

RECEIVED

OCT 10 1995

OSTI

---

**COBRA-SFS:  
A Thermal-Hydraulic Analysis  
Code for Spent Fuel Storage  
And Transportation Casks**

**T. E. Michener  
D. R. Rector  
J. M. Cuta**

**R. E. Dodge  
C. W. Enderlin**

---

**September 1995**

**Prepared for the U.S. Department of Energy  
under Contract DE-AC06-76RLO 1830**

**Pacific Northwest Laboratory  
Operated for the U.S. Department of Energy  
by Battelle Memorial Institute**





**COBRA-SFS:  
A Thermal-Hydraulic Analysis  
Code for Spent Fuel Storage  
And Transportation Casks**

Documentation for Cycle 2

T. E. Michener  
D. R. Rector  
J. M. Cuta  
R. E. Dodge  
C. W. Enderlin


September 1995

Prepared for  
the U.S. Department of Energy  
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory  
Richland, Washington 99352

**MASTER**

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED



## DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Battelle Memorial Institute, 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, or Battelle Memorial Institute. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

PACIFIC NORTHWEST LABORATORY  
*operated by*  
BATTELLE MEMORIAL INSTITUTE  
*for the*  
UNITED STATES DEPARTMENT OF ENERGY  
*under Contract DE-AC06-76RLO 1830*

Printed in the United States of America

Available to DOE and DOE contractors from the  
Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831;  
prices available from (615) 576-8401.

Available to the public from the National Technical Information Service,  
U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161



This document was printed on recycled paper.

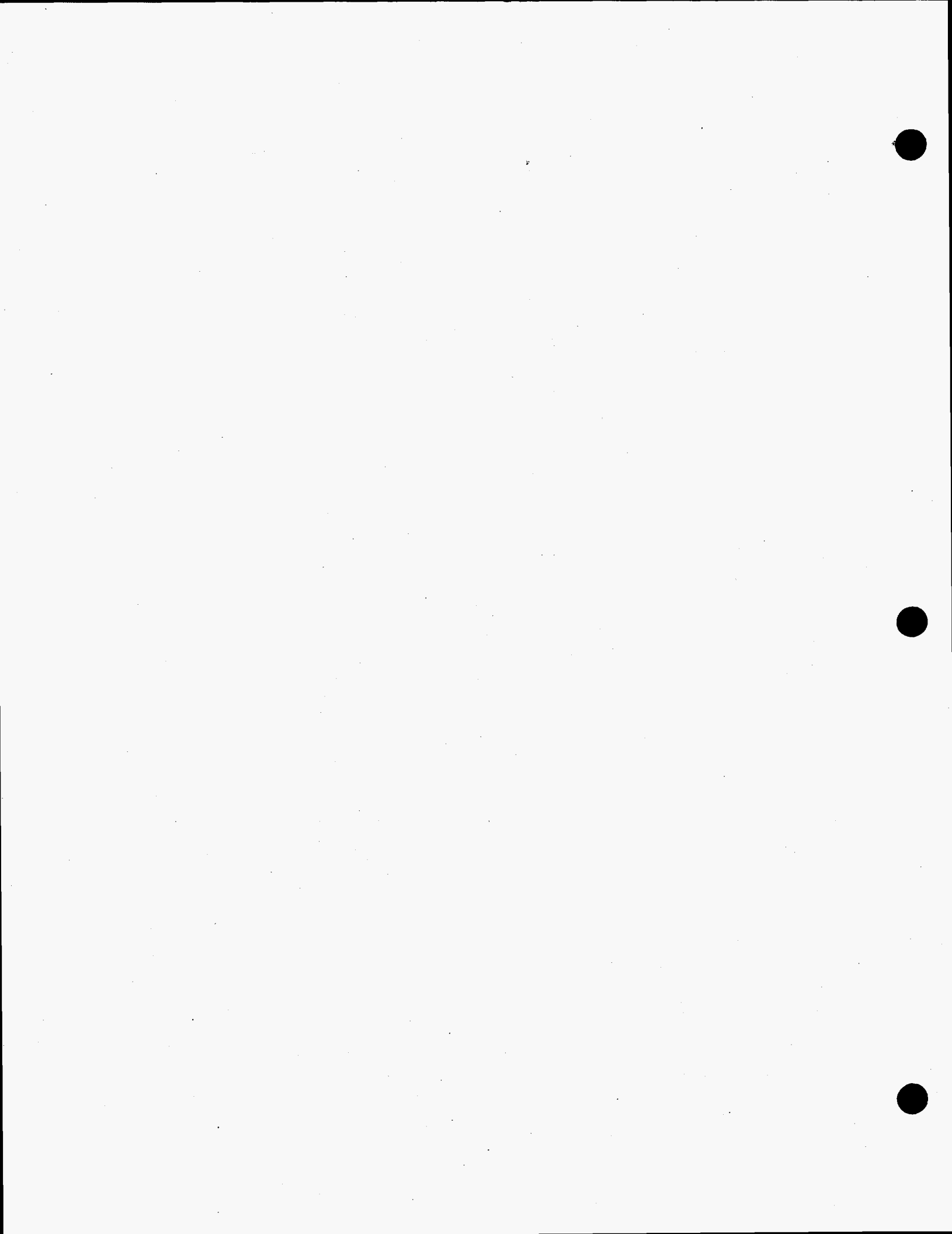
## **DISCLAIMER**

**Portions of this document may be illegible  
electronic image products. Images are  
produced from the best available original  
document.**

## Abstract

COBRA-SFS is a general thermal-hydraulic analysis computer code for prediction of material temperatures and fluid conditions in a wide variety of systems. The code has been validated for analysis of spent fuel storage systems, as part of the Commercial Spent Fuel Management Program of the U.S. Department of Energy. The code solves finite volume equations representing the conservation equations for mass, moment, and energy for an incompressible single-phase heat transfer fluid. The fluid solution is coupled to a finite volume solution of the conduction equation in the solid structure of the system.

This document presents a complete description of Cycle 2 of COBRA-SFS, and consists of three main parts. Part I describes the conservation equations, constitutive models, and solution methods used in the code. Part II presents the User Manual, with guidance on code applications, and complete input instructions. This part also includes a detailed description of the auxiliary code RADGEN, used to generate grey body view factors required as input for radiative heat transfer modeling in the code. Part III describes the code structure, platform dependent coding, and program hierarchy. Installation instructions are also given for the various platform versions of the code that are available.



## Acknowledgments

The authors would like to thank Leroy Stewart and Jeff Williams of the U.S. Department of Energy for sponsoring this work. Thanks are also extended to Mike McKinnon, Jim Creer, and Max Kreiter of the Pacific Northwest Laboratory Commercial Spent Fuel Management Program Office, who have provided support during this endeavor. In addition, the authors would like to acknowledge the Nuclear Regulatory Commission (NRC) Technical Review team for their thorough review of the COBRA-SFS computer code and documentation set. Finally, thanks go to Frank Ryan, Technical Editor, for his efforts in the preparation of this manuscript, and Zen Antoniak for his independent technical review.





## Nomenclature

- A channel cross-sectional area (ft<sup>2</sup>) in the axial direction
- $\underline{A}$  surface area for generalized control volume (ft<sup>2</sup>)
- $\bar{A}$  average cross-sectional area (ft<sup>2</sup>) over the subchannel control volume
- $\underline{A}$  surface area of generalized control volume (ft<sup>2</sup>)
- $A_{\text{fuel}}$  axial cross-sectional area of fuel (ft<sup>2</sup>)
- $A_c$  axial cross-sectional area of cladding (ft<sup>2</sup>)
- $A_I$  area of the interface between two solid nodes for heat transfer (ft<sup>2</sup>)
- $A_s$  surface area of slab node (ft<sup>2</sup>)
- $B_{ki}$  black body view factor from surface k to surface i for radiative heat transfer (dimensionless)
- $C_Q$  fraction of power generated in the coolant
- $c_p$  specific heat at constant pressure (Btu/lbm-°F)
- $C_T$  turbulent momentum factor (dimensionless empirical parameter)
- $\bar{D}$  average hydraulic diameter of lateral control volume (ft)
- $D_h$  hydraulic diameter of subchannel, based on wetted perimeter (ft)
- $D_e$  hydraulic diameter of subchannel, based on heated perimeter (ft)
- $D_{\text{fuel}}$  diameter of fuel (ft)
- $D_{\text{rod}}$  outside diameter of fuel rod (ft)
- $d_k$  diameter of kth conduction heat transfer node in fuel or cladding (ft)
- $E_j$  local mass continuity error at level j (lbm/sec)
- e energy per unit mass (Btu/lbm)

$F$	body force vector per unit mass (lbf) <sup>(a)</sup>
$\underline{F}$	fluid surface area for generalized control volume (ft <sup>2</sup> )
$F_m$	body force due to turbulent mixing (lbf-ft/sec <sup>2</sup> )
$F_{nm}$	grey body view factor from surface n to surface m for radiative heat transfer (dimensionless)
$f$	axial friction factor, Darcy formulation (dimensionless)
$f'$	friction for lateral flow, Darcy formulation (dimensionless)
$g$	acceleration of gravity vector, (ft/sec <sup>2</sup> )
$g$	scalar gravitational acceleration constant, (ft/sec <sup>2</sup> )
$g_c$	force/mass conversion constant for English engineering units, (32.2 lbf-ft/lbfm-sec <sup>2</sup> )
$\bar{G}$	average axial mass velocity in lateral control volume, (lbfm/sec-ft <sup>2</sup> )
$Gr$	Grashof Number (dimensionless)
$H$	heat transfer coefficient, (Btu/sec-ft <sup>2</sup> -°F)
$H_{gap}$	conductance across the fuel-cladding gap (Btu/sec-ft <sup>2</sup> -°F)
$H_{surf}$	surface heat transfer coefficient (Btu/sec-ft <sup>2</sup> -°F)
$h_f$	saturation enthalpy for the liquid phase, (Btu/lbfm)
$h_{fg}$	latent heat of vaporization, ( $h_g - h_f$ ), (Btu/lbfm)
$h_g$	saturation enthalpy for the vapor phase (Btu/lbfm)
$h$	enthalpy (Btu/lbfm)
$\hat{I}$	Identity tensor
$i$	internal thermal energy per unit mass (Btu/lbfm)
$K$	form drag for axial flow (dimensionless)
$K_G$	form drag for lateral flow (dimensionless)

---

(a) To properly balance units in all terms in the integral formulation of the energy conservation equation, the force vector  $F$  must be divided by the proportionality factor  $J$ , relating the ft-lbf force to thermal energy in units of Btu, as  $J = 778 \text{ ft-lbf/Btu}$ .

$k$	thermal conductivity (Btu/sec-ft-°F)
$l$	centroid length for gap $k$ (ft)
$m$	axial mass flow rate (lbm/sec)
$n$	unit vector, outward normal to $\underline{A}$
$n_A$	unit vector, normal to subchannel axial flow area $A$
$n_s$	unit vector, normal to gap lateral flow area, $S\Delta X$
$Nu$	Nusselt number (dimensionless)
$P$	pressure (psia)
$P_w$	wetted perimeter of subchannel control volume (ft)
$P_h$	heated perimeter of subchannel control volume (ft)
$Pr$	Prandtl number (dimensionless)
$q$	heat flux vector (Btu/sec-ft <sup>2</sup> )
$q'$	linear heat rate (Btu/sec-ft)
$q''$	surface heat flux (Btu/sec-ft <sup>2</sup> )
$q'''$	volumetric heat generation rate (Btu/sec-ft <sup>3</sup> )
$q_l$	energy deposited per unit volume (Btu/sec-ft <sup>3</sup> )
$Q_l$	energy deposited per unit length (Btu/sec-ft)
$Q_t$	turbulent energy exchange source term (Btu/sec)
$Q_w$	energy deposited per unit length from the heated surfaces, (Btu/sec-ft)
$Q_m$	total energy input due to turbulent mixing (Btu/sec)
$R$	thermal conduction resistance (Btu/sec-ft <sup>2</sup> -°F)
$Re$	Reynolds number (dimensionless)
$\dot{r}$	internal heat generation rate per unit mass (Btu/lbm-sec)
$S$	width (ft) of gap $k$ connecting channel $ii$ to channel $jj$

$t$  time (sec)  
 $T$  temperature ( $^{\circ}\text{F}$ )  
 $T_{\text{clad}}$  cladding surface temperature ( $^{\circ}\text{F}$ )  
 $T_{\text{fs}}$  temperature of the fuel surface ( $^{\circ}\text{F}$ )  
 $T_i$  fluid temperature in current node of subchannel  $i$  ( $^{\circ}\text{F}$ )  
 $T_{\text{fluid}}$  average fluid temperature seen by rod  $n$  ( $^{\circ}\text{F}$ )  
 $T_w$  wall surface temperature ( $^{\circ}\text{F}$ )  
 $T_{\text{wall}}$  wall surface temperature ( $^{\circ}\text{F}$ )  
 $T_b$  bulk fluid temperature in the subchannel control volume ( $^{\circ}\text{F}$ )  
 $\hat{T}$  tensor of surface stresses acting on the fluid  
 $t_{\text{clad}}$  cladding thickness (ft)  
 $U$  scalar velocity (ft/sec) in the axial direction  
 $U$  local axial velocity vector (ft/sec)  
 $\bar{U}_j$  average axial velocity at  $j$  for momentum cell (ft/sec)  
 $U$  composite thermal conductance (Btu/sec- $^{\circ}\text{F}$ )  
 $V$  local lateral velocity vector, (ft/sec)  
 $\underline{V}$  volume of generalized control volume (ft<sup>3</sup>)  
 $V$  scalar velocity in the lateral direction (ft/sec)  
 $\bar{V}_k$  average lateral velocity in gap  $k$  for momentum cell (ft/sec)  
 $\underline{W}$  wall surface area for generalized control volume (ft<sup>2</sup>)  
 $w$  mass flow rate in the lateral direction, per unit length (lbm/sec-ft)  
 $w'$  turbulent crossflow for momentum and enthalpy exchange between adjacent channels (lbm/ft-sec)  
 $X$  axial distance (ft)  
 $Z_k$  empirical shape factor for conduction length of gap  $k$  (dimensionless)

**Greek symbols:**

- $m$  empirical mixing coefficient for turbulent crossflow model (dimensionless)
- $\Delta X$  axial node length (ft)
- $\Delta h$  change in enthalpy (Btu/lbm)
- $\Delta r$  radial increment in fuel noding (ft)
- $\Delta U$  axial velocity difference between adjacent channels for turbulent mixing momentum exchange (ft/sec)
- $\Delta t$  transient time increment (sec)
- $\theta$  angle of channel axial orientation relative to vertical (degrees)
- $\epsilon$  surface emissivity of a solid node (dimensionless)
- $\epsilon_{ik}$  sign determiner for lateral velocity in gap k with respect to channel i.

$$\epsilon_{ik} = \begin{array}{l} 1.0 \text{ if } i = ii \\ -1.0 \text{ if } i = jj \end{array}$$

where  $ii < jj$

- $\epsilon_t$  eddy diffusivity for turbulence (ft<sup>2</sup>/sec)
- $\delta_{ki}$  Kronecker-delta function
- $\hat{\pi}$  shear stress tensor
- $\mu$  viscosity, (lbm/sec-ft<sup>2</sup>)
- $\rho$  density (lbm/ft<sup>3</sup>)
- $\sigma$  Stefan-Boltzmann constant;  $4.76(10^{-13})$  Btu/sec-ft<sup>2</sup>-R<sup>4</sup>
- $\phi_n$  fraction of perimeter of rod n facing a given subchannel (dimensionless)
- $\Sigma_{kei}$  summation on all gaps k connected to channel i

- $\Sigma$   
 $m \in n$  summation on all channels  $m$  facing rod  $n$
- $\Sigma$   
 $n \in i$  summation on all rods  $n$  with heat transfer surfaces facing channel  $i$
- $\Sigma$   
 $n \in k$  summation on all gaps  $n$  connected to gap  $k$  for lateral momentum transport
- $\Sigma$   
 $n \in (N+1)$  summation on all rod surface nodes  $n$  seen by cladding node  $N+1$
- $\Sigma$   
 $m \in (N+1)$  summation on all slab nodes  $m$  seen by cladding node  $N+1$

**Subscripts:**

- $c$  cladding material property
- $f_i$  fuel material property
- $ii$  lower-numbered channel of a pair connected by gap  $k$
- $j$  axial level index number
- $J$  axial level index for lateral momentum transport, (i.e.,  $J = j + 1/2$ )
- $jj$  higher-numbered channel of a pair connected by gap  $k$
- $k$  gap index number
- $\ell$  laminar flow
- $L$  laminar flow
- $s$  slab node property
- $t$  turbulent flow
- $w$  wall
- $\theta$  circumferential angle for radial fuel noding (radians)

**Superscripts:**

n previous time-step value

o current iteration value

N current time step value

" per unit cell quantity (ft<sup>3</sup>)





