT AND OUTPUT FILES
C...  OPEN(NI,FILE='DATA',  STATUS='OLD')
C...  OPEN(NO,FILE='OUTPUT',STATUS='NEW')
C...
C...  MODIFY I/O STATEMENTS FOR SSCVX1
      OPEN(NI,FILE='DATA',  STATUS='UNKNOWN')
      OPEN(NO,FILE='OUTPUT',STATUS='UNKNOWN')
C...
C...  INITIALIZE THE RUN COUNTER
      NORUN=0
```

```
C...
C...  BEGIN A RUN
1     NORUN=NORUN+1
C...
C...  INITIALIZE THE RUN TERMINATION VARIABLE
      NSTOP=0
C...
C...  READ THE FIRST LINE OF DATA
      READ(NI,1000,END=999)(TITLE(I),I=1,20)
C...
C...  TEST FOR  END OF RUNS  IN THE DATA
      DO 2 I=1,3
      IF(TITLE(I).NE.ENDRUN(I))GO TO 3
2     CONTINUE
C...
C...  AN END OF RUNS HAS BEEN READ, SO TERMINATE EXECUTION
999   STOP
C...
C...  READ THE SECOND LINE OF DATA
3     READ(NI,1001,END=999)TO,TF,TP
C...
C...  READ THE THIRD LINE OF DATA
      READ(NI,1002,END=999)NEQN,ERROR
C...
C...  PRINT A DATA SUMMARY
      WRITE(NO,1003)NORUN,(TITLE(I),I=1,20),
     1                    TO,TF,TP,
     2                    NEQN,ERROR
C...
C...  INITIALIZE TIME
      T=TO
C...
C...  SET THE INITIAL CONDITIONS
      CALL INITAL
C...
C...  SET THE INITIAL DERIVATIVES (FOR POSSIBLE PRINTING)
      CALL DERV
C...
C...  PRINT THE INITIAL CONDITIONS
      CALL PRINT(NI,NO)
C...
C...  SET THE INITIAL CONDITIONS FOR SUBROUTINE RKF45
      TV=TO
      DO 5 I=1,NEQN
      YV(I)=Y(I)
5     CONTINUE
C...
C...  SET THE PARAMETERS FOR SUBROUTINE RKF45
      RELERR=ERROR
      ABSERR=ERROR
      IFLAG=1
```

```
      TOUT=TO+TP
C...
C...  CALL SUBROUTINE RKF45 TO START THE SOLUTION FROM THE INITIAL
C...  CONDITION (IFLAG = 1) OR COMPUTE THE SOLUTION TO THE NEXT PRINT
C...  POINT (IFLAG = 2)
4     CALL RKF45(FCN,NEQN,YV,TV,TOUT,RELERR,ABSERR,IFLAG,WORK,IWORK)
C...
C...  PRINT THE SOLUTION AT THE NEXT OUTPUT POINT
      T=TV
      DO 6 I=1,NEQN
      Y(I)=YV(I)
6     CONTINUE
      CALL DERV
      CALL PRINT(NI,NO)
C...
C...  TEST FOR AN ERROR CONDITION
      IF(IFLAG.NE.2)THEN
C...
C...     PRINT A MESSAGE INDICATING AN ERROR CONDITION
         WRITE(NO,1004)IFLAG
C...
C...     GO ON TO THE NEXT RUN
         GO TO 1
      END IF
C...
C...  CHECK FOR A RUN TERMINATION
      IF(NSTOP.NE.0)GO TO 1
C...
C...  CHECK FOR THE END OF THE RUN
      TOUT=TV+TP
      IF(TV.LT.(TF-0.5D0*TP))GO TO 4
C...
C...  THE CURRENT RUN IS COMPLETE, SO GO ON TO THE NEXT RUN
      GO TO 1
C...
C...  ******************************************************************
C...
C...  FORMATS
C...
1000  FORMAT(20A4)
1001  FORMAT(3D10.0)
1002  FORMAT(I5,20X,D10.0)
1003  FORMAT(1H1,
     1 ' RUN NO. - ',I3,2X,20A4,//,
     2 ' INITIAL T - ',D10.3,//,
     3 '   FINAL T - ',D10.3,//,
     4 '   PRINT T - ',D10.3,//,
     5 ' NUMBER OF DIFFERENTIAL EQUATIONS - ',I3,//,
     6 ' MAXIMUM INTEGRATION ERROR - ',D10.3,//,
     7 1H1)
1004  FORMAT(1H ,//,' IFLAG = ',I3,//,
```