# Contents

Abstract .....	iii
Acknowledgments .....	v
Nomenclature .....	vii
1.0 Introduction .....	1.1
2.0 Conservation Equations .....	2.1
2.1 Integral Balance Laws .....	2.1
2.2 Subchannel Partial Differential Equations .....	2.4
2.3 Finite Difference Equations .....	2.13
2.3.1 Conservation of Mass .....	2.15
2.3.2 Conservation of Axial Momentum .....	2.16
2.3.3 Conservation of Lateral Momentum .....	2.18
2.3.4 Conservation of Energy for the Fluid .....	2.20
2.4 Energy Conservation in Fuel Rods and Solid Structures .....	2.22
2.4.1 Rod Energy Equation .....	2.22
2.4.2 Solid Structure Energy Equation .....	2.27
3.0 Constitutive Models .....	3.1
3.1 Fluid Flow Models .....	3.1
3.1.1 Wall Friction .....	3.1
3.1.2 Form Drag Models .....	3.3
3.1.3 Turbulent Mixing Models .....	3.4
3.2 Energy Exchange Models .....	3.5
3.2.1 Convective Heat Transfer Correlations .....	3.5
3.2.2 Fluid Conduction Shape Factor .....	3.6
3.2.3 Solid-to-Solid Conduction .....	3.6
3.2.4 Radiation Exchange Factors .....	3.8
4.0 Numerical Solution Methods .....	4.1
4.1 Fluid Flow Solution .....	4.1
4.1.1 Tentative Flow Solution .....	4.3
4.1.2 Pressure and Linearized Flow Solution .....	4.6

4.2	Energy Solution . . . . .	4.12
4.2.1	Fluid and Slab Energy Solution . . . . .	4.12
4.2.2	Fuel Rod Energy Solution . . . . .	4.14
5.0	Boundary Conditions . . . . .	5.1
5.1	Fluid Flow Boundary Conditions . . . . .	5.1
5.1.1	Inlet Mass Flow Rate Boundary Condition Options . . . . .	5.2
5.1.2	Inlet Pressure Boundary Condition Options . . . . .	5.3
5.2	Thermal Boundary Conditions . . . . .	5.5
5.2.1	Side Thermal Boundary Conditions . . . . .	5.6
5.2.2	Plenum Boundary Conditions . . . . .	5.7
5.3	Transient Forcing Functions . . . . .	5.10
6.0	User Manual . . . . .	6.1
6.1	User Guidance on Code Applications . . . . .	6.1
6.1.1	COBRA-SFS Cask Model Optimization . . . . .	6.1
6.1.2	Summary of Applications of COBRA-SFS . . . . .	6.8
6.1.3	What To Do When the Code Fails . . . . .	6.10
6.2	Input Instructions for COBRA-SFS . . . . .	6.16
6.2.1	Problem Initiation Input . . . . .	6.17
6.2.2	Group PROP--Fluid and Material Properties . . . . .	6.19
6.2.3	Group CHAN--Flow Field Geometry . . . . .	6.21
6.2.4	Group VARY--Geometry Variations . . . . .	6.29
6.2.5	Group RODS--Fuel Rod Geometry . . . . .	6.31
6.2.6	Group SLAB--Solid Structure Geometry . . . . .	6.39
6.2.7	Group RADG--Radiative Heat Transfer Exchange Factors . . . . .	6.48
6.2.8	Group HEAT--Heat Transfer Correlations . . . . .	6.54
6.2.9	Group DRAG--Friction Factors and Loss Coefficients 6.59	
6.2.10	Group BDRY--Thermal Boundary Conditions . . . . .	6.69
6.2.11	Group OPER--Operating Conditions . . . . .	6.80
6.2.12	Group CALC--Calculational Parameters . . . . .	6.91
6.2.13	Group OUTP--Output Options . . . . .	6.95
6.2.14	Termination of Input File . . . . .	6.100
7.0	Validation for Cycle 2 . . . . .	7.1
7.1	Benchmark Test: Cycle 1 to Cycle 2 . . . . .	7.1
7.2	Single-Assembly Test Cases . . . . .	7.2
7.2.1	SAHTT 15 x 15 PWR Test Assembly . . . . .	7.3
7.2.2	Mitsubishi 15 x 15 PWR Test Assembly . . . . .	7.8
7.2.3	BNFL 16 x 16 PWR Test Assembly . . . . .	7.11

7.3	Multiple-Assembly Shipping/Storage Cask Test Cases .....	7.13
7.3.1	TN24P with Unconsolidated Fuel .....	7.14
7.3.2	PSN Concrete Cask .....	7.26
7.4	Validation on Alternative Platforms .....	7.38
8.0	Code Structure and Installation .....	8.1
8.1	Functional Hierarchy of COBRA-SFS .....	8.1
8.1.1	Input Routines .....	8.1
8.1.2	Solution Routines .....	8.1
8.1.3	Output Routines .....	8.2
8.2	Subroutine Descriptions .....	8.3
8.3	Dimension Parameters .....	8.14
8.4	Platform-Dependent Coding .....	8.19
8.5	Installation Guide .....	8.32
8.5.1	Code Transmittal Package .....	8.32
8.5.2	Installation Instructions .....	8.34
8.5.3	Redimensioning COBRA-SFS and RADGEN .....	8.37
9.0	References .....	9.1
	Appendix A - Common Block Variables .....	A.1

## Figures

2.1	Relation of Subchannel Control Volume to Storage System . . . . .	2.4
2.2	Subchannel Control Volume . . . . .	2.5
2.3	Lateral Momentum Control Volume . . . . .	2.9
2.4	Subchannel Computational Cell . . . . .	2.14
2.5	COBRA-SFS Finite-Volume Fuel Rod Model with Central Void . . . . .	2.24
2.6	COBRA-SFS Finite-Volume Fuel Rod Model Without Central Void . . . . .	2.25
2.7	Solid Control Volume . . . . .	2.28
3.1	Solid-to-Solid Resistance Network . . . . .	3.7
4.1	Flow Chart of the RECIRC Solution Scheme . . . . .	4.2
5.1	Schematic Description of the Network Model for Pressure Drop Through the Reactor Vessel . . . . .	5.5
5.2	Simple Illustration of a COBRA-SFS Cask Model . . . . .	5.9
5.3	Example of COBRA-SFS Plenum Model . . . . .	5.10
5.4	Example of Channels Used to Model a Plenum in COBRA-SFS . . . . .	5.11
6.1	TN24P Cask Cross Section . . . . .	6.3
6.2	TN24P One-Half Section of Symmetry Cask Model . . . . .	6.4
6.3	TN24P One-Eighth Section of Symmetry Cask Model . . . . .	6.5
6.4	Rod-and-Subchannel Model for Fuel Assembly . . . . .	6.6
6.5	Lumped Rod and Channel Model for Fuel Assembly . . . . .	6.7
7.1	Diagram of SAHTT Assembly . . . . .	7.4
7.2	COBRA-SFS subchannel model of SAHTT Assembly . . . . .	7.5
7.3(a)	Comparison of COBRA-SFS Results with SAHTT Data: Rod Temperature Profiles on the Diagonal . . . . .	7.7

7-3(b)	Comparison of COBRA-SFS Results with SAHTT Data: Rod Temperature Profiles; Top Face to Bottom Face .....	7.7
7.4	COBRA-SFS Subchannel Model of Mitsubishi Test Assembly .....	7.9
7.5(a)	Mitsubishi Results for SICOH-3D Code .....	7.10
7.5(b)	Mitsubishi Results for COBRA-SFS, Cycle 2 .....	7.10
7.6	Cross-Section of BNFL Test Assembly .....	7.12
7.7(a)	Comparison of COBRA-SFS, Cycle 2 Results with BNFL Data: Rod Temperature Profile on the Diagonal .....	7.13
7.7(b)	Comparison of COBRA-SFS Results with BNFL Data: Rod Temperature Profiles; Top Face to Bo .....	7.13
7.8	TN24P PWR Spent Fuel Storage Cask .....	7.15
7.9	TN24P PWR Spent Fuel Storage Cask Cross-Section .....	7.16
7.10	Axial Noding for COBRA-SFS Model of TN24P Cask .....	7.18
7.11	COBRA-SFS Model of One-Eighth Section of Symmetry of TN24P Cask .....	7.19
7.12	Subchannel Model of One-Half Section of Fuel Assembly .....	7.20
7.13	Lumped Channel and Rod Model of Fuel Assembly .....	7.21
7.14	COBRA-SFS Model of One-Half Section of Symmetry of TN24P Cask .....	7.22
7.15(a)	Comparison of COBRA-SFS Calculations with Measured Temperatures in the Center Assembly of the TN24P Cask (vertical orientation, helium back-fill) .....	7.24
7.15(b)	Comparison of COBRA-SFS Calculations with Measured Temperatures in the Outer Assembly of the TN24P Cask (vertical orientation, helium back-fill) .....	7.24
7.16	COBRA-SFS Calculation of Top-to-Bottom Peak Temperature Profile Through the Centerline of the Cross-Section of the Horizontal TN24P Cask, Compared with Measured Temperatures .....	7.25
7.17	Diagram of PSN/VSC-17 Cask Structure .....	7.27
7.18	PSN/VSC-17 Cask Multi-Assembly Sealed Basket .....	7.28
7.19	Cross-Section of Consolidated Fuel Assembly .....	7.29

7.20	Axial Noding for COBRA-SFS Model of PSN/VSC-17 Cask . . . . .	7.31
7.21	COBRA-SFS Model of PSN/VSC-17 Cask: Half-Section of Symmetry . . . . .	7.32
7.22	COBRA-SFS Model of Structures Within the MSB in the PSN Cask . . . . .	7.33
7.23	COBRA-SFS Model of Consolidated Fuel Assemblies in PSN Cask . . . . .	7.34
7.24	Comparison of COBRA-SFS Calculations with Measured Temperatures in the PSN/VSC-17 Cask; vertical orientation, vents open, helium back-fill . . . . .	7.37
7.25	Comparison of COBRA-SFS Calculations with Measured Temperatures in the PSN/VSC-17 Cask; vertical orientation, vents open, nitrogen back-fill . . . . .	7.37

## Tables

4.1	Flow Derivatives with Respect to Pressure . . . . .	4.7
4.2	Continuity Error Derivatives with Respect to Pressure . . . . .	4.8
4.3	Definitions of Coefficients in the A Matrix . . . . .	4.16
4.4	Definitions of Terms in the Y Matrix. . . . .	4.17
6.1	Single-Assembly Validation of the COBRA-SFS Code . . . . .	6.9
6.2	Multi-Assembly Validation of COBRA-SFS Code . . . . .	6.10
6.3	Input and Output Files in RADGEN . . . . .	6.16
7.1	Comparison of COBRA-SFS Predictions to Measured Temperatures for SAHTT Test No. 12 . . . . .	7.8
7.2	TN24P Cask Test Matrix and Peak Temperatures . . . . .	7.17
7.3	Peak Temperature Predictions for TN24P Cask Tests . . . . .	7.23
7.4	PSN/VSC-17 Cask Test Matrix and Peak Temperatures . . . . .	7.30
7.5	Peak Temperature Predictions for PSN/VSC-17 Cask Tests . . . . .	7.36
7.6	Run Time on Alternative Platforms . . . . .	7.38
8.1	Dimension Parameters for COBRA-SFS Code . . . . .	8.15
8.2	Dimension Parameters for RADGEN Code . . . . .	8.18
8.3	COBRA-SFS Code Package . . . . .	8.33



# 1.0 Introduction

COBRA-SFS (Spent Fuel Storage)<sup>(a)</sup>, is a computer program that performs thermal-hydraulic analyses of multi-assembly spent-fuel storage and transportation systems, (Rector 1986a; Rector et al. 1986a; Lombardo et al. 1986a; Rector and Michener 1989). It uses a lumped-parameter, finite-difference approach to predict flow and temperature distributions in spent fuel storage systems and fuel assemblies, under forced and natural convection heat transfer conditions, in both steady-state and transients. Derived from the COBRA family of codes (Rowe 1973; Stewart et al. 1977; George et al. 1980; Khan et al. 1981), which have been extensively evaluated against in-pile and out-of-pile data, COBRA-SFS retains all the important features of the COBRA codes for single-phase analysis<sup>(b)</sup>, and extends the range of application to problems with two-dimensional radiative and three-dimensional conductive heat transfer. With these added capabilities, COBRA-SFS has been used to analyze various single- and multi-assembly spent fuel storage systems containing unconsolidated and consolidated fuel, with a variety of fill media (Cuta et al. 1984; Lombardo et al. 1986b; Cuta and Creer 1986; Wiles et al. 1986; Rector et al. 1986b; Rector et al. 1986a; McKinnon et al. 1986; Wheeler et al. 1986).

Cycle 0 of COBRA-SFS was released in 1986. Subsequent applications of the code required the development of additional capabilities, leading to the release of Cycle 1 in February 1989. Since then, the code has been subjected to an independent technical review as part of a submittal to the Nuclear Regulatory Commission (NRC) a generic license to apply the code to spent fuel storage system analysis. Minor modifications and error corrections were developed in response to the reviewers' recommendations. In addition, new capabilities and improvements to the code have been developed, and these changes have been combined to form a new release of the code, Cycle 2.

This report constitutes the documentation of Cycle 2 of COBRA-SFS. Much of its content has been reported elsewhere, in documentation of specific applications of the code and in validation and documentation of Cycle 1. However, to make life easier for the user, particularly the new user of the code, this report endeavors to be "the only COBRA-SFS manual you will ever need" for Cycle 2. It presents a complete description of the mathematical modeling and solution methods used in the code, documents the validation of Cycle 2, and guides the user through various applications of the code, and the input instructions. As a result, the report may at first glance seem intimidatingly large and complicated, containing rather more than one might really want to know about the code.

- 
- (a) The acronym COBRA stands for Coolant Boiling in Rod Arrays, and was originally coined to describe the code's application to thermal-hydraulic analysis of fuel rod bundles and reactor cores. COBRA-SFS uses essentially the same subchannel formulation, but with modifications and improvements for application to single-phase gas-cooled spent fuel storage casks with radiative, convective and conductive heat transfer.
  - (b) COBRA-SFS is not applicable to two-phase flow analysis as currently formulated. The models for subcooled boiling, the effect of phase change on the energy equation, and momentum effects of phase slip have been removed.

The organization of this manual, however, is intended to make it relatively easy to use. It is divided into three parts: a theory manual, a user guide, and a programmer's manual, organized as follows:

### **I. Theory Manual**

- Section 2.0 *Conservation Equations*
- Section 3.0 *Constitutive Models*
- Section 4.0 *Numerical Solution Methods*

### **II. Users Guide**

- Section 5.0 *Boundary Conditions*
- Section 6.0 *User Manual*

### **III. Programmer's Manual**

- Section 7.0 *Validation for Cycle 2*
- Section 8.0 *Code Structure and Installation*

In brief, Part I shows the user what sort of problems the code is suited for. Part II shows how to use the code to solve a problem. Part III presents some reassuring evidence that the code's results can be trusted to be reasonably accurate, and describes the code structure. This part also gives installation instructions for the code, and describes platform dependent programming required to allow the code to run on various workstations.

Part I describes the mathematical modeling and numerical solution methods used in the code. In Section 2.0, the formulation of the conservation equations for the working fluid and the equations for energy exchange with solid structures are described in detail. The constitutive models required to achieve closure of the equation set are presented in Section 3.0, for both the fluid equations and the energy equations. The numerical solution methods used to solve the fluid conservation equations and the energy equation for the solid structures are described in Section 4.0.

It is recommended that the user develop at least a passing familiarity with the equations solved in the COBRA-SFS code and the methods by which the solution is accomplished. This will aid the user in assessing the applicability of the code to new problems, and perhaps help develop insight into the most efficient way to model a given problem. However, an intimate acquaintance with the inner workings of the code is not required to actually use it in practical applications.

Applications information is covered in, Part II, consisting of Sections 5.0 and 6.0. Specification of boundary conditions for COBRA-SFS calculations are described in detail in Section 5.0. COBRA-SFS has a wide variety of options for specification of boundary conditions, including thermal-hydraulic boundary conditions on the working fluid and thermal boundary conditions on the solid structures. The user has a great deal of flexibility in specifying the boundary conditions for a problem, but that freedom comes at a price. The user must understand the workings of the boundary conditions options well enough to make rational choices which suit the particular problem at hand. The code includes some built-in checks on the boundary conditions, primarily to avoid inconsistent specifications, but there is no way to design an automatic check to verify that the boundary conditions *specified* are what the user actually *intended*.

The input instructions for the code are given in Section 6.0. As guidance for the new user, this section also includes a summary of previous applications of the code, with a list of previously published reports that document those applications. It provides examples of what to expect from the code, and demonstrates some appropriate ways to model various shipping cask and storage configuration problems. An earnest attempt has been made to make the input instructions as clear as possible, but COBRA-SFS is a complicated code which requires of the new user a significant investment of time on the learning curve. It is strongly recommended that the user carefully study the examples of previous code applications before attempting to set up a new problem.

Section 6.0 also includes a subsection describing how to use the RADGEN code. This is an auxiliary code that generates grey-body view factors for the radiative heat transfer modeling in COBRA-SFS. The view-factor information is merely another element of the code input, and it could come from any source that generates the view factors in a manner consistent with the way they are used in the code. In actuality, however, it is generally not practical to obtain them for rod arrays by any other means than the RADGEN code.

The Programmer's Manual, Part III, is also of interest to the practical code user, although it is perhaps somewhat less compelling than the User Guide in Part II. Validation of the Cycle 2 release is documented in Section 7.0. There is one "benchmark" test, comparing the results obtained with Cycle 2 to those obtained with the previous release, Cycle 1. In addition, three single-assembly test cases are presented, with data comparisons from test sections consisting of electrically heated rod bundles. Three test cases consisting of multi-assembly shipping or storage casks are also presented. The results are compared to measured data, and to previously reported results obtained with interim versions of the code.