```
     1 ' INDICATING AN INTEGRATION ERROR, SO THE CURRENT RUN'      ,/,
     2 ' IS TERMINATED.  PLEASE REFER TO THE DOCUMENTATION FOR'    ,/,
     3 ' SUBROUTINE',//,25X,'RKF45',//,
     4 ' FOR AN EXPLANATION OF THESE ERROR INDICATORS'              )
      END

      SUBROUTINE FCN(TV,YV,YDOT)
C...
C...  SUBROUTINE FCN IS AN INTERFACE ROUTINE BETWEEN SUBROUTINES RKF45
C...  AND DERV
C...
C...  NOTE THAT THE SIZE OF ARRAYS Y AND F IN THE FOLLOWING COMMON AREA
C...  IS ACTUALLY SET BY THE CORRESPONDING COMMON STATEMENT IN MAIN
C...  PROGRAM HELIUM
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/       Y(1)
     2      /F/       F(1)
C...
C...  THE NUMBER OF DIFFERENTIAL EQUATIONS IS AVAILABLE THROUGH COMMON
C...  /N/
      COMMON/N/       NEQN
C...
C...  ABSOLUTE DIMENSION THE DEPENDENT VARIABLE, DERIVATIVE VECTORS
      DIMENSION YV(1000), YDOT(1000)
C...
C...  TRANSFER THE INDEPENDENT VARIABLE, DEPENDENT VARIABLE VECTOR
C...  FOR USE IN SUBROUTINE DERV
      T=TV
      DO 1 I=1,NEQN
      Y(I)=YV(I)
1     CONTINUE
C...
C...  EVALUATE THE DERIVATIVE VECTOR
      CALL DERV
C...
C...  TRANSFER THE DERIVATIVE VECTOR FOR USE BY SUBROUTINE RKF45
      DO 2 I=1,NEQN
      YDOT(I)=F(I)
2     CONTINUE
      RETURN
      END

      DOUBLE PRECISION FUNCTION DFLOAT(I)
      DFLOAT=DBLE(FLOAT(I))
      RETURN
      END
```

Appendix 1f

Functions UPT, RHOPT, PRHOU, TRHOU, HRHOU

UPT - Internal energy from pressure and temperature

RHOPT - Density from pressure and temperature

PRHOU - Pressure from density and internal energy

TRHOU - Temperature from density and internal energy

HRHOU - Enthalpy from density and internal energy

Units: Internal energy, j/kg; pressure, atm; temperature, k;
density, kg/m^3; enthalpy (j/kg)

```
      DOUBLE PRECISION FUNCTION UPT(P,T)
C...
C...  INTERNAL ENERGY FROM PRESSURE AND TEMPERATURE
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=2
      kd=0
      kh=0
      ku=1
      ks=0
      kcp=0
      call thermo(      t,      p,    vof,  ifase, ifcode,
     +                 kd,     kh,     ku,     ks,    kcp,
     +                 dt,     dv,     dl,
     +                 ht,     hv,     hl,
     +                 ut,     uv,     ul,
     +                 st,     sv,     sl,
     +                        cpv,    cpl,
     +              imodel,  ipert,  itdbg,    ker)
C...
C...  CONVERT INTERNAL ENERGY IN J/GM-MOL TO J/KG
      UPT=ut*1000.0d0/4.00d0
      RETURN
      END

      DOUBLE PRECISION FUNCTION RHOPT(P,T)
C...
C...  DENSITY FROM PRESSURE AND TEMPERATURE
C...
```

```
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=2
      kd=1
      kh=0
      ku=0
      ks=0
      kcp=0
      call thermo(      t,      p,    vof,  ifase, ifcode,
     +                 kd,     kh,     ku,     ks,    kcp,
     +                 dt,     dv,     dl,
     +                 ht,     hv,     hl,
     +                 ut,     uv,     ul,
     +                 st,     sv,     sl,
     +                        cpv,    cpl,
     +              imodel,  ipert,  itdbg,    ker)
C...
C...  CONVERT DENSITY IN GM-MOL/LITER TO KG/M**3
      RHOPT=4.0D0*dt
      RETURN
      END