Section 8.0 presents a description of the code structure and organization, including a summary of the program flow, and a brief description of each subroutine. It also describes the dimension parameters in the code, and explains how they may be changed to tailor the size of the code to a particular application. The COBRA-SFS code is designed to run on a variety of platforms, and Section 8.0 also describes how to obtain the various versions from the released source, and how to install the code on the various platforms.

## 2.0 Conservation Equations

The governing equations for flow of a single-component mixture can be formulated on an arbitrary fixed Eulerian control volume (Slattery 1972). This approach is used to develop the conservation equations solved in the COBRA-SFS code. Integral balances for mass, energy, and linear momentum are formed on the arbitrary Eulerian control volume, then applied to subchannel modeling with appropriate definitions and simplifications, and converted to partial differential equations over the subchannel control volume. The resulting system of subchannel equations is then expressed in finite-difference form, to be solved numerically using the procedures described in Section 4.0. The following subsections describe the integral balance laws, and show the simplifications and assumptions used to convert them to the subchannel partial differential equations. The final subsections show the approximation of these subchannel equations with the finite difference equations that are solved in the code.

### 2.1 Integral Balance Laws

Starting with an arbitrary Eulerian control volume,  $\underline{V}$ , bounded by a fixed surface  $\underline{A}$ , (which may consist of both fluid and solid boundaries), the integral balance for an arbitrary mixture property,  $Q$ , can be expressed as

$$\frac{\partial}{\partial t} \int_{\underline{V}} Q \, d\underline{V} + \int_{\underline{A}} Q (\mathbf{U} \cdot \mathbf{n}) \, d\underline{A} = \int_{\underline{V}} S_{\underline{V}} \, d\underline{V} - \int_{\underline{A}} (S_{\underline{A}} \cdot \mathbf{n}) \, d\underline{A} \quad (2.1)$$

The sum of all sources and sinks,  $S$ , of  $Q$  within the volume  $\underline{V}$  and on the surface  $\underline{A}$  must equal the sum of the change in  $Q$  inside  $\underline{V}$  and the rate at which  $Q$  moves across the surface  $\underline{A}$ . This integral balance can also be expressed in terms of the total derivatives, using Green's theorem, as

$$\frac{d}{dt} \int_{\underline{V}} Q \, d\underline{V} = \int_{\underline{V}} S_{\underline{V}} \, d\underline{V} - \int_{\underline{A}} (S_{\underline{A}} \cdot \mathbf{n}) \, d\underline{A}$$

In the COBRA-SFS, the appropriate mixture property  $Q$  for each of the conservation equations is as follows:

- mass conservation equation -- fluid density
- energy conservation equation -- density times fluid internal energy
- momentum equation -- density times the velocity vector.

The general form of the conservation equations, therefore, can be expressed as

mass:

$$\frac{\partial}{\partial t} \int_V \rho \, dV + \int_A \rho (\mathbf{U} \cdot \mathbf{n}) \, dA = 0 \quad (2.2)$$

energy:

$$\begin{aligned} \frac{\partial}{\partial t} \int_V \rho e \, dV + \int_A \rho e (\mathbf{U} \cdot \mathbf{n}) \, dA &= \int_V [\rho (\mathbf{F} \cdot \mathbf{U}) + \rho \dot{r}] \, dV \\ &+ \int_A [(\hat{\mathbf{T}} \cdot \mathbf{U}) - \mathbf{q}] \cdot \mathbf{n} \, dA \end{aligned} \quad (2.3)$$

momentum:

$$\frac{\partial}{\partial t} \int_V \rho \mathbf{U} \, dV + \int_A \rho \mathbf{U} (\mathbf{U} \cdot \mathbf{n}) \, dA = \int_V \rho \mathbf{F} \, dV + \int_A (\hat{\mathbf{T}} \cdot \mathbf{n}) \, dA \quad (2.4)$$

A number of simplifying assumptions are applied to these integral balance laws, consistent with the intended application of the code. Chief among these are the following:

1. The flow is at sufficiently low velocity so that kinetic and potential energy are negligible compared to internal thermal energy.
2. Work done by body forces and shear stresses is small compared to surface heat transfer and convective energy transport in the energy equation.
3. Gravity is the only significant body force in the momentum equation.
4. The fluid is incompressible but thermally expandable, so that density and transport properties vary only with the local temperature.
5. The radioactive energy source in the fuel is sufficiently decayed so that direct energy deposition in the fluid is negligible, and there is no significant gamma heating.
6. There is normally no lateral cross-coupling for momentum. (Note: this assumption can be superseded by special modeling options in the code.)

Considering the geometries that COBRA-SFS will be used to model, the surface integral over  $\underline{A}$  can be separated into a fluid component,  $\underline{F}$ , with fluid transport across the face, and a wall

component,  $\underline{W}$ , where the unit normal velocity vector is zero. The surface stresses can be similarly treated as consisting of a hydrostatic pressure component and a shear tensor component, such that

$$\hat{T} = -P\hat{I} + \hat{\pi}$$

Applying assumption No. 1 above, the fluid energy,  $e$ , can be expressed as

$$e = i + \frac{1}{2}U^2 \approx i$$

Using the definition of enthalpy, the fluid energy can be written as

$$\rho h = \rho i + P \approx \rho e + P$$

In substituting this relation for  $e$  in the integral balance for energy, it is assumed that the time derivative of the pressure over the volume is small enough with respect to surface heat transfer terms to be neglected.

These assumptions and simplifications allow the final form of the integral balance equations to be written as

mass:

$$\frac{\partial}{\partial t} \int_{\underline{V}} \rho d\underline{V} + \int_{\underline{E}} \rho (\underline{U} \cdot \underline{n}) d\underline{E} = 0 \quad (2.5)$$

energy:

$$\frac{\partial}{\partial t} \int_{\underline{V}} \rho h d\underline{V} + \int_{\underline{E}} \rho h (\underline{U} \cdot \underline{n}) d\underline{E} = - \int_{\underline{W}} (\underline{q} \cdot \underline{n}) d\underline{W} \quad (2.6)$$

momentum:

$$\begin{aligned} \frac{\partial}{\partial t} \int_{\underline{V}} \rho \underline{U} d\underline{V} + \int_{\underline{E}} \rho \underline{U} (\underline{U} \cdot \underline{n}) d\underline{E} &= \int_{\underline{V}} \rho \underline{g} d\underline{V} - \int_{\underline{E}} \underline{P} \underline{n} d\underline{E} \\ &- \int_{\underline{W}} \underline{P} \underline{n} d\underline{W} + \int_{\underline{W}} (\hat{\pi} \cdot \underline{n}) d\underline{W} \end{aligned} \quad (2.7)$$

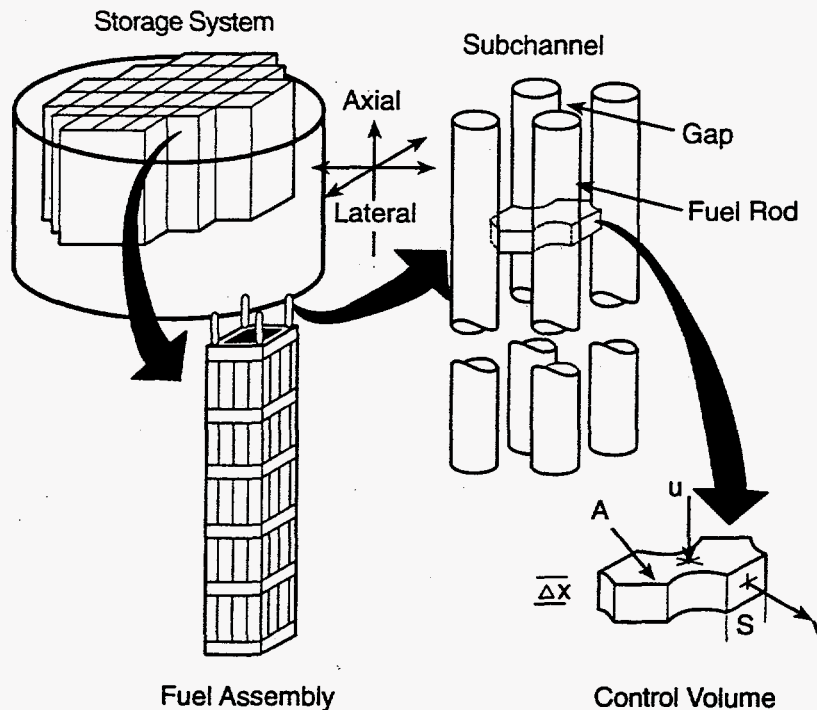
## 2.2 Subchannel Partial Differential Equations

In COBRA-SFS, the Eulerian control volume is the fluid subchannel in the rod array. Fluid flow is constrained by the surfaces of the closely spaced fuel rods, and on a small scale, the fuel rods partition the flow area into many subchannels that communicate laterally by crossflow through narrow gaps. This geometry is illustrated in Figure 2.1, which shows the subchannel in relation to the assembly. Applying the integral balance relations (Eqs. [2.5], [2.6], and [2.7]) to the subchannel control volume yields a set of subchannel equations which can be approximated in finite difference form.

The volume and surface area of the control volume comprising the subchannel are defined as shown in Figure 2.2, where  $A$  denotes the axial area for flow. Lateral flow occurs through the gaps between the fuel rods forming the subchannel. The subchannel illustrated in Figure 2.2 has three gaps, each of which with a lateral flow area given by  $S\Delta X$ . Assuming linear variation in  $A$  over axial distance (i.e., the distance along the main axis of the rod bundle), the volume,  $V$ , of the subchannel control volume is given by

$$V = \bar{A} \Delta X$$

where  $\bar{A} = \frac{1}{2}(A_x + A_{x+\Delta X})$



S9503052.1

Figure 2.1. Relation of Subchannel Control Volume to Storage System

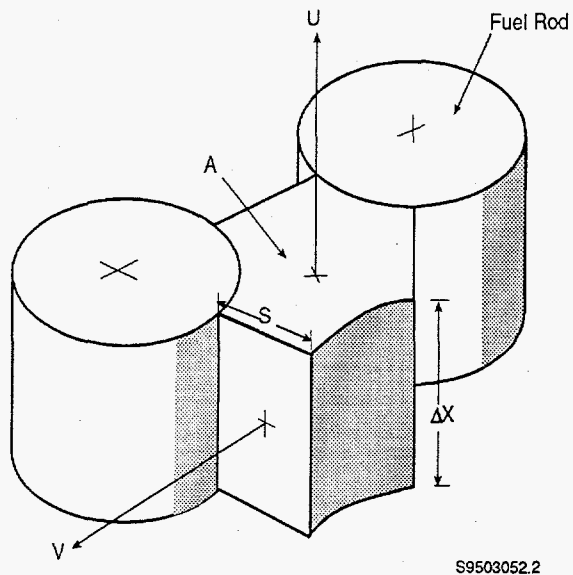


Figure 2.2. Subchannel Control Volume

The total fluid surface  $\underline{F}$  of the control volume consists of the axial flow area  $A$  at the top and bottom of the control volume, plus the lateral flow areas of the gaps between the fuel rods,  $S_k \Delta X$ . The solid surface  $\underline{W}$  of the control volume is defined by the surfaces of the rods forming the subchannels.

For each rod (of which there are three in Figure 2.2; note that the third one has been "cut away" so that the subchannel itself can be seen), the area of the solid surface is given by  $(\pi D_{rod}) \phi_n \Delta X$ .

It is assumed that the flow field is space- and time-averaged in such a way that the quantities of interest,  $(\rho, \rho U, \rho V, \rho h)$  have continuous derivatives. In addition, the volume and surface averages must be defined in terms of the volume and surface integrals. For example, the density and mass fluxes are defined as

$$\langle\langle \rho \rangle\rangle \equiv \frac{1}{V} \int_V \rho dV$$

$$\langle \rho U \rangle \equiv \frac{1}{A} \int_{E=A} \rho (\mathbf{U} \cdot \mathbf{n}) dE$$

$$\langle \rho V \rangle \equiv \frac{1}{S \Delta X} \int_{E=S \Delta X} \rho (\mathbf{V} \cdot \mathbf{n}) dE$$



Substituting these definitions into the integral balance equation for mass continuity (Eq. [2.5]), yields

$$\bar{A} \frac{\partial}{\partial t} \langle \rho \rangle + \langle \rho U \rangle_{X+\Delta X} - \langle \rho U \rangle_{X} + \Delta X \sum_{k \in I} \epsilon_{ik} \langle \rho \rangle V S = 0 \quad (2.8)$$

The summation for the lateral flow is over all gaps  $k$  of the subchannel  $I$ , and the term  $\epsilon_{ik}$  is a function to determine the sign of the flow. (See Section 2.3 and the Nomenclature for a complete explanation.)

This equation can be expressed in partial differential form by dividing by  $\Delta X$  and taking the limit as  $\Delta x$  becomes small. Therefore, the subchannel equation for mass continuity is

$$\bar{A} \frac{\partial}{\partial t} \langle \rho \rangle + \frac{\partial}{\partial X} \langle \rho U \rangle A + \sum_{k \in I} \epsilon_{ik} \langle \rho V \rangle S = 0 \quad (2.9)$$

A similar set of definitions can be made for the energy equation, so that the integrals relate to the averaged quantities as

$$\langle \rho h \rangle \equiv \frac{1}{V} \int_V \rho h \, dV$$

$$\langle \rho h U \rangle \equiv \frac{1}{A} \int_{F=A} \rho h (\mathbf{U} \cdot \mathbf{n}) \, dF$$

$$\langle \rho V h \rangle \equiv \frac{1}{S \Delta X} \int_{F=S \Delta X} \rho h (\mathbf{U} \cdot \mathbf{n}) \, dF$$

In addition, energy entering or leaving the control volume through the wall surfaces can be characterized by a summation over the rod surfaces facing the subchannel control volume. This yields the following convenient definition for the wall heat flux integral,

$$- \int_{\underline{W}} (\mathbf{q} \cdot \mathbf{n}) \, d\underline{W} \equiv \Delta X \sum_{n \in I} P_w \phi_n \langle q'' \rangle_X$$

The average heat flux,  $\langle q'' \rangle$ , is defined by an empirical heat transfer coefficient and the difference between the wall or rod surface temperature and the bulk fluid temperature. (See Section 3.0 for further discussion of constitutive models required for closure of the equation set.)

It should also be noted at this point that the surface integrals for energy transport, as defined above with the averaged quantities, do not take into account the effect of energy transport due to turbulent mixing. This is neglected in the definition, but because it is not always an insignificant portion of the energy exchanged between subchannels, it is included explicitly as an additional term that is determined using an empirical model. A time-fluctuating crossflow,  $w'$ , is defined as an equal mass exchange between adjacent control volumes, and related to the eddy diffusivity by

$$w' = \epsilon_t \rho \left[ \frac{S}{\ell} \right]$$

The total energy input due to turbulent mixing,  $Q_m$ , is treated as a source term in the energy balance,

$$Q_m = -\Delta X \sum_{kei} w' \Delta h$$

Substituting these relations and definitions into the integral balance equation for energy, (Eq. [2.6]), yields

$$\begin{aligned} \overline{A} \Delta X \frac{\partial}{\partial t} \langle\langle \rho h \rangle\rangle + A \langle \rho U h \rangle_{x+\Delta X} - \langle \rho U h \rangle_x + \Delta X \sum_{kei} e_{ik} \langle \rho V h \rangle S \\ = \Delta X \sum_{nei} P_w \phi_n \langle q'' \rangle - \Delta X \sum_{kei} w' \Delta h \end{aligned} \quad (2.10)$$

If the above equation is divided by  $\Delta X$  and the limit taken as  $\Delta X$  approaches zero, the result is a partial differential equation of the form

$$\begin{aligned} A \frac{\partial}{\partial t} \langle\langle \rho h \rangle\rangle + \frac{\partial}{\partial X} \langle \rho U h \rangle A + \sum_{kei} e_{ik} \langle \rho V h \rangle S \\ = \sum_{nei} P_w \phi_n \langle q'' \rangle - \sum_{kei} w' \Delta h \end{aligned} \quad (2.11)$$

The flowing enthalpy is defined as a function of the averaged energy and mass fluxes, such that

$$h \equiv \frac{\langle \rho U h \rangle}{\langle \rho U \rangle} \equiv \frac{\langle \rho V h \rangle}{\langle \rho V \rangle}$$

This allows consistent treatment of energy transport for both axial and lateral flow.

Consideration of axial and lateral momentum transport requires the development of separate equations for the axial and lateral components. The axial equation considers transport of momentum in the direction parallel to the longitudinal axis of the fuel rods. The lateral equation considers flow perpendicular to the fuel rods, through the gaps (see Figure 2.1). In most applications, the axial direction is vertical, with positive flow defined in the upward direction, while the lateral direction is

horizontal. However, COBRA-SFS has the capability to model horizontal or inclined assemblies and casks, and can correctly account for the buoyancy forces in both the axial and lateral directions.

Defining the momentum balance law integrals in terms of space- and time-averaged quantities over the subchannel control volume yields

$$-\int_{\underline{E}} P \mathbf{n} d\underline{E} \equiv -\langle P \rangle A_{X+\Delta X} + \langle P \rangle A_X$$

$$-\int_{\underline{W}} P \mathbf{n} d\underline{W} \equiv \langle \bar{P} \rangle (A_{X+\Delta X} - A_X)$$

The gravity force is decomposed into axial and lateral components as

$$\int_{\underline{V}} \rho \mathbf{g} d\underline{V} \equiv -[\bar{A} \Delta X \langle \langle \rho \rangle \rangle \cos \theta] \mathbf{n}_{\text{axial}} - [S \Delta X \ell \langle \langle \bar{\rho} \rangle \rangle \sin \theta] \mathbf{n}_{\text{lateral}}$$

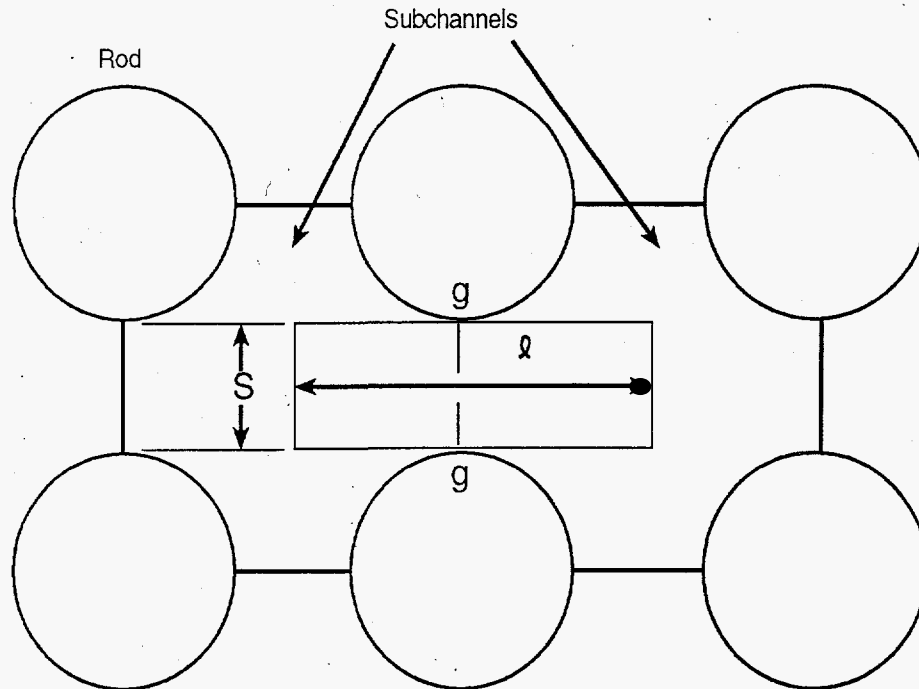
The volume and surface integrals for axial momentum consider the flux of axial momentum through all fluid surfaces of the control volume. These averaged quantities are defined in terms of the integrals as

$$\langle \langle \rho U \rangle \rangle \equiv \frac{1}{V} \int_{\underline{V}} \rho U d\underline{V} \cdot \mathbf{n}_{\text{axial}}$$

$$\langle \rho U^2 \rangle \equiv \frac{1}{A} \int_{\underline{E}=A} \rho U (U \cdot \mathbf{n}) d\underline{E} \cdot \mathbf{n}_{\text{axial}}$$

$$\langle \rho U V \rangle \equiv \frac{1}{S \Delta X} \int_{\underline{E}=S \Delta X} \rho U (U \cdot \mathbf{n}) d\underline{E} \cdot \mathbf{n}_{\text{axial}}$$

A modified control volume is used to define the corresponding quantities for the lateral component of the momentum equation. It consists of the volume defined by the gap width and centroid length, as illustrated in Figure 2.3.



S9503052.3

Figure 2.3. Lateral Momentum Control Volume

Using this control volume, the integrals for lateral momentum fluxes can be defined as

$$\langle\langle \rho V \rangle\rangle \equiv \frac{1}{S \Delta X \ell} \int_{\underline{V}=S\Delta X \ell} \rho \underline{U} d\underline{V} \cdot \underline{n}_{\text{lateral}}$$

$$\langle \rho V U \rangle \equiv \frac{1}{S \ell} \int_{\underline{E}=S} \rho \underline{U} (\underline{U} \cdot \underline{n}) d\underline{E} \cdot \underline{n}_{\text{lateral}}$$

The remaining integral terms in the balance equations must be defined using empirical relations. Shear stresses consist of wall shear and an optional fluid-fluid shear term for detailed modeling of large open flow regions with several channels.

The solid surface stress integral is approximated with empirical wall friction correlations and form loss coefficients. Axial wall and form drag is approximated as

$$\int_{\underline{W}} (\hat{\pi} \cdot \underline{n}) d\underline{W} = -\frac{1}{2} \left( \frac{f \Delta X}{D_h} + K \right) \langle \rho U^2 \rangle A$$

This formula is generally applicable to rod bundle geometries, and has been well validated for such cases. However, when channels are used to represent a large open flow region, the shear

stresses are primarily fluid-to-fluid. The wall shear stress is seen only by the outermost channel, and is more properly expressed as a function of channel velocity than as a friction factor. In such channels, the wall shear stress integral can be approximated as

$$\int_{\underline{W}} (\hat{\tau} \cdot \mathbf{n}) d\underline{W} = -\mu \left( \frac{P_w}{\ell} \right) \frac{\langle \rho U \rangle}{\langle \langle \rho \rangle \rangle} \Delta X$$

The shear stress term on the "sides" of the channels that do not see the wall is fluid-fluid shear, and the fluid surface stress integral is applied. Axial momentum is transferred between adjacent channels through the lateral gap connections, and the integral is approximated as

$$\int_{\underline{F}} (\hat{\tau} \cdot \mathbf{n}) d\underline{F} = - \sum_{k \neq i} e_{ik} \left[ \frac{\langle \rho U \rangle_{ii}}{\langle \langle \rho \rangle \rangle_{ii}} - \frac{\langle \rho U \rangle_{jj}}{\langle \langle \rho \rangle \rangle_{jj}} \right] \frac{S}{\ell} \mu_k \Delta X + F_{\text{lateral terms}}$$

When fluid-fluid shear is considered, it is no longer appropriate to neglect the fluid-to-fluid lateral transfer of momentum between adjacent lateral control volumes. Therefore,  $F_{\text{lateral terms}}$  in the above approximation of the fluid shear integral includes empirical terms for the lateral transport of lateral momentum and the axial transport of lateral momentum. The definitions of these terms depend on the subchannel formulation of the conservation equations presented in Section 2.3.3.

Wall shear stress for lateral momentum exchange is treated as a lateral form drag for lateral flow between adjacent channels, and is approximated as

$$\int_{\underline{W}} (\hat{\tau} \cdot \mathbf{n}) d\underline{W} = -\frac{1}{2} K_G \langle \rho V^2 \rangle S \Delta X$$

Momentum exchange due to turbulent mixing is also modeled empirically, in the same manner as in the energy equation. The total axial force,  $F_m$ , on the control volume as a result of turbulent mixing is defined as

$$F_m = -C_T \Delta X \sum_{k \neq i} w' \Delta U$$

Using these definitions, the axial component of the integral momentum balance (Eq. [2.7]) can be expressed as

$$\begin{aligned}
 \Delta X \frac{\partial}{\partial t} \langle \langle \rho U \rangle \rangle \bar{A} + \langle \rho U^2 \rangle A_{X+\Delta X} - \langle \rho U^2 \rangle A_X + \Delta X \sum_{k\epsilon i} e_{ik} \langle \rho UV \rangle S \\
 = -\bar{A} (\langle P \rangle_{X+\Delta X} - \langle P \rangle_X) - \bar{A} \Delta X \langle \langle \rho \rangle \rangle g \cos \theta \\
 - \frac{1}{2} \left( \frac{f \Delta X}{D_h} + K \right) \langle \rho U^2 \rangle A - \Delta X C_T \sum_{k\epsilon i} w' \Delta U
 \end{aligned} \tag{2.12}$$

The lateral component of Eq. (2.7) can be similarly expressed as

$$\begin{aligned}
 S \ell \Delta X \frac{\partial}{\partial t} \langle \langle \rho V \rangle \rangle + S \ell \langle \rho VU \rangle_{X+\Delta X} - S \ell \langle \rho VU \rangle_X \\
 = -\frac{1}{2} K_G \langle \rho V^2 \rangle S \Delta X + S \Delta X [\langle P \rangle_{ii} - \langle P \rangle_{jj}] - S \Delta X \langle \langle \bar{\rho} \rangle \rangle g \sin \theta
 \end{aligned} \tag{2.13}$$

Equations (2.12) and (2.13) apply to rod bundles or other geometries similar to rod arrays. In applications where large open flow regions are modeled, in which fluid-fluid shear must be included, these equations appear as shown below. The axial component of the momentum equation with fluid-fluid shear is

$$\begin{aligned}
 \Delta X \frac{\partial}{\partial t} \langle \langle \rho U \rangle \rangle \bar{A} + \langle \rho U^2 \rangle A_{X+\Delta X} - \langle \rho U^2 \rangle A_X \\
 + \Delta X \sum_{k\epsilon i} e_{ik} \langle \rho UV \rangle S = -\bar{A} (\langle P \rangle_{X+\Delta X} - \langle P \rangle_X) \\
 - \bar{A} \Delta X \langle \langle \rho \rangle \rangle g \cos \theta - \mu \left( \frac{P_w}{\ell} \right) \frac{\langle \rho U \rangle}{\langle \langle \rho \rangle \rangle} \Delta X \\
 - \sum_{k\epsilon i} \left[ \frac{\langle \rho U \rangle_{ii}}{\langle \langle \rho \rangle \rangle_{ii}} - \frac{\langle \rho U \rangle_{jj}}{\langle \langle \rho \rangle \rangle_{jj}} \right] \frac{S}{\ell} \mu_k \Delta X \\
 - \Delta X C_T \sum_{k\epsilon i} w' \Delta X
 \end{aligned} \tag{2.12a}$$

The lateral component of the momentum equation with fluid-fluid shear can be expressed as

$$\begin{aligned}
 S \ell \Delta X \frac{\partial}{\partial t} \langle \langle \rho V \rangle \rangle + S \ell \langle \rho V U \rangle_{x+\Delta X} - S \ell \langle \rho V U \rangle_x \\
 = -\frac{1}{2} K_G \langle \rho V^2 \rangle S \Delta X + S \Delta X [\langle P \rangle_{ii} - \langle P \rangle_{jj}] \\
 - S \Delta X \langle \langle \bar{\rho} \rangle \rangle g \sin \theta + F_{\text{lateral terms}}
 \end{aligned} \tag{2.13a}$$

Equations (2.12) and (2.13) can be treated in the same manner as the mass continuity and energy equations. The subchannel partial differential equations are obtained by dividing by  $\Delta X$  and taking the limit as  $\Delta X$  goes to zero. For the axial momentum equation, this yields

$$\begin{aligned}
 \frac{\partial}{\partial t} \langle \langle \rho U \rangle \rangle A + \frac{\partial}{\partial X} \langle \rho U^2 \rangle A + \sum_{k \in i} e_{ik} \langle \rho U V \rangle S = -A \frac{\partial}{\partial X} \langle P \rangle \\
 - \frac{1}{2} \left( \frac{f}{D_h} + K \right) \langle \rho U^2 \rangle A - A \langle \langle \rho \rangle \rangle g \cos \theta - C_T \sum_{k \in i} w' \Delta U
 \end{aligned} \tag{2.14}$$

The corresponding subchannel equation for the lateral component of momentum is

$$\begin{aligned}
 \frac{\partial}{\partial t} \langle \langle \rho V \rangle \rangle S + \frac{\partial}{\partial X} \langle \rho V U \rangle S = \frac{S}{\ell} (\langle P \rangle_{ii} - \langle P \rangle_{jj}) \\
 - \frac{1}{2} \frac{S}{\ell} K_G \langle \rho V^2 \rangle - S \Delta X \langle \langle \bar{\rho} \rangle \rangle g \sin \theta
 \end{aligned} \tag{2.15}$$

The same thing can be done for Eqs. (2.12a) and (2.13a), which yields subchannel partial differential equations of the following form. For the axial component;

$$\begin{aligned}
 \frac{\partial}{\partial t} \langle \langle \rho U \rangle \rangle A + \frac{\partial}{\partial X} \langle \rho U^2 \rangle A + \sum_{k \in i} e_{ik} \langle \rho U V \rangle S \\
 = -A \frac{\partial}{\partial X} \langle P \rangle - \mu \left( \frac{P_w}{\ell} \right) \frac{\langle \rho U \rangle}{\langle \langle \rho \rangle \rangle} \\
 - \sum_{k \in i} \left[ \frac{\langle \rho U \rangle_{ii}}{\langle \langle \rho \rangle \rangle_{ii}} - \frac{\langle \rho U \rangle_{jj}}{\langle \langle \rho \rangle \rangle_{jj}} \right] \frac{S}{\ell} \mu_k \\
 - A \langle \langle \rho \rangle \rangle g \cos \theta - C_T \sum_{k \in i} w' \Delta U
 \end{aligned} \tag{2.14a}$$

The corresponding lateral equation is

$$\frac{\partial}{\partial t} \langle \langle \rho V \rangle \rangle S + \frac{\partial}{\partial X} \langle \rho V U \rangle S = \frac{S}{\ell} (\langle P \rangle_{ii} - \langle P \rangle_{jj})$$

$$- \frac{1}{2} \frac{S}{\ell} K_G \langle \rho V^2 \rangle + F_{\text{lateral terms}} - S \Delta X \langle \bar{\rho} \rangle g \sin \theta$$
(2.15a)

The following subsection shows how these subchannel partial differential equations for mass continuity (Eq. [2.9]), for fluid energy conservation (Eq. [2.11]), and for momentum conservation (Eqs. [2.14] and [2.15], or Eqs. [2.14a] and [2.15a]) are approximated as finite difference equations for solution in the code. However, it should be noted that the equations derived here are applicable to more general problems than simply subchannel analysis. They can be used to model any flow field that can be adequately represented as a set of parallel channels with predominantly axial flow which communicate through lateral flow paths connecting the channels. The only requirement is that the assumptions and simplifications described above for subchannel modeling are not violated.

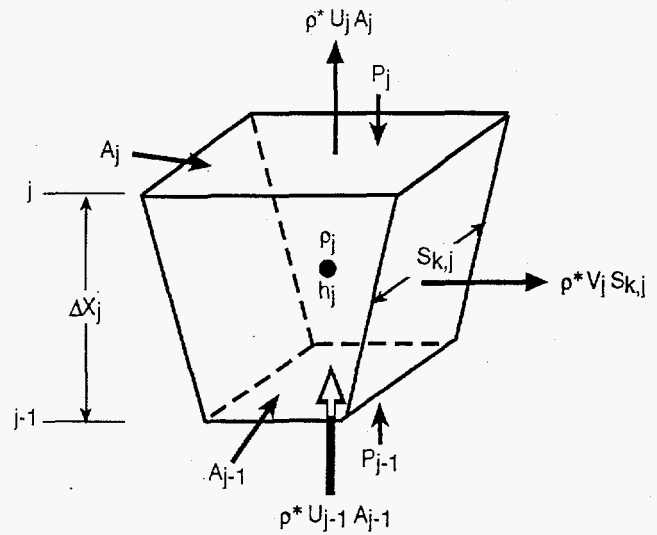
### 2.3 Finite Difference Equations

To develop the finite difference equations, the control volume is represented by the computational cell shown in Figure 2.4, with the computational variables located as shown. The state variables of density,  $\rho$ , and enthalpy,  $h$ , are defined at the cell center and are indexed by the node number. The axial flow rate ( $\rho^*UA$ ), the pressure,  $P$ , and axial flow area,  $A$ , are defined at the upper and lower cell boundaries, and are indexed by the corresponding axial levels,  $j$  and  $j-1$ . For a given gap  $k$ , the gap width,  $S_k$ , and the crossflow per unit length, ( $\rho^*VS_k$ ), are defined on the transverse cell boundary midway between the axial levels and are indexed by the gap number and axial level of the upper face,  $j$ , of the cell.

The subchannel equations are formulated using velocity as the transportive variable. However, within the code the solution is expressed in terms of the mass flow rates ( $m$  and  $w$  for the axial and lateral directions, respectively). This requires some additional definitions to specify the flow rates and momentum flux terms with appropriate donor cell quantities. The positive direction for the axial velocity,  $U$ , is the direction from the channel inlet to the exit, along the main axis of the fuel rods. For the conventional application to vertically oriented fuel bundles, this is the upward flow direction. Negative axial velocities define flow in the opposite direction.