      DOUBLE PRECISION FUNCTION PRHOU(RHO,U)
C...
C...  PRESSURE FROM DENSITY AND INTERNAL ENERGY
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=6
      kd=0
      kh=0
      ku=0
      ks=0
      kcp=0
C...
C...  CONVERT DENSITY IN KG/M**3 TO GM-MOL/LITER
      dt=RHO/4.0d0
C...
C...  CONVERT INTERNAL ENERGY IN J/KG TO J/GM-MOL
      ut=U*4.00d0/1000.0d0
      call thermo(      t,      p,    vof,  ifase, ifcode,
     +                 kd,     kh,     ku,     ks,    kcp,
     +                 dt,     dv,     dl,
     +                 ht,     hv,     hl,
     +                 ut,     uv,     ul,
     +                 st,     sv,     sl,
     +                        cpv,    cpl,
```

```
     +                imodel,   ipert,   itdbg,     ker)
      PRHOU=p
      RETURN
      END

      DOUBLE PRECISION FUNCTION TRHOU(RHO,U)
C...
C...  TEMPERATURE FROM DENSITY AND INTERNAL ENERGY
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=6
      kd=0
      kh=0
      ku=0
      ks=0
      kcp=0
C...
C...  CONVERT DENSITY IN KG/M**3 TO GM-MOL/LITER
      dt=RHO/4.0d0
C...
C...  CONVERT INTERNAL ENERGY IN J/KG TO J/GM-MOL
      ut=U*4.00d0/1000.0d0
      call thermo(       t,       p,     vof,  ifase, ifcode,
     +                  kd,      kh,      ku,     ks,    kcp,
     +                  dt,      dv,      dl,
     +                  ht,      hv,      hl,
     +                  ut,      uv,      ul,
     +                  st,      sv,      sl,
     +                           cpv,     cpl,
     +               imodel,   ipert,   itdbg,     ker)
      TRHOU=t
      RETURN
      END

      DOUBLE PRECISION FUNCTION HRHOU(RHO,U)
C...
C...  ENTHALPY FROM DENSITY AND INTERNAL ENERGY
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMMON/IMODEL/imodel
      integer itdbg(3)
      ifase=1
      ifcode=6
      kd=0
      kh=1
      ku=0
      ks=0
      kcp=0
```

```
C...
C...  CONVERT DENSITY IN KG/M**3 TO GM-MOL/LITER
      dt=RHO/4.0D0
C...
C...  CONVERT INTERNAL ENERGY IN J/KG TO J/GM-MOL
      ut=U*4.00d0/1000.0d0
      call thermo(      t,      p,    vof,  ifase, ifcode,
     +                 kd,     kh,     ku,     ks,    kcp,
     +                 dt,     dv,     dl,
     +                 ht,     hv,     hl,
     +                 ut,     uv,     ul,
     +                 st,     sv,     sl,
     +                        cpv,    cpl,
     +              imodel,  ipert,  itdbg,    ker)
C...
C...  CONVERT ENTHALPY IN J/GM-MOL TO J/KG
      HRHOU=ht*1000.0d0/4.00d0
      RETURN
      END
```

Appendix 2a

Subroutine DERV for the System of Figure 2a

```
      SUBROUTINE DERV
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=46,NR=46)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),   RHOM(NM),    UMG(NM),
     +                          RHOD,        UDG,
     +             WR(NR),   RHOR(NR),    URG(NR)
     2      /F/    WMT(NM),  RHOMT(NM),   UMGT(NM),
     +                         RHODT,       UDGT,
     +            WRT(NR),  RHORT(NR),   URGT(NR)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM),
     +                 UD,         PD,         TD,         HD,
     +             UR(NR),     PR(NR),     TR(NR),     HR(NR)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM),
     +      QD,        TDI,        PDI,         VD,         AC,
     +                 TRI,        PRI,        WRI,     QR(NR)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
C...  FOR THE MAGNETS
      DO 1 I=1,NM
         UM(I)=UMG(I)/RHOM(I)
         PM(I)=PRHOU(RHOM(I),UM(I))
         TM(I)=TRHOU(RHOM(I),UM(I))
         HM(I)=HRHOU(RHOM(I),UM(I))
         IF((I.GE.1).AND.(I.LE.NM-5))THEN
           QM(I)=1.0D+05*(1.0D0-DEXP(-T/0.5D0))*(10.0D0-TM(I))
         ELSE IF(I.GT.NM-5)THEN
           QM(I)=0.0D0
         END IF
1     CONTINUE
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, ENTHALPY, HEAT INPUT
C...  FOR THE RETURN
      DO 2 I=1,NR
         UR(I)=URG(I)/RHOR(I)
         PR(I)=PRHOU(RHOR(I),UR(I))
         TR(I)=TRHOU(RHOR(I),UR(I))
         HR(I)=HRHOU(RHOR(I),UR(I))
         QR(I)=0.0D0
2     CONTINUE
C...
C...  INTERNAL ENERGY, PRESSURE, TEMPERATURE, HEAT INPUT FOR THE DEWAR
      UD=UDG/RHOD
      PD=PRHOU(RHOD,UD)
```