The sign convention on the lateral velocity, however, does not have such a convenient reference as gravity, and so is less intuitively obvious. The lateral velocity in gap  $k$ ,  $V_k$ , is defined as positive when flow is from the lower-numbered subchannel of the pair forming the gap (denoted  $ii$ ), into the higher-numbered channel of the pair (denoted  $jj$ ). It is negative if flow is in the opposite





S9503052.4

Figure 2.4. Subchannel Computational Cell

direction. This convention is implemented by means of the unitary switch function,  $\epsilon_{ik}$ , which is applied as a multiplier on terms containing the lateral velocity,  $V$ . It is defined for gap  $k$  such that

$$\begin{aligned} \epsilon_{ik} &= 1.0, & \text{if } i = ii \\ \epsilon_{ik} &= -1.0, & \text{if } i = jj \end{aligned}$$

Employing the above definitions for positive and negative flow directions, the donor cell convention for convected quantities results in the following definitions of the momentum flux terms;

$$\begin{aligned} \text{for } U_j \geq 0.0, & \quad \rho^* U_j = \rho_j U_j \\ \text{for } U_j < 0.0, & \quad \rho^* U_j = \rho_{j+1} U_j \\ \text{for } V_k \geq 0.0, & \quad \rho^* V_k = \rho_{ii} V_k \\ \text{for } V_k < 0.0, & \quad \rho^* V_k = \rho_{jj} V_k \end{aligned}$$

The axial flow rate can then be defined generally as

$$m_j = \rho^* U_j A_j$$

Similarly, the general definition for the lateral flow rate (which is by convention, treated on a per unit length basis to accommodate variable axial noding) is

$$w_k = \rho^* V_k S_k$$

Using these definitions, the subchannel partial differential equations, (Eqs. [2.9], [2.11], [2.14], and [2.15]), can be expressed in terms of flow rates.

mass:

$$A \frac{\partial \rho}{\partial t} + \frac{\partial m}{\partial X} + \sum_{k \in i} e_{ik} w_k = 0 \quad (2.16)$$

energy:

$$\begin{aligned} A \frac{\partial(\rho h)}{\partial t} + \frac{\partial(mh)}{\partial X} + \sum_{k \in i} e_{ik} w_k h \\ = \sum_{n \in i} P_w \phi_n \langle q'' \rangle - \sum_{k \in i} w'_k \Delta h \end{aligned} \quad (2.17)$$

axial momentum:

$$\begin{aligned} \frac{\partial m}{\partial t} + \frac{\partial(mU)}{\partial X} + \sum_{k \in i} e_{ik} w_k U = A \frac{\partial P}{\partial X} \\ - \frac{1}{2} \left( \frac{f}{D_h} + K \right) |m| \frac{m}{\rho A} - A \rho g \cos \theta - C_T \sum_{k \in i} w'_k \Delta U \end{aligned} \quad (2.18)$$

lateral momentum:

$$\frac{\partial w}{\partial t} + \frac{\partial(wU)}{\partial X} = \frac{S}{\ell} (P_{ii} - P_{jj}) - \frac{1}{2\ell} K_G |w| \frac{w}{\rho S} - S \bar{\rho} g \sin \theta \quad (2.19)$$

The finite difference equations are derived from these subchannel equations by approximating the time derivative with the time step,  $\Delta t$ , and the spatial derivative with the noding increment,  $\Delta x$ , as shown in the following subsections.

### 2.3.1 Conservation of Mass

The assumptions made in the derivation of the continuity equation are that the channel area changes linearly with distance over the length of the control volume, the fluid density is uniform throughout the control volume, the axial and lateral velocities are uniform over the respective areas, and the lateral connection width is constant over the length of the control volume.

The final form of the equation for conservation of mass in the COBRA-SFS code is

$$\bar{A}_j \frac{\Delta X_j}{\Delta t} (\rho - \rho^n)_j + m_j - m_{j-1} + \Delta X_j \sum_{k \in i} e_{ik} w_k = 0 \quad (2.20)$$

### 2.3.2 Conservation of Axial Momentum

The time and space derivatives for the flow rates can be approximated for the momentum equations in the same way as for the mass continuity equation. The derivative of the pressure is slightly more complicated, however. It is approximated by assuming a linear pressure variation such that

$$-\bar{P} = \frac{P_j + P_{j-1}}{2}$$

The pressure difference can then be written as

$$P_{j-1}A_{j-1} - P_jA_j = \bar{P} (A_{j-1} - A_j) + \bar{A} (P_{j-1} - P_j)$$

The pressure force resulting from the area change is canceled, and

$$P_{j-1}A_{j-1} - P_jA_j - \bar{P}(A_{j-1} - A_j) = \bar{A} (P_{j-1} - P_j)$$

In addition to the definition of the axial and transverse flow rates ( $m$  and  $w$ ) and the convective terms defined above, appropriate averaging across node boundaries must be defined for the axial and lateral convection of axial momentum. For the axial convection of axial momentum, the transporting velocity is the velocity at the cell center, which is defined as the average of the velocity at  $j$  and  $j+1$ , such that

$$\bar{U}_j = \frac{\frac{1}{2}(m_j + m_{j+1})}{\frac{1}{2}(\rho_j + \rho_{j+1}) \frac{1}{2}(\bar{A}_j + \bar{A}_{j+1})}$$

This velocity convects either  $m_j$  or  $m_{j+1}$ , depending on its direction, so that

$$\bar{U}_j m^* = \begin{cases} \bar{U}_j m_j & \text{if } \bar{U}_j \geq 0 \\ \bar{U}_j m_{j+1} & \text{if } \bar{U}_j < 0 \end{cases}$$

For the lateral transport of axial momentum, the transporting velocity is the lateral velocity at the cell boundary, which is obtained by averaging the crossflows at  $j$  and  $j+1$ , such that

$$\bar{V}_{kj} = \frac{\frac{1}{2}(w_j + w_{j+1})}{\frac{1}{2}(\bar{\rho}_{kj} + \bar{\rho}_{k,j+1}) \frac{1}{2}(S_{kj} + S_{k,j+1})}$$

The average densities at  $j$  and  $j+1$  used in this relation are defined as

$$\bar{\rho}_k = \frac{1}{2}(\rho_{ii} + \rho_{jj})$$

The average lateral velocity convects the axial momentum at  $j$  of either channel  $ii$  or  $jj$ , depending on its sign, so that for any given axial level  $j$ , the transport term for gap  $k$  is

$$\bar{V}_k \left( \frac{m_j}{A_j} \right)^* = \begin{cases} \bar{V}_k \frac{m_{ii}}{A_{ii}} & \text{if } \bar{V}_k \geq 0 \\ \bar{V}_k \frac{m_{jj}}{A_{jj}} & \text{if } \bar{V}_k < 0 \end{cases}$$

The  $\Delta U$  term in the turbulent momentum exchange must also be defined, using appropriate averaging, to obtain the axial velocities at a given level. For channel  $I$  (which may be either  $ii$  and  $jj$ ) at level  $j$ , the average axial velocity is given by

$$\bar{U}_i = \frac{m_i}{\bar{\rho}_i A_i}$$

where  $\bar{\rho}_i = \frac{1}{2}(\rho_j + \rho_{j+1})$

Using this definition, the turbulent cross-flow exchange can be expressed as

$$\sum_{k \in i} e_{ik} w' \left( \frac{m_{ii}}{\bar{\rho}_{ii} A_{ii}} - \frac{m_{jj}}{\bar{\rho}_{jj} A_{jj}} \right) C_T \Delta X$$

The turbulent cross-flow produces no net mass exchange between adjacent channels. However, it does transport both momentum and energy from one channel to the other. The empirical proportionality constant relating turbulent momentum to turbulent energy transport is  $C_T$ . This coefficient has the same function as a turbulent Prandtl number. If it is equal to 1.0, energy and momentum are

exchanged at equal rates. If it is set to 0.0, there is no lateral momentum exchange due to turbulence. The actual value used in the code is determined by user input.

When the above definitions and approximations are used in the subchannel equation for axial momentum, Eq. (2.18), the finite difference form of the axial momentum equation is

$$\begin{aligned}
 & \frac{\Delta X_j (m_j - m_j^n)}{\Delta t} + m_j^* \bar{U}_j - m_{j-1}^* \bar{U}_{j-1} + \sum_{k \in i} \left[ e_{ik} \left( \frac{m}{A} \right)_k^* \bar{V}_k S_k \right]_j \Delta X_j \\
 & + \sum_{k \in i} e_{ik} w_k \left( \frac{m_{ii}}{\bar{\rho}_{ii} A_{ii}} - \frac{m_{jj}}{\bar{\rho}_{jj} A_{jj}} \right) C_T \Delta X_j = \bar{A}_j g_c (P_{j-1} - P_j) \\
 & - \Delta X_j \bar{A}_j \rho_j g \cos \theta - \frac{1}{2} \left( \frac{f}{D_h} + \frac{K}{\Delta X_j} \right) \Delta X_j \left| \frac{m_j}{\rho_j A_j} \right| m_j
 \end{aligned} \tag{2.21}$$

In applications where fluid-fluid shear must be included in the axial momentum equation, the finite difference form is derived from Eq. (2.14a) rather than Eq. (2.14). All of the terms are the same as shown in Eq. (2.21), except for the wall shear stress term. The finite difference form in this case is expressed as follows;

$$\begin{aligned}
 & \frac{\Delta X_j (m_j - m_j^n)}{\Delta t} + m_j^* \bar{U}_j - m_{j-1}^* \bar{U}_{j-1} + \sum_{k \in i} \left[ e_{ik} \left( \frac{m}{A} \right)_k^* \bar{V}_k S_k \right]_j \Delta X_j \\
 & + \sum_{k \in i} e_{ik} w_k \left( \frac{m_{ii}}{\bar{\rho}_{ii} A_{ii}} - \frac{m_{jj}}{\bar{\rho}_{jj} A_{jj}} \right) C_T \Delta X_j = \bar{A}_j g_c (P_{j-1} - P_j) \\
 & - \Delta X_j \bar{A}_j \rho_j g \cos \theta - \sum_{k \in i} e_{ik} (U_{ii} - U_{jj}) \frac{S}{\ell} \mu_k \Delta X_j - \mu \frac{P_w}{\ell} U_j \Delta X_j
 \end{aligned} \tag{2.21a}$$

### 2.3.3 Conservation of Lateral Momentum

As with the axial momentum equation, average velocities for momentum transport must be defined in order to formulate the finite difference equation. In the axial convection of transverse momentum, the transporting velocity is defined as

$$\bar{U}_j = \frac{1}{(A_{ii} + A_{jj})} \left( \frac{m_{ii}}{\bar{\rho}_{ii}} + \frac{m_{jj}}{\bar{\rho}_{jj}} \right)$$

The cross-flow convected by this average velocity defines the transport term as

$$\bar{U}_j w^* = \begin{cases} \bar{U}_j w_j & \text{if } \bar{U}_j \geq 0 \\ \bar{U}_j w_{j+1} & \text{if } \bar{U}_j < 0 \end{cases}$$

When the above definitions are substituted into Eq. (2.15), the final form of the finite difference equation for lateral momentum becomes

$$\begin{aligned} \Delta X_j \left( \frac{w_j - w_j^n}{\Delta t} \right) + w_j^* \bar{U}_j - w_{j-1}^* \bar{U}_{j-1} &= \frac{S_j \Delta X_j}{\ell} g_c (P_{ii} - P_{jj})_{j-1} \\ &- S_j \ell \Delta X_j \bar{\rho}_j g \sin \theta - \frac{1}{2} K_G \left| \frac{w_j}{\bar{\rho}_j S_j} \right| w_j \frac{\Delta X_j}{\ell} \end{aligned} \quad (2.22)$$

In applications where fluid-fluid shear must be included in the lateral momentum equation, the finite difference form is derived from Eq. (2.15a) rather than Eq. (2.15). All of the terms are the same as shown in Eq. (2.22) above, except for the additional fluid shear stress terms.

$$F_{\text{lateral terms}} = F_{\text{axl}} + F_{\text{lat}}$$

The axial transport of lateral momentum,  $F_{\text{axl}}$ , is defined as

$$F_{\text{axl}} = -S\ell \left[ 2\mu_{j-1} \frac{V_j - V_{j-1}}{\Delta X_j + \Delta X_{j+1}} + 2\mu_j \frac{V_j - V_{j+1}}{\Delta X_j + \Delta X_{j+1}} \right]$$

The viscosities are calculated by averaging between the values of the adjacent channels connected by the gap at the axial levels  $j$  and  $j-1$ , and at  $j$  and  $j+1$ , as follows:

$$\begin{aligned} \mu_j &= \frac{1}{4} [\mu_{iij} + \mu_{iij} + \mu_{iij+1} + \mu_{jjj+1}] \\ \mu_{j-1} &= \frac{1}{4} [\mu_{iij} + \mu_{jjj} + \mu_{iij-1} + \mu_{jjj-1}] \end{aligned}$$

The lateral transport of lateral momentum,  $F_{lat}$ , is defined as

$$F_{lat} = - \sum_{nek} \ell \Delta X_j \bar{\mu} \left[ \frac{2(V_k - V_n)}{S_k + S_n} \right]$$

In the summation, n is the index of a gap connected to gap k by fluid-fluid shear. The average viscosity is calculated at a given axial level as

$$\bar{\mu} = \frac{1}{4} [\mu_{ii,k} + \mu_{jj,k} + \mu_{ii,n} + \mu_{jj,n}]$$

The  $F_{lateral\ terms}$  defined above are substituted into the lateral momentum equation for the case with fluid-fluid shear. The finite difference form is in therefore expressed as follows:

$$\begin{aligned} \Delta X_j \left( \frac{w_j - w_j^n}{\Delta t} \right) + w_j^* \bar{U}_j - w_{j-1}^* \bar{U}_{j-1} &= \frac{S_j \Delta X_j}{\ell} g_c (P_{ii} - P_{jj})_{j-1} \\ &- S_j \ell \Delta X_j \bar{\rho}_j g \sin\theta - \frac{1}{2} K_G \left| \frac{w_j}{\bar{\rho}_j S_j} \right| w_j \frac{\Delta X_j}{\ell} \\ &- S \ell \left[ 2\mu_{j-1} \frac{V_j - V_{j-1}}{\Delta X_j + \Delta X_{j+1}} + 2\mu_j \frac{V_j - V_{j+1}}{\Delta X_j + \Delta X_{j+1}} \right] \\ &- \sum_{nek} \ell \Delta X_j \bar{\mu} \left[ \frac{2(V_k - V_n)}{S_k + S_n} \right] \end{aligned} \tag{2.22a}$$

The lateral fluid-fluid shear term is incorporated into the COBRA-SFS code in a manner that permits only two other cross-flows to be connected to a given gap, k, for the exchange of lateral momentum.

### 2.3.4 Conservation of Energy for the Fluid

The finite difference approximation of the subchannel equation for fluid energy, Eq. (2.17), can be written directly, using the definitions and assumptions noted above for the mass conservation equation. The only significant differences are that the axial and lateral flows (m and w) convect

enthalpy rather than density in this equation, and there are energy fluxes through the solid surfaces as well as the fluid surfaces of the control volume. The finite difference form of the energy equation is, therefore,

$$\begin{aligned}
 & \bar{A}_j \Delta X_j \frac{(\rho h - (\rho h)^n)_j}{\Delta t} + m_j h^* - m_{j-1} h^* \\
 & + \Delta X_j \sum_{k \in i} e_{ik} w_j h^* = \sum_{n \in i} P_w \phi_n \Delta X_j q'' \\
 & + \sum_{k \in i} e_{ik} S_j \Delta X_j \bar{k}_t \frac{(T_{ii} - T_{jj})_j}{\ell Z_k} + \Delta X_j \sum_{k \in i} e_{ik} w'_j (h_{ii} - h_{jj})_j
 \end{aligned} \tag{2.23}$$

In the actual solution of the energy equation in COBRA-SFS, however, it is necessary to separate the mass continuity error from the energy error. This is done by the simple expedient of multiplying the mass continuity equation by the flowing enthalpy and subtracting the result from the energy equation. So the final form of the finite difference equation for the fluid energy in COBRA-SFS is

$$\begin{aligned}
 & \bar{A}_j \Delta X_j \frac{(\rho h - (\rho h)^n)_j}{\Delta t} + m_j (h^* - h_j) - m_{j-1} (h^* - h_{j-1}) \\
 & + \Delta X_j \sum_{k \in i} e_{ik} w_j (h^* - h_j) = \sum_{n \in i} P_w \phi_n \Delta X_j q'' \\
 & + \sum_{k \in i} e_{ik} S_j \Delta X_j \bar{k}_t \frac{(T_{ii} - T_{jj})_j}{\ell Z_k} + \Delta X_j \sum_{k \in i} e_{ik} w'_j (h_{ii} - h_{jj})_j
 \end{aligned} \tag{2.24}$$

The heat flux through the solid surfaces of the control volume, denoted by  $q''$  in the above equation, is the surface-averaged convective heat flux over the given node. If the conduction model is not used, the heat flux is simply a boundary condition specified by user input. When the conduction model is used, however, the heat flux is a calculated quantity determined in the solution of the conduction equation for heat transfer in the fuel rods or solid structure nodes (see Section 2.4). Heat transfer between the fluid and wall is modeled using empirical heat transfer coefficients, such that

$$q'' = H_{\text{surf}} (T_w - T)$$

The surface temperature of the rod or slab node,  $T_w$ , is solved for in the solid conduction energy equation (see Section 2.4). The sink temperature for the heat flux calculation is the fluid temperature corresponding to the enthalpy of the subchannel in that node.



## 2.4 Energy Conservation in Fuel Rods and Solid Structures

Heat transfer in the fuel rods and other solid structures is determined in COBRA-SFS using the following conduction equation:

$$q'' = -k \nabla T$$

It is approximated using a control volume formulation, consistent with the finite difference formulation of the fluid conservation equations. In these structures, surface area of the control volume is defined by the product of the axial height of the adjacent fluid control volume ( $\Delta X_j$ , as used in Eq. [2.24]), and the segment of the rod or wall perimeter that is connected to the fluid node. (This information is specified by user input.) A surface temperature is determined for each node by solving the energy equation. The user also has the option of modeling internal nodes within the solid structure, and solving for the local temperatures of these nodes.

For steady-state problems in which radiative heat transfer can be neglected, the noding of the fuel rods is used only to define the surface heat flux into the fluid control volumes (i.e., the sub-channels). Unheated solid structures are not modeled in the heat transfer solution, but are treated simply as adiabatic surfaces. This is the simplest form of the heat transfer solution in COBRA-SFS, and there are very few cases where it is an adequate representation of the problem. More often, it is necessary to include radiation exchange between the rods and conduction through walls, support baskets, and other unheated structures in contact with the fluid.

Conduction through unheated structures is modeled in COBRA-SFS with the "slab" energy equation, which is simply the conduction equation with boundary conditions defined by the fluid temperature and user-specified heat transfer coefficients at the node surfaces. These surfaces can also exchange energy with each other and with the fuel rods via radiation. There are two options available for modeling heat conduction in the fuel rods. For steady-state problems, it may be sufficient to solve a simplified form of the conduction equation for the cladding only to obtain surface temperatures of the fuel rods. In transient calculations, or if fuel centerline temperatures are needed, it is necessary to solve the conduction equation for the fuel pellet as well as for the cladding.

The following subsections describe these heat transfer models and their formulation in COBRA-SFS. They are coupled to the fluid conservation equations via the energy flux terms for heat transfer to the fluid, which are expressed in terms of the fluid temperature and appropriate heat transfer coefficients. The manner in which these equations are solved together is described in Section 4.0.

### 2.4.1 Rod Energy Equation

When the heat generation in the fuel rods is treated as a simple heat flux boundary condition on the fluid, it is not necessary to solve the conduction equation for the fuel rods. In that case, the surface heat flux for the fluid energy equation is calculated as

$$q'' = \frac{A_{\text{fuel}}}{\pi D_{\text{rod}}} q'''$$

The heat flux,  $q''$ , is calculated for each rod from the input values for rod geometry and the fuel volumetric heat generation rate,  $q'''$ .

If thermal radiation is important for a given problem, however, it is necessary to determine the surface temperatures of the fuel rods. For steady-state problems, it is often sufficient to solve the conduction equation over the cladding only, treating the heat generated in the fuel as a source term at the inner boundary. The total heat removed by convection varies as a function of fluid temperature and the surface heat transfer coefficient. The radiation term includes contributions from slabs and other rods within the same assembly. It is assumed that slabs and rod surfaces exchange radiant energy only in the same plane. This greatly simplifies the determination of the appropriate view factors for radiation exchange, since these are therefore needed only in two dimensions. Given the axial uniformity of fuel bundles, this is a reasonable assumption for most geometries to which COBRA-SFS is likely to be applied.

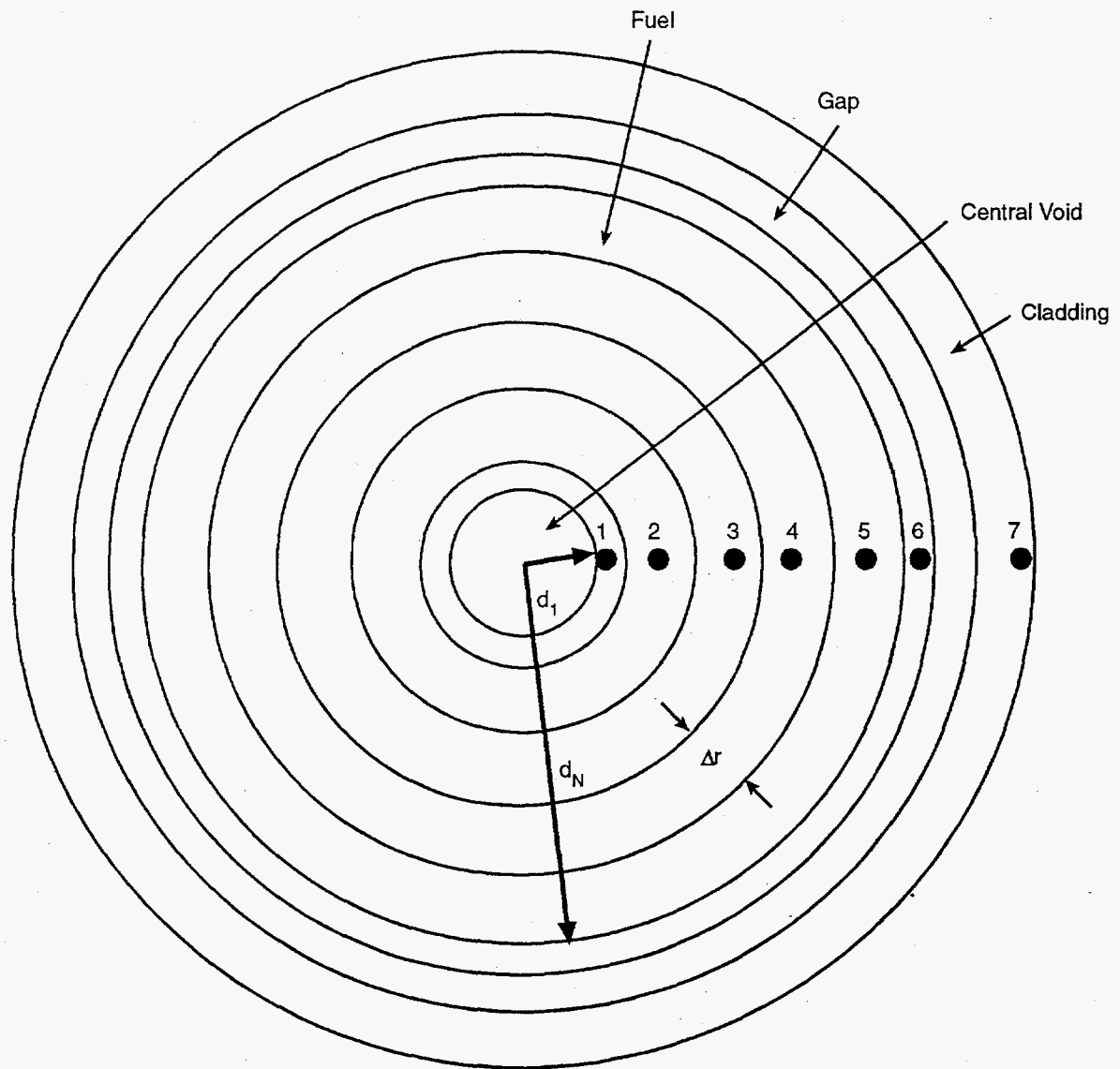
When using the option for the cladding surface conduction only, it is assumed that there is no axial heat transfer, and that the temperature is uniform around the circumference of the rod at a given axial level. The rod energy equation for the clad alone can therefore be expressed as

$$\begin{aligned}
 0 = & - \Delta X_j \sum_{nei} (\pi D_R) \phi_n H_{surf} (T_{clad} - T_i)_j \\
 & - \Delta X_j \sum_{nem} (\pi D_R) \sigma F_{nm} (T_{clad_n}^4 - T_{clad_m}^4) \\
 & - \Delta X_j \sum_{nem} (\pi D_R) \sigma F_{nm} (T_{clad_n}^4 - T_{wall_m}^4) + A_{fuel} \Delta X_j q'''
 \end{aligned} \tag{2.25}$$

The rod heat transfer model represented by Eq. (2.25) is adequate for nearly all steady-state applications of COBRA-SFS. For transient applications, or cases where the internal rod temperature distribution is important, a formulation of the conduction equation must be used that includes internal nodes for the fuel rods. In this model, separate conduction equations are written for the cladding and for the fuel. The cladding is represented with a single node, but the fuel is divided into a number of radial rings, as illustrated in Figure 2.5 (for fuel with a central void), and in Figure 2.6 (for fuel without a central void). The number of fuel nodes in a given case is determined by user input, and the nodes are assumed to be of equal radial thickness, except for the innermost and outermost nodes, which are one-half the thickness of the other nodes.

The conduction equation for heat transfer in the cladding is similar to Eq. (2.25), except that in this formulation the time-dependent energy storage term cannot be neglected, and the energy entering the cladding from the fuel does so via conduction, rather than as a source term. With these changes, the conduction equation for the cladding becomes

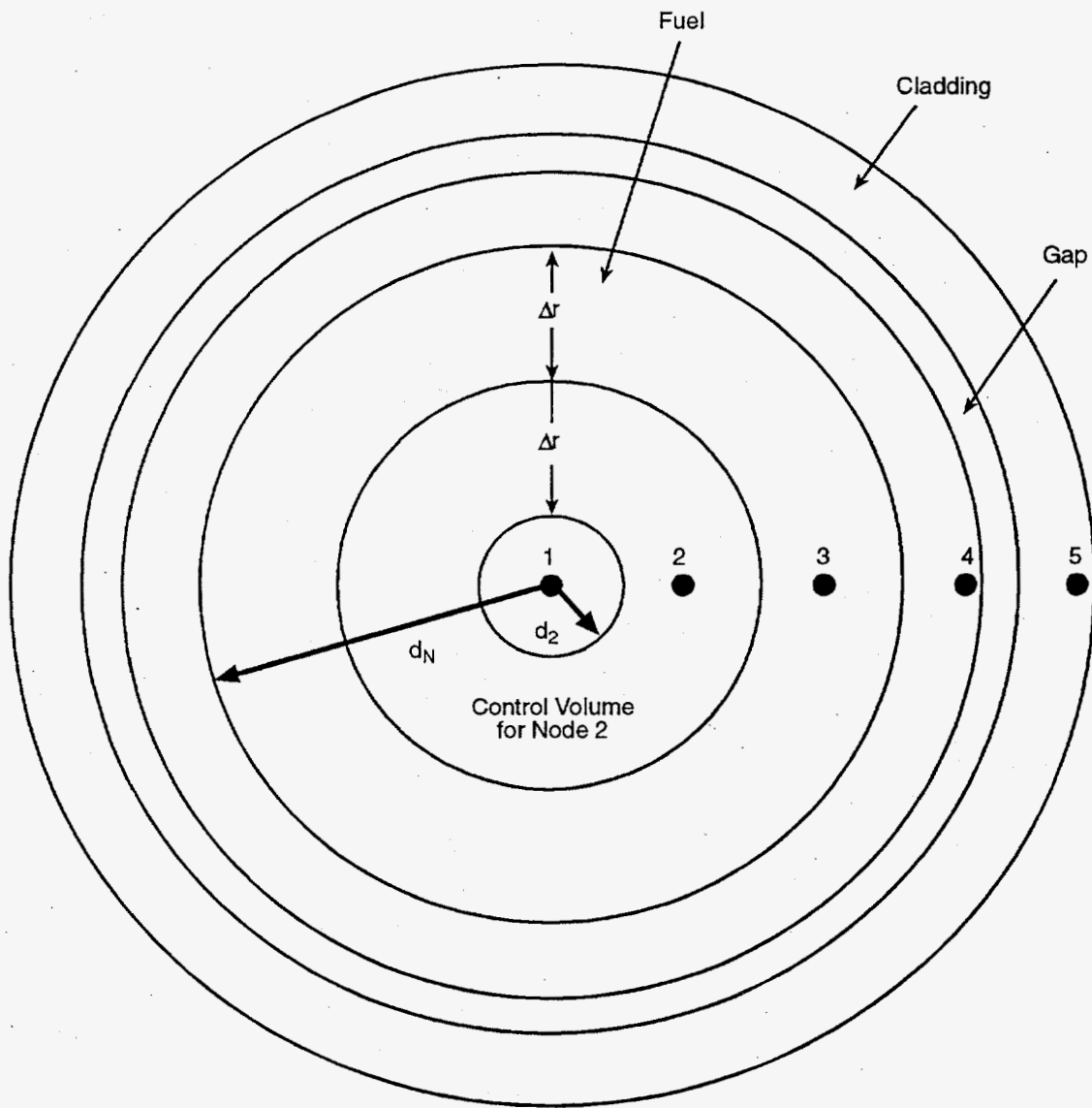
$$\begin{aligned}
 A_c \Delta X_j \rho_c c_p \frac{T_{clad} - T_{clad}^n}{\Delta t} = & - \Delta X_j \sum_{nei} (\pi D_{rod}) \phi_n H_{surf} (T_{clad} - T_i) \\
 & - \Delta X_j \sum_{nem} (\pi D_{rod}) \sigma F_{nm} (T_{clad_m}^4 - T_{clad_n}^4) \\
 & - \Delta X_j \sum_{nem} (\pi D_{rod}) \sigma F_{mn} (T_{clad_n}^4 - T_{wall_m}^4) + (\pi D_{fuel}) H_{gap} (T_{clad} - T_{fs})
 \end{aligned} \tag{2.26}$$



S9503052.5

Figure 2.5. COBRA-SFS Finite-Volume Fuel Rod Model with Central Void

The clad conduction equation is coupled to the fuel conduction equation through the gap heat transfer coefficient,  $H_{\text{gap}}$ , and the temperature difference across the gap between the fuel pellet and the clad. The gap heat transfer coefficient is an empirical parameter, defined by user input. In the clad node, heat transfer is considered in the radial direction only, with a uniform temperature distribution radially around the circumference of the rod. This is the usual approach in the fuel pellet, as well, but the conduction equations are formulated to allow consideration of azimuthal as well as radial noding. Nodes in the circumferential direction are counted with the variable  $N_{\theta}$ , and in the radial direction with  $N$ . The fuel pellet node temperatures are identified as  $T_{k,m}$  where  $k$  is the radial location (1 to  $N$ ), and  $m$  is the circumferential location (1 to  $N_{\theta}$ ).



S9503052.6

Figure 2.6. COBRA-SFS Finite-Volume Fuel Rod Model Without Central Void

The set of conduction equations in finite-difference form for each of the N fuel nodes and the cladding node, N+1, can be written as follows:

Fuel node 1 (inner-most node):

$$\begin{aligned}
 & \pi \left( \frac{d_2^2 - d_{\text{void}}^2}{4N_\theta} \right) \rho_f c_{p_f} \frac{T_{1,m} - T_{1,m}^n}{\Delta t} + \frac{\pi d_2}{\Delta r N_\theta} k_f (T_{1,m} - T_{2,m}) \\
 & + \frac{2\Delta r N_\theta}{\pi(d_1 - d_2)} k_f (2T_{1,m} - T_{1,m-1} - T_{1,m+1}) \\
 & = q_f''' \pi \left( \frac{d_2^2 - d_1^2}{4N_\theta} \right) + \frac{\pi}{N_\theta} d_1 q_{\text{void}}
 \end{aligned} \tag{2.27}$$

Fuel nodes 2 through N-1:

$$\begin{aligned}
 & \pi \left( \frac{d_{k+1}^2 - d_k^2}{4N_\theta} \right) \rho_f c_{p_f} \frac{T_{k,m} - T_{k,m}^n}{\Delta t} + \frac{\pi d_k}{\Delta r N_\theta} k_f (T_{k,m} - T_{k-1,m}) \\
 & + \frac{\pi d_{k+1}}{\Delta r N_\theta} (T_{k,m} - T_{k+1,m}) \\
 & + \frac{2\Delta r N_\theta}{\pi(d_k + d_{k+1})} k_f (2T_{k,m} - T_{k,m-1} - T_{k,m+1}) = q_f''' \pi \left( \frac{d_{k+1}^2 - d_k^2}{4N_\theta} \right)
 \end{aligned} \tag{2.28}$$

Fuel node N (fuel surface node):

$$\begin{aligned}
 & \pi \left( \frac{d_{\text{fuel}}^2 - d_N^2}{4N_\theta} \right) \rho_f c_{p_f} \frac{T_{N,m} - T_{N,m}^n}{\Delta t} + \frac{\pi d_N}{\Delta r N_\theta} k_f (T_{N,m} - T_{N-1,m}) \\
 & + \frac{\pi d_{\text{fuel}} H_g}{N_\theta} (T_{N,m} - T_{c,m}) \\
 & + \frac{2\Delta r N_\theta}{\pi(d_{\text{fuel}} + d_N)} (2T_{N,m} - T_{N,m-1} - T_{N,m+1}) = q_f''' \pi \left( \frac{d_{\text{fuel}}^2 - d_N^2}{4N_\theta} \right)
 \end{aligned} \tag{2.29}$$

Cladding Node N + 1:

$$\begin{aligned}
 & \pi \left[ \frac{d_{\text{clad}}^2 - (d_{\text{clad}} - 2t_{\text{clad}})}{4N_{\theta}} \right] \rho_c c_p c \frac{T_{c,m} - T_{c,m}^n}{\Delta t} + \pi \frac{d_{\text{fuel}} H_{\text{gap}}}{N_{\theta}} (T_{c,m} - T_{n,m}) \\
 & + \frac{\pi d_{\text{clad}} H_{\text{surf}}}{N_{\theta}} (T_{c,m} - T_{\text{fluid},m}) \\
 & + \frac{\Delta r N_{\theta} k_c}{\pi (d_{\text{clad}} - t_{\text{clad}})} (2T_{c,m} - T_{c,m-1} - T_{c,m+1}) \\
 & + \sum_{n \in (N+1)} (\pi D_{\text{rod}}) \sigma F_{N+1,n} (T_c^4 - T_{c,n}^4) + \sum_{m \in (N+1)} (A_s / \Delta X_j) \sigma F_{(N+1),m} (T_c^4 - T_{w,m}^4) = 0
 \end{aligned} \tag{2.30}$$

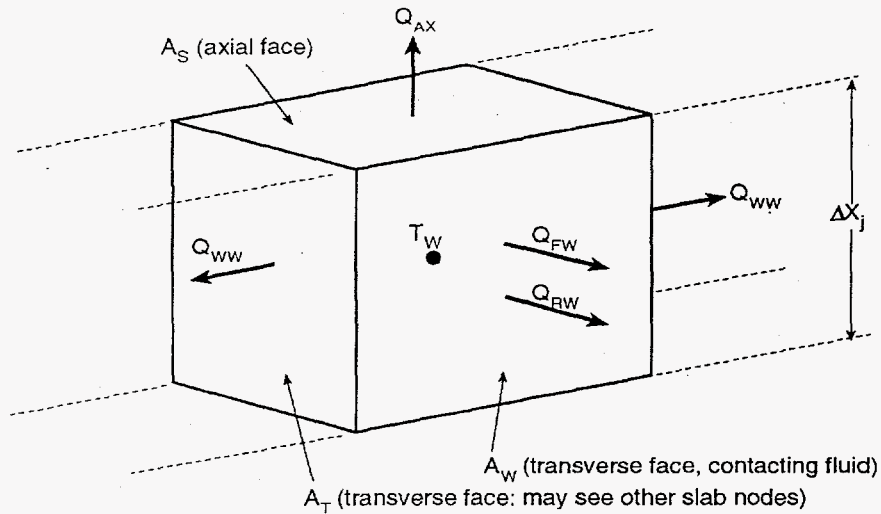
In this model, it is assumed that axial heat transfer is negligible, heat is generated uniformly throughout the fuel at a given axial location, and material properties of the fuel do not vary significantly with the radial variation in temperature.