```
      TD=TRHOU(RHOD,UD)
      QD=0.0D0
C...
C...  BOUNDARY CONDITIONS
C...
C...  MAGNETS, Z = ZL
         DP=PM(1)-PD
         IF(DP.LE.0.0D0)THEN
            WM(1)=0.0D0
         ELSE
            WM(1)=-300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
         END IF
C...
C...  MAGNETS, Z = ZR
         DP=PM(NM)-PR(1)
         IF(DP.LE.0.0D0)THEN
            WM(NM)=0.0D0
         ELSE
            WM(NM)= 300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
         END IF
C...
C...  RETURN, Z = ZL
         DP=PR(NR)-PD
         IF(DP.LE.0.0D0)THEN
            WR(NR)=0.0D0
         ELSE
            WR(NR)= 300.0D0*(1.0-DEXP(-T/0.1D0))*DSQRT(DP)
         END IF
C...
C...  RETURN, Z = ZR
         DP=PM(NM)-PR(1)
         IF(DP.LE.0.0D0)THEN
            WR(1)=0.0D0
         ELSE
            WR(1)=WM(NM)
         END IF
C...
C...  PDES FOR THE MAGNETS
      DO 11 I=1,NM
C...
C...    RIGHT END
        IF(I.EQ.NM)THEN
            WMT(I)=0.0D0
         RHOMT(I)=-(WM(I)-WM(I-1))/DZ
          UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ+QM(I)
C...
C...    LEFT END
        ELSE IF(I.EQ.1)THEN
            WMT(I)=0.0D0
         RHOMT(I)=-(WM(I+1)-WM(I))/DZ
          UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ+QM(I)
```

```
C...
C...     INTERIOR POINTS
         ELSE
C...
C...        MOMENTUM BALANCE
            GIM1=WM(I-1)**2/RHOM(I-1)
            GI  =WM(I  )**2/RHOM(I  )
            GIP1=WM(I+1)**2/RHOM(I+1)
            DVDZ=(WM(I+1)/RHOM(I+1)
     +          -2.0D0*WM(I)/RHOM(I)
     +          +WM(I-1)/RHOM(I-1))/(DZ**2)
            EPS=1.0D+05
            DPDZ=1.013D+05*((PM(I+1)+PM(I))/2.0D0
     +                     -(PM(I-1)+PM(I))/2.0D0)/DZ
            WMT(I)=-((GIP1+GI)/2.0D0
     +              -(GIM1+GI)/2.0D0)/DZ
     +              -DPDZ+EPS*DVDZ
C...
C...        MASS BALANCE
            IF(WM(I).LT.0.0D0)THEN
               RHOMT(I)=-(WM(I+1)-WM(I))/DZ
            ELSE IF(WM(I).GE.0.0D0)THEN
               RHOMT(I)=-(WM(I)-WM(I-1))/DZ
            END IF
C...
C...        ENERGY BALANCE
            DTDZ=(TM(I+1)-2.0D0*TM(I)+TM(I-1))/(DZ**2)
            DIF=1.0D+03
            IF(WM(I).LT.0.0D0)THEN
               UMGT(I)=-(WM(I+1)*HM(I+1)-WM(I)*HM(I))/DZ
     +                +QM(I)+DIF*DTDZ
            ELSE IF(WM(I).GE.0.0D0)THEN
               UMGT(I)=-(WM(I)*HM(I)-WM(I-1)*HM(I-1))/DZ
     +                +QM(I)+DIF*DTDZ
            END IF
         END IF
11     CONTINUE
C...
C...   PDES FOR THE RETURN
       DO 12 I=1,NR
C...
C...     RIGHT END
         IF(I.EQ.NR)THEN
             WRT(I)=0.0D0
           RHORT(I)=-(WR(I)-WR(I-1))/DZ
            URGT(I)=-(WR(I)*HR(I)-WR(I-1)*HR(I-1))/DZ+QR(I)
C...
C...     LEFT END
         ELSE IF(I.EQ.1)THEN
             WRT(I)=0.0D0
           RHORT(I)=-(WR(I+1)-WR(I))/DZ
```