#### 2.4.2 Solid Structure Energy Equation

The heat transfer model for the solid structure nodes is formulated for an arbitrary control volume that can exchange energy with the fluid and with other solid structure nodes via conductive or radiative heat transfer. In addition, the solid structure nodes can exchange energy with the rods by means of radiation. The generic control volume for a solid structure node, (also called a "slab" node, to differentiate it from a fuel rod node), is illustrated in Figure 2.7. The axial length of the control volume is denoted by  $\Delta x_j$ , which corresponds to axial noding of the fluid subchannels. The cross-sectional area for axial heat transfer from a slab node is defined by user input. A slab control volume may have any number of surfaces connected to adjacent slab nodes or fluid subchannels. These connections and their dimensions are defined by user input.

In addition to conductive and convective heat transfer between the slab node and adjacent slab or fluid nodes, the control volume can also exchange energy via thermal radiation with the surfaces of fuel rods and other slabs. Given the above assumptions and definitions, the conduction equation for the slab node can be written in finite difference form as

$$\begin{aligned}
 A_s \Delta X_j \rho_w c_{p_w} \frac{T_w - T_w^n}{\Delta t} = & - \Delta X_j \sum_{i \in m} P_w H_{\text{surf}} (T_{\text{wall}} - T_i) \\
 & - \sum_{m \in i} U (T_{\text{wall}} - T_{\text{wall},m}) \\
 & + \sum_{m \in i} A_w \sigma F_{im} (T_{\text{wall}}^4 - T_{\text{wall},m}^4) - \sum_{n \in i} A_T \sigma F_{nm} (T_{\text{wall}}^4 - T_{\text{clad},n}^4) \\
 & + [U_j (T_{\text{wall}} - T_{\text{wall},j+1}) + U_{j-1} (T_{\text{wall}} - T_{\text{wall},j-1})]
 \end{aligned} \tag{2.31}$$



S9503052.7

Figure 2.7. Solid Control Volume

As with the rod equation, radiation is assumed to occur only in a given axial plane. Conduction between two slab nodes is modeled using a composite thermal conductance,  $U$ , which accounts for the heat transfer area, the thermal conductivity of the slab materials, and any gap resistance or thermal radiation at the slab interface. In the axial direction, a composite thermal conductance through the top and the bottom of the slab node is calculated, denoted as  $U_j$  and  $U_{j-1}$ , respectively. The actual values used in a given problem are determined from user input of appropriate empirical conduction resistances for the various node connections. (See Section 3.2 for a discussion of the composite thermal conductance as a constitutive model in the code, and to Section 6.2.6 for the group SLAB input instructions.)

## 3.0 Constitutive Models

The three-equation model presented in Section 2.0 fully describes the fluid behavior in systems modeled using COBRA-SFS in terms of the independent variables of pressure, mass flow rate, and fluid enthalpy. With the addition of the heat transfer models for the fuel rods and other solid structures, a complete flow and heat transfer system can be described. To solve the equations, however, closure relations are required to define terms in the equations that are not amenable to calculation from first principles. This section presents the empirical models used in COBRA-SFS to achieve closure of the equation set. These models are reasonable within the intended application of the code, and every attempt has been made to adequately verify the formulation. However, it must be noted that constitutive models are empirical approximations, and can always be improved. They should be used with careful consideration of their range of applicability, and it is to be hoped that eventually some state-of-the-art advances will come along and make them obsolete. In that happy event, it should be a relatively simple matter to replace the existing models with the improved ones. This is made particularly simple by the fact that most of the empirical models in the code are defined by user input.

### 3.1 Fluid Flow Models

The momentum exchange terms are a primary source of empiricism in the fluid flow equations in COBRA-SFS. As discussed in Section 2.2 in the derivation of the subchannel partial differential equations for momentum, the solid surface stress tensor is represented by wall friction and form drag correlations. The fluid stress tensor is replaced by velocity-dependent models for fluid-fluid shear. Energy and momentum exchange due to turbulent mixing must also be represented by empirical models, since turbulence is neglected in the primary formulation of the mixture balance laws over the control volume. The following subsections describe these constitutive models as currently implemented in COBRA-SFS.

#### 3.1.1 Wall Friction

The shear stress terms in the axial momentum equation for rod bundle geometries were defined in Section 2.2 by an approximation of the shear stress integral in the momentum balance. In the normal subchannel formulation, this integral was approximated by a wall friction term that included an empirical friction factor,  $f$ , as shown in Eqs. (2.12) and (2.14). An alternative formulation was also presented for wall shear in channels modeling an open flow region. This formulation is a function of fluid properties and geometry only, and was presented in Eqs. (2.12a) and (2.14a). Although the fluid shear stress terms in these equations are approximations of the mathematical model for shear stress, they are not constitutive models in the usual sense of the term, since they are not based on fits to data. Therefore, the friction factor is the only constitutive relation used in the wall shear stress term. The friction factor used in COBRA-SFS is the Darcy-Weisback friction factor, a dimensionless number determined from experimental measurements of axial pressure drop in a wide range of geometries and flows. These data are primarily for flow in pipes and tubes, but it is assumed that these flow geometries scale to subchannel flow by means of the hydraulic diameter.



The friction factor correlation for turbulent flow is expressed in COBRA-SFS as

$$f = aRe^b + cRe^d + e \quad (3.1)$$

The coefficients a, b, c, d, and e are empirical constants specified by user input, and Re is the local subchannel Reynolds number, defined as

$$Re = \frac{mD_h}{A\mu} \quad (3.2)$$

The user has the option of specifying several friction factor correlations of the form shown in Eq. (3.1), in order to model different friction losses in different subchannels within the problem (refer to Section 6.2.9, which contains the input instructions for group DRAG).

The friction factor correlation for laminar flow is expressed in COBRA-SFS as

$$f_l = a_l Re^{b_l} + c_l \quad (3.3)$$

As in the correlation for turbulent flow above, the empirical constants  $a_l$ ,  $b_l$ , and  $c_l$  are specified by user input. The actual value of the friction factor used in the axial momentum equation for a given subchannel node is determined as the maximum of the laminar and turbulent correlation values at the local Reynolds number.

The friction factor correlations described by Eqs. (3.1) and (3.3) are based on a Reynolds number defined with the subchannel-averaged fluid viscosity. There is in reality a distribution of viscosity due to the subchannel radial temperature profile in the fluid. As an option specified by user input, COBRA-SFS includes an empirical correlation to account for the viscosity variation near a heated surface. The relationship used is that developed by Tong (Tong 1968), and has the form

$$f_{\mu_{wall}} = f \left[ 1.0 + \frac{P_h}{P_w} \left[ \left( \frac{\mu_{wall}}{\mu_{bulk}} \right)^{0.6} - 1.0 \right] \right]$$

The viscosity at the wall,  $\mu_{wall}$ , is determined from the fluid properties using an estimated fluid temperature at the wall. This temperature is calculated from a simple heat balance as

$$T_{wall} = T_{bulk} + \frac{q'}{P_h H}$$

It is important to note that the calculated wall temperature used in this correlation is based on the total heat input to the channel in the given node, over the total heated perimeter of the channel. It is assumed for the purposes of this calculation that the heat input to the channel is uniform over the

perimeter, when in fact the heat inputs from the various rods surrounding the channel may be different. However, these differences will in most cases be relatively small. Any error this might introduce into the approximation will probably be quite small, at least in relation to the uncertainty inherent in the Tong viscosity correction correlation itself.

The user should also take note of the fact that the Tong viscosity correction correlation was derived from data obtained with water as the working fluid. In most COBRA-SFS applications, the coolant is a gas such as air, nitrogen, argon, or helium. The correlation is formulated in terms of the fluid properties, and therefore should be applicable to different fluids, but it has not been validated for fluids other than water. In general, the wall viscosity correction will be very small, and in many cases it will make more sense to ignore it. It is not usually needed for cases where the working fluid is a gas with a high thermal conductivity, such as helium.

### 3.1.2 Form Drag Models

In addition to the momentum effects of friction at the wall, local obstructions to the flow due to structures such as grid spacers, orifice plates, and local variations in the flow area result in axial pressure losses. The reversible component of such losses is calculated in the axial momentum equation, if the area change is included in the COBRA-SFS model. The irreversible component must be explicitly included by means of the constitutive model for form drag.

The pressure drop across a local obstruction is expressed as

$$\Delta P = \frac{K|m|m}{2g_c \rho A^2} \quad (3.4)$$

The loss coefficient  $K$  is an empirical parameter determined from measured data in similar structures. In COBRA-SFS, these local loss coefficients can be modeled as constants or as functions of the local Reynolds number with a form similar to that of the wall friction correlation in Eq. (3.1). Refer to Section 6.2.9 for the input instructions for group DRAG.

In the formulation in Eq. (3.4), the cross-sectional area  $A$  refers to the nominal channel flow area, and not to the flow area at the obstruction itself. The user must be aware of this definition when determining appropriate input values for  $K$  from pressure drop measurement data. In some cases, the reported  $K$  values are determined based on the flow area at the obstruction. When using such values as input for a COBRA-SFS model, the measured loss coefficients must be scaled by the area ratio, so that the input loss coefficient is defined as

$$K_{\text{input}} = K_{\text{measured}} \left( \frac{A_{\text{nominal}}}{A_{\text{obstruction}}} \right)^2$$

The shear stress integral for the lateral momentum equation is modeled with a simpler approximation than that used for the axial momentum equation. Both wall friction and form drag are treated

with a single approximation of the pressure loss coefficient. This permits formulation of the pressure loss in terms of known geometric quantities, such that

$$\Delta P = \frac{K_G |w| w}{2g_c \rho S^2} \quad (3.5)$$

In rod assemblies, the coefficient  $K_G$  can be thought of as the form loss for flow through the gap between two adjacent fuel rods. The value for  $K_G$  is dependent on the geometry of the given problem and must be specified by user input (refer to the input instructions for group DRAG, in Section 6.2.9).

### 3.1.3 Turbulent Mixing Models

The turbulent mixing model in COBRA-SFS approximates the momentum effects of turbulence by defining a fluctuating cross-flow per unit length,  $w'$ , that represents an equal mass exchange between adjacent channels. This fluctuating cross-flow is related to the turbulent eddy diffusivity by the definition

$$w' = \epsilon_t \rho \left[ \frac{S}{\ell} \right]$$

The local value of this fluctuating cross-flow is not determined directly from the eddy diffusivity, however, since this is not an easily determined quantity over the flow field as modeled in subchannel analysis. Instead, the fluctuating cross-flow is calculated from an empirical relation using the geometry of the gap, the average axial flow rate in the gap, and experimentally determined mixing coefficients.

There are four such turbulent mixing correlations included in COBRA-SFS (Rogers and Todreas 1968; Ingesson and Hedberg 1970; Rogers and Rosehart 1972). They are selected by user input, and are formulated as follows:

$$w_T = a S_k \bar{G}$$

$$w_T = a \text{Re}^b \bar{D} \bar{G}$$

$$w_T = a \text{Re}^b \frac{S_k}{\ell_k} \bar{D} \bar{G}$$

$$w_T = a \text{ Re}^b S_k \bar{G}$$

where

$$\text{Re} = \frac{\bar{G} \bar{D}}{\mu}$$

$$\bar{D} = 4 (A_{ii} + A_{jj}) / (P_{w_{ii}} + P_{w_{jj}})$$

$$\bar{G} = (m_{ii} + m_{jj}) / (A_{ii} + A_{jj})$$

The user may specify different correlations or different coefficients for each assembly type (refer to the input instructions for group HEAT, in Section 6.2.8).

## 3.2 Energy Exchange Models

As formulated in Section 2.0 above, it is implicitly assumed that the surface heat flux terms that appear in the fluid energy equation and in the conduction equations for the fuel rods and solid structures are known, or can be calculated from known quantities. In actual practice, these heat fluxes depend on convective heat transfer, radiative heat transfer, and contact conductance, all of which can be determined only with the aid of empirical heat transfer models. Constitutive relationships are required for surface heat transfer coefficients, radiation exchange factors, and contact conductance. The following subsections describe how these models are implemented in COBRA-SFS.

### 3.2.1 Convective Heat Transfer Correlations

Convective heat transfer is modeled in COBRA-SFS using a heat transfer coefficient defined in terms of the Nusselt number. The heat transfer coefficient is defined by the expression

$$H = \text{Nu} \frac{k}{D_h}$$

The Nusselt number is defined as a function of Reynolds number and Prandtl number, using the empirical relation

$$\text{Nu} = A \text{ Re}^a \text{ Pr}^b + B$$

The local heat transfer coefficient is calculated in the code using this formula. The values for the empirical constants are specified by user input, and different correlations can be specified for different channels (refer to Section 6.2.8, containing the input instructions for group HEAT).

### 3.2.2 Fluid Conduction Shape Factor

Conduction in the fluid is modeled directly in the formulation of the fluid energy equation in COBRA-SFS, but only in the lateral direction. Axial conduction is assumed negligible. Lateral conduction is calculated through the gaps over the centroid length between the subchannels. However, the exact length of the centroid is a matter of definition, and may or may not be the appropriate length for the conduction path. The fluid conduction model in COBRA-SFS therefore includes an option to specify the centroid length for conduction as a value different from that used for the lateral momentum equation. This is implemented by defining the conduction length  $\ell_c$  in terms of the centroid length  $\ell$  using an empirical shape factor,

$$\ell_c = \ell Z_k$$

The appropriate value for  $Z_k$  is problem-dependent, and general guidelines are not readily available. Experience has shown that with air or helium as the working fluid, a value near unity gives good agreement with experimental data.

### 3.2.3 Solid-to-Solid Conduction

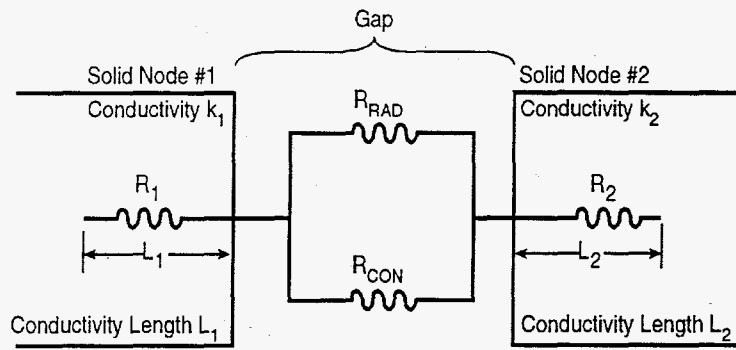
In theory, the heat transfer due to conduction between adjacent solid nodes can be calculated exactly using the conduction equation as formulated in Section 2.4. In practice, the contact conductance at the interface between adjacent structures is not always known with great accuracy. Since this contact conductance (or resistance, if considered as a reciprocal) plays a large part in determining the rate of heat transfer, accurate characterization of this parameter becomes a very important modeling consideration.

In COBRA-SFS, contact conductance between adjacent solid structure nodes is modeled using a composite thermal conductance,  $U$ . This conductance includes the heat transfer area of the interface and the thermal conductivity of the node materials. In addition, it also accounts for any gap resistance due to imperfect contact between the two surfaces, and includes thermal radiation across the gap. The components of the total thermal resistance are shown schematically in Figure 3.1. The terms  $R_1$  and  $R_2$  denote the thermal resistance from the center of each slab node to the interface between them, and are defined by user input. The total series resistance of the solid portion of the heat transfer path between the two slab nodes is given by

$$R_{\text{solid}} = R_1 + R_2$$

The resistance for node  $n$  is defined as

$$R_n = \frac{L_n}{A_1} k$$



S9503052.8

Figure 3.1. Solid-to-Solid Resistance Network

The values of  $L_n$  and  $A_i$  are determined from user input, where  $L_n$  is the distance from the center of the slab node to the interface (as shown in Figure 3.1), and  $A_i$  is the area of the interface between the two nodes.

The radiative heat transfer across the interface is approximated as radiation between two parallel plates where the surfaces are assumed to be grey bodies. The thermal resistance is given by the relation

$$R_{\text{RAD}} = \frac{\frac{1}{\epsilon_1} + \frac{1}{\epsilon_2} - 1}{\sigma A (T_1^2 + T_2^2)(T_1 + T_2)}$$

In this formula,  $T_1$  and  $T_2$  are absolute temperatures (in the code, the units are degrees Rankine),  $\epsilon_1$  and  $\epsilon_2$  are the surface emissivities of the slab nodes, and  $\sigma$  is the Stefan-Boltzmann constant.