```
            URGT(I)=-(WR(I+1)*HR(I+1)-WR(I)*HR(I))/DZ+QR(I)
C...
C...     INTERIOR POINTS
         ELSE
C...
C...        MOMENTUM BALANCE
            GIM1=WR(I-1)**2/RHOR(I-1)
            GI  =WR(I  )**2/RHOR(I  )
            GIP1=WR(I+1)**2/RHOR(I+1)
            DVDZ=(WR(I+1)/RHOR(I+1)
     +          -2.0D0*WR(I)/RHOR(I)
     +          +WR(I-1)/RHOR(I-1))/(DZ**2)
            EPS=1.0D+05
            DPDZ=1.013D+05*((PR(I+1)+PR(I))/2.0D0
     +                     -(PR(I-1)+PR(I))/2.0D0)/DZ
            WRT(I)=-((GIP1+GI)/2.0D0
     +              -(GIM1+GI)/2.0D0)/DZ
     +              -DPDZ+EPS*DVDZ
C...
C...        MASS BALANCE
            IF(WR(I).LT.0.0D0)THEN
               RHORT(I)=-(WR(I+1)-WR(I))/DZ
            ELSE IF(WR(I).GE.0.0D0)THEN
               RHORT(I)=-(WR(I)-WR(I-1))/DZ
            END IF
C...
C...        ENERGY BALANCE
            DTDZ=(TR(I+1)-2.0D0*TR(I)+TR(I-1))/(DZ**2)
            DIF=1.0D+03
            IF(WR(I).LT.0.0D0)THEN
               URGT(I)=-(WR(I+1)*HR(I+1)-WR(I)*HR(I))/DZ
     +                +QR(I)+DIF*DTDZ
            ELSE IF(WR(I).GE.0.0D0)THEN
               URGT(I)=-(WR(I)*HR(I)-WR(I-1)*HR(I-1))/DZ
     +                +QR(I)+DIF*DTDZ
            END IF
         END IF
12    CONTINUE
C...
C...  ODES FOR THE DEWAR
      RHODT=-(AC/VD)*WM( 1)
     +      +(AC/VD)*WR(NR)
       UDGT=-(AC/VD)*WM( 1)*HM( 1)
     +      +(AC/VD)*WR(NR)*HR(NR)
     +      +QD/VD
      RETURN
      END
```

Appendix 2b

Subroutine INITAL for the System of Figure 2a

```
      SUBROUTINE INITAL
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=46,NR=46)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),   UMG(NM),
     +                          RHOD,       UDG,
     +             WR(NR),  RHOR(NR),   URG(NR)
     2      /F/    WMT(NM), RHOMT(NM),  UMGT(NM),
     +                         RHODT,      UDGT,
     +            WRT(NR), RHORT(NR),  URGT(NR)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM),
     +                 UD,         PD,         TD,         HD,
     +             UR(NR),     PR(NR),     TR(NR),     HR(NR)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM),
     +      QD,        TDI,        PDI,         VD,         AC,
     +                 TRI,        PRI,        WRI,     QR(NR)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  SELECT THE THERMO MODEL
C...
C...     IMODEL = 3 - MODIFIED MARTIN EQUATION (VAN DER WAALS FAMILY)
C...
         IMODEL=3
C...
C...  LOWER AND UPPER LIMITS OF Z
      ZL=0.0D0
      ZR=90.0D0
C...
C...  MAGNET GRID SPACING
      DZ=(ZR-ZL)/DFLOAT(NM-1)
C...
C...  MAGNET AXIAL GRID
      DO 11 I=1,NM
         Z(I)=ZL+DFLOAT(I-1)*DZ
11    CONTINUE
C...
C...  DEWAR VOLUME
      VD=5.0D0
C...
C...  LINE CROSS SECTIONAL AREA
      AC=0.04D0
C...
C...  INITIAL MAGNET FLOW RATE
      WMI=1.0D0
C...
```