In parallel with the radiation across the gap, the conduction resistance is also calculated based on the user-specified material properties for the fluid in the gap. The total gap resistance, therefore, can be expressed as

$$\frac{1}{R_{\text{GAP}}} = \frac{1}{R_{\text{RAD}}} + \frac{1}{R_{\text{CON}}}$$

The overall thermal conductance between the adjacent slab nodes can be calculated as

$$U = \frac{1}{R_{\text{SOLID}} + R_{\text{GAP}}}$$

This formulation is used for heat transfer between slab nodes in both the transverse and the axial directions, as shown in Eq. (2.31). The appropriate composite conductance  $U$  for any given connection is calculated from the user-specified definitions of solid-to-solid connections and slab node geometry (refer to the input instructions for group SLAB in Section 6.2.6).

### 3.2.4 Radiation Exchange Factors

The radiation heat transfer between surfaces is modeled using radiation exchange factors. The net rate of radiant energy outflow from surface  $I$  to surface  $j$ , denoted by  $Q_{ij}$ , is defined in terms of an exchange factor,  $F_{ij}$ , such that

$$Q_{ij} = A_i F_{ij} \sigma (T_i^4 - T_j^4) \quad (3.5)$$

The quantity  $F_{ij}$  defines the fraction of the total amount of radiant energy emitted from surface  $I$  that actually reaches surface  $j$  and is absorbed. This includes all the paths by which radiation may reach surface  $j$  from surface  $I$ , including direct and reflective paths. This has been termed a *grey body* view factor, and gives a much more realistic estimate of the energy exchange between surfaces due to radiation than can be obtained using standard 'black-body' view factors. Black body view factors, as the name implies, assume that all incident radiation is absorbed, and none reflected. For surfaces with an emissivity of 1.0, the grey body view factors reduce to the black body factors, since an emissivity of 1.0 implies black surfaces for radiation heat transfer, by definition.

The radiation exchange factors  $F_{ij}$  can be calculated from the geometry of a particular problem and the ideal black body view factors for the given surfaces. However, this is not a trivial calculation for a geometry of any complexity, such as a fuel rod array in a canister. The preferred means for calculating these values for input to COBRA-SFS is to use the auxiliary code RADGEN, which solves the set of equations (Cox 1977);

$$\sum_{i=1}^n [B_{ki} \left( \frac{1.0 - \epsilon_i}{\epsilon_i} \right) - \frac{\delta_{ki}}{\epsilon_i}] F_{ij} = -B_{kj} \epsilon_{kj}$$

Every surface  $I$  sees every surface  $j=1,2,3,\dots,n$ , and by reflection from each surface  $j$ , also sees every surface  $k=1,2,3,\dots,n$ , where  $n$  is the total number of surfaces that can exchange energy by radiation.

For enclosures that consist simply of wall nodes, the system of equations can be solved in the COBRA-SFS code, as part of the input processing. The user supplies only the black body view factors for the surfaces, and their emissivities. The code calculates the appropriate grey body view factors internally. For enclosures that include rods, however, the code expects the grey body view factors as input. These must be calculated before-hand, using the RADGEN code, or any similar code that produces the grey body view factors in the appropriate format. Correct and consistent input for the grey body view factors is absolutely essential for the radiation model in COBRA-SFS to work properly. Where possible, input checks have been installed in the code, but the conscientious user would be well advised to study carefully the input instructions for the RADG group of the COBRA-SFS input (see Section 6.2.7), and those for the RADGEN code (see Section 6.3).

## 4.0 Numerical Solution Methods

The solution of the governing equations for fluid flow and heat transfer in COBRA-SFS is fully implicit and proceeds iteratively through a series of steps that addresses each of the conservation equations in turn. Within an iteration, the code solves the momentum equations for the velocity field, then the energy equations for the temperatures and enthalpies, and then the mass continuity equation for the pressure field. Within this simplistic description, however, lies a wealth of intricate detail.

The basic fluid solution is applicable to single-phase flow at very low velocities, with or without buoyancy-driven natural circulation. The code is also capable of resolving the flow and pressure fields for systems in which the net flow is zero. This calculational flexibility is a feature of the numerical solution method for the fluid equations that COBRA-SFS has inherited from its immediate predecessor, COBRA-WC (George et al. 1980). In spent fuel assemblies and storage or shipping casks, the flow field may be fairly simple to obtain, but the strong coupling of the fluid energy equation and the heat transfer models for the solid structures presents special problems of stability and convergence rates. In particular, the energy equations for the fluid and solid structure nodes must be solved simultaneously, and the rate of convergence of the temperature solution tends to dominate convergence behavior, rather than the elimination of momentum or mass conservation errors.

The following subsections describe the flow and energy solutions in the code in some detail. Section 4.1 presents the fluid flow solution, and Section 4.2 the solution of the energy equations. It should be borne in mind, however, that although the solutions are described separately, they are performed together within an iteration of the solution procedure.

### 4.1 Fluid Flow Solution

The finite difference equations for mass continuity, (Eq. [2.20]), momentum, (Eqs. [2.21] and [2.22]), and energy, (Eq. [2.23]) for the fluid are solved using the RECIRC solution method, adapted from the COBRA-WC code. In this method, the set of equations is solved iteratively to obtain the flow and pressure fields. The primary advantage of the RECIRC method is that it is applicable to reverse and recirculating flows, such as those occurring in storage systems cooled by natural circulation. RECIRC uses a Newton-Raphson technique similar to the one developed by Hirt (Hirt and Cook 1972) to solve the conservation equations, but it has been made implicit in time, as was done in the SABRE code (Gosman et al. 1973). The solution method is quite complex, and it is summarized here before examining each separate step in detail.

The RECIRC flow field solution is divided into two parts: a tentative flow solution and a pressure solution. The tentative flow solution is achieved by iteratively sweeping the bundle from inlet to exit. In each sweep, tentative axial flows,  $\bar{m}$ , and cross-flows,  $w$ , are computed for the bundle by evaluating the two linearized momentum equations with current values for pressure and other independent variables. After all tentative flows and cross-flows have been computed at all axial levels, the flows and pressures are adjusted to satisfy continuity by a Newton-Raphson method. The flow field obtained in this pressure solution is then used in the energy equation to obtain an enthalpy and fluid properties distribution for the next axial sweep. A flow chart of this procedure is shown in Figure 4.1. The tentative flow solution is described in more detail in Section 4.1.1 and the pressure solution is described in Section 4.2.2.



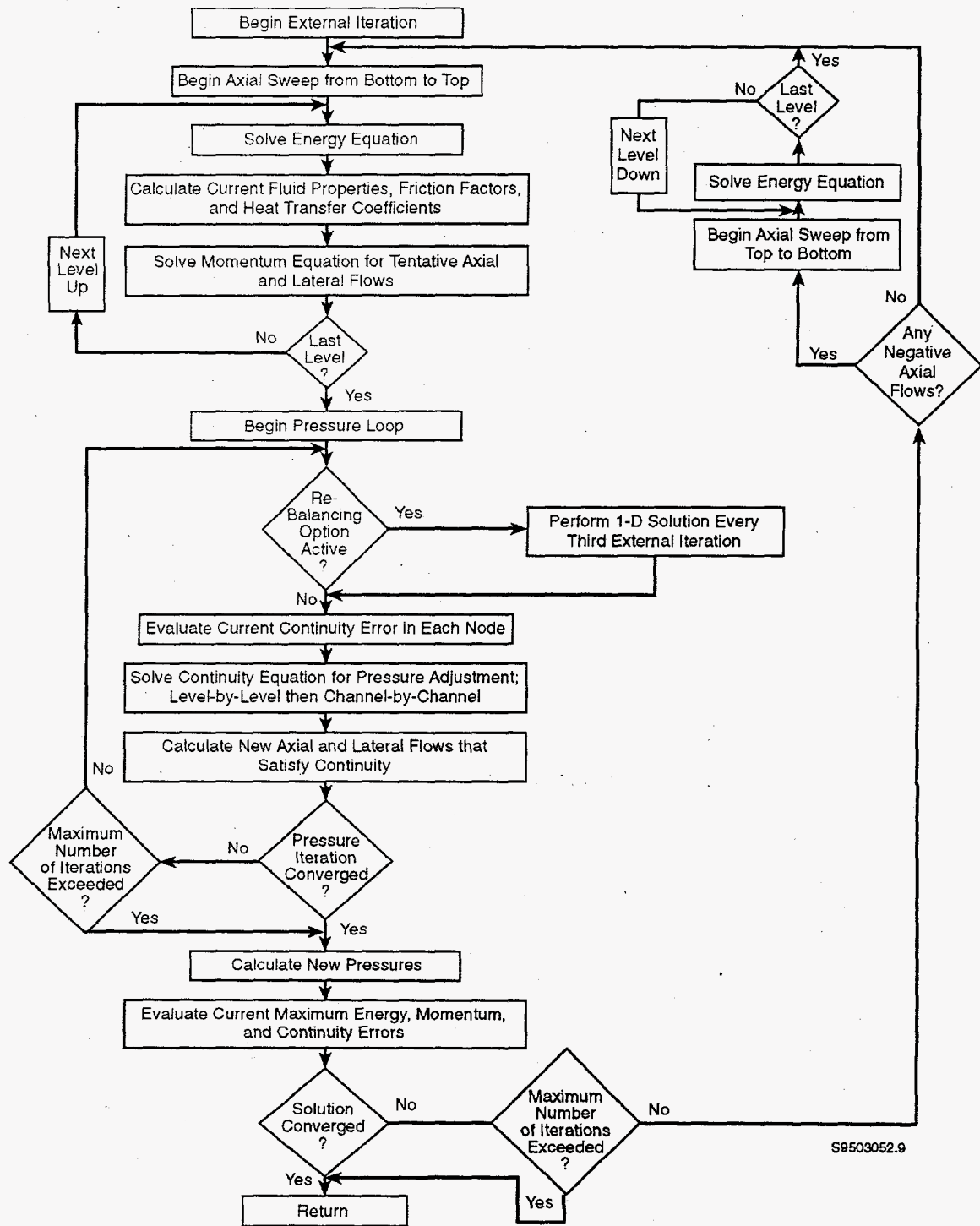


Figure 4.1. Flow Chart of the RECIRC Solution Scheme

### 4.1.1 Tentative Flow Solution

The first step in the RECIRC flow field solution is to solve the axial and transverse momentum equations for the tentative flow rates  $\bar{m}$  and  $\bar{w}$ . Since the axial flow in channel I is affected by axial flows in adjacent channels through the crossflow terms, the tentative flows,  $\bar{m}_j$ , in all channels must be solved for simultaneously at an axial level. The axial momentum equation, (Eq. [2.21]), is linearized and applied to all channels at a level to form a system of N equations, where N is the number of channels, such that

$$[A] \bar{m}_j = -g_c A_j (\bar{P}_j - \bar{P}_{j-1}) + [B] \quad (4.1)$$

The diagonal elements of [A] are defined by

$$A_{jj} = \frac{\Delta X_j}{\Delta t} + \delta_j \bar{U}_j - \delta_{j-1} \bar{U}_{j-1} + \sum_{kei} \delta_k e_{ik} \bar{V}_k \frac{S_k}{A_j} \Delta X_j + \left( \frac{f}{D_h} + \frac{K}{\Delta X_j} \right) \Delta X_j \left| \frac{\bar{m}_j}{\rho_j A_j} \right| + \sum_{kei} \frac{w'_k}{\bar{\rho}_j A} \Delta X_j \quad (4.2)$$

The switch function,  $\delta$ , depends of the direction of flow and is defined as

$$\delta_j = \begin{cases} 0 & \text{if } \bar{U}_j \leq 0 \\ 1 & \text{if } \bar{U}_j > 0 \end{cases}$$

$$\delta_{j-1} = \begin{cases} 0 & \text{if } \bar{U}_{j-1} > 0 \\ 1 & \text{if } \bar{U}_{j-1} < 0 \end{cases}$$

$$\delta_k = \begin{cases} 0 & \text{if } e_{ik} \bar{V}_k \leq 0 \\ 1 & \text{if } e_{ik} \bar{V}_k > 0 \end{cases}$$

The off-diagonal elements of [A] contain zeros except for the columns representing channels adjacent to channel I, where the elements are defined as

$$A_{jn} = (1 - \delta_k) e_{ik} \bar{V}_k \frac{S_k \Delta X_j}{A_n} - \frac{w'_k}{\rho_n A_n} \Delta X_j \quad (4.3)$$

In this definition, the subscript n refers to a channel adjacent to I, connected through a gap k such that I is either the ii or jj channel of the pair. The index for n, therefore, is defined as  $n = ii + jj - I$ .

The vector [B] consists of the source terms, and is calculated as

$$B_i = \bar{A}_j \Delta X_j \rho_j g \cos\theta + \frac{1}{2} \left( \frac{f}{D_h} + \frac{K}{\Delta X_j} \right) \Delta X_j \left| \frac{m_j}{\rho_j A_j} \right| m_j - (1-\delta_j) m_{j+1} \bar{U}_j + (1-\delta_{j-1}) m_{j-1} \bar{U}_{j-1} + \frac{\Delta X_j}{\Delta t} m_j^n \quad (4.4)$$

The pressures and flow rates in Eq. (4.4) are those generated during the previous pressure iteration, except for  $m_{j-1}$ , which is the predicted axial flow rate obtained in the solution for the previous level. To ensure that friction effects have the proper weight when differentiating Eq. (4.1) with respect to pressure, the linearized pressure loss is defined as

$$\Delta P_f = -\frac{1}{2} \left( \frac{f}{D_h} + \frac{K}{\Delta X_j} \right) \Delta X_j \left| \frac{m_j}{\rho_j A_j} \right| (2\tilde{m}_j - m_j)$$

The matrices of Eq. (4.1) are stored in arrays consisting of only the non-zero elements, and is solved by direct elimination at each level.

Once a set of tentative axial flows,  $\tilde{m}$ , has been obtained, the tentative cross-flows,  $\tilde{w}$ , are calculated. Since the transverse momentum equation does not depend on cross-flows in other gaps, a simultaneous solution is not necessary. The tentative cross-flows in each gap are calculated using the linearized form of Eq. (2.22), as

$$\tilde{w}_j = \frac{M_{\text{numerator}}}{M_{\text{denominator}}} \quad (4.5)$$

where

$$M_{\text{numerator}} = \frac{S \Delta X_j g_c}{\ell} (P_{ii} - P_{jj})_{j-1} - (1-\delta_j) w_{j+1} \bar{U}_j + (1-\delta_{j-1}) w_{j-1} \bar{U}_{j-1} + \frac{\Delta X_j}{\Delta t} w_j^n + \frac{1}{2} K_G \left| \frac{w_j}{\rho_j S} \right| w_j \frac{\Delta X_j}{\ell}$$

$$M_{\text{denominator}} = \frac{\Delta X_j}{\Delta t} + \delta_j \bar{U}_j - \delta_{j-1} \bar{U}_{j-1} + K_G \left| \frac{w_j}{\rho_j S} \right| \frac{\Delta X_j}{\ell}$$

The pressures and flow rates used in Eq. (4.5) are those generated during the previous pressure iteration, except for the axial velocities at  $j-1$ , which are based on the flow rate predicted for the previous axial level in the current iteration. The tentative flow field is determined for each channel and gap by sweeping the assembly from inlet to exit. The resulting tentative flows and cross-flows satisfy the linearized momentum equations, but they do not, in general, satisfy continuity.

The efficiency of the overall solution can be enhanced if the flow rate used to define the donor cell axial momentum flux is forced to satisfy continuity during the tentative flow axial sweep. This is accomplished by defining a set of "predicted" axial flows  $m_j$  at level  $j$  that are used to evaluate the tentative flow rate at level  $j+1$ . The predicted flows are determined by solving the combined linearized momentum and continuity equations while assuming that the previous axial level flows are fixed. At level  $j$  only, the continuity error for each cell is calculated as

$$E_j = \bar{A}_j \Delta X_j / \Delta t (\rho - \rho^n)_j + \tilde{m}_j - m'_{j-1} + \Delta X_j + \sum_{k \in \psi_j} e_{ik} w'_{j-1} \quad (4.6)$$

The predicted flows,  $m'_{j-1}$  and  $w'_{j-1}$ , are calculated at the previous axial level and  $\tilde{m}_j$  is the just-computed tentative axial flow. It is then possible to calculate an estimate of the pressure changes needed to satisfy continuity, using the derivatives of the tentative flows with respect to pressure, from the Newton-Raphson expression

$$\delta P'_{j-1} + \frac{\partial \tilde{m}_j}{\partial P_{j-1}} + \sum_{k \in i} \delta P'_{n,j-1} \frac{\partial \tilde{w}_j}{\partial P_{nj-1}} = - E_j$$

The above equation is written for a given channel  $I$ , and  $n$  in the summation represents a channel connected to  $I$  through a gap  $k$ , such that  $n = (ii+jj)-1$ . The form of the pressure derivatives is described below, in Section 4.1.2, with the discussion of the pressure and continuity solution.

Once a set of estimated pressure changes has been calculated, the predicted flows to be used in the donor momentum flux terms for the next axial level are calculated using

$$m_j = \tilde{m}_j + \frac{\partial \tilde{m}_j}{\partial P_{j-1}} \delta P'_{j-1} \quad (4.7)$$

$$w_j = \tilde{w}_{j-1} + \frac{\partial \tilde{w}_j}{\partial P_{j-1}} (\delta P'_{ii,j-1} - \delta P'_{jj,j-1}) \quad (4.8)$$

The tentative flows and pressures at  $j$  are not updated using this information, however