```
C...   INITIAL MAGNET TEMPERATURE
       TMI=4.0D0
C...
C...   INITIAL MAGNET PRESSURE
       PMI=4.0D0
C...
C...   INITIAL DEWAR TEMPERATURE
       TDI=4.5D0
C...
C...   INITIAL DEWAR PRESSURE
       PDI=4.0D0
C...
C...   INITIAL RETURN FLOW RATE
       WRI=1.0D0
C...
C...   INITIAL RETURN TEMPERATURE
       TRI=4.0D0
C...
C...   INITIAL RETURN PRESSURE
       PRI=1.0D0
C...
C...   MAGNET INITIAL CONDITIONS
       DO 1 I=1,NM
          WM(I)=WMI
        RHOM(I)=RHOPT(PMI,TMI)
         UMG(I)=  UPT(PMI,TMI)*RHOM(I)
1      CONTINUE
C...
C...   DEWAR INITIAL CONDITIONS
       RHOD=RHOPT(PDI,TDI)
        UDG=  UPT(PDI,TDI)*RHOD
C...
C...   RETURN INITIAL CONDITIONS
       DO 2 I=1,NR
          WR(I)=WRI
        RHOR(I)=RHOPT(PRI,TRI)
         URG(I)=  UPT(PRI,TRI)*RHOR(I)
2      CONTINUE
C...
C...   INITIAL DERIVATIVES
       CALL DERV
C...
C...   INITIALIZE COUNTER FOR PLOTTED SOLUTION
       IP=0
       RETURN
       END
```

Appendix 2c

Subroutine PRINT for the System of Figure 2a

```
      SUBROUTINE PRINT(NI,NO)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      PARAMETER (NM=46,NR=46)
      COMMON/T/          T,      NSTOP,      NORUN
     1      /Y/     WM(NM),  RHOM(NM),    UMG(NM),
     +                          RHOD,        UDG,
     +              WR(NR),  RHOR(NR),    URG(NR)
     2      /F/    WMT(NM), RHOMT(NM),   UMGT(NM),
     +                         RHODT,       UDGT,
     +             WRT(NR), RHORT(NR),   URGT(NR)
     3      /V/     UM(NM),     PM(NM),     TM(NM),     HM(NM),
     +                 UD,         PD,         TD,         HD,
     +              UR(NR),     PR(NR),     TR(NR),     HR(NR)
     4      /C/         ZL,         ZR,         DZ,      Z(NM),
     5       R,        TMI,        PMI,        WMI,     QM(NM),
     +      QD,        TDI,        PDI,         VD,         AC,
     +                 TRI,        PRI,        WRI,     QR(NR)
     6      /I/         IP
      COMMON/IMODEL/IMODEL
C...
C...  INCREMENT THE COUNTER FOR THE PLOTTED OUTPUT
      IP=IP+1
C...
C...  MONITOR THE OUTPUT
      WRITE(*,*)IP
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE MAGNETS
      WRITE(NO,1)T,(Z(I),WM(I),PM(I),TM(I),RHOM(I),QM(I),I=1,NM)
1     FORMAT(//,' Magnets, time = ',F7.3,//,
     +  '          z','          W','          P',
     +  '          T','        rho',
     +  '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE DEWAR
      WRITE(NO,2)PD,TD,RHOD,WM(1),QD
2     FORMAT(/,    ' Dewar',
     +  /,'  PD = ',F7.2,'  TD = ',F7.3,'  RHOD = ',F7.3,
     +    '  WM = ',F7.2,'  QD = ',F7.1)
C...
C...  WRITE THE NUMERICAL SOLUTION FOR THE RETURN
      WRITE(NO,3)T,(Z(I),WR(I),PR(I),TR(I),RHOR(I),QR(I),I=1,NR)
3     FORMAT(/,' Return, time = ',F7.3,/,
     +  '          z','          W','          P',
     +  '          T','        rho',
     +  '            Q',/,(2F10.2,2F10.3,F10.2,E12.3))
      RETURN
      END
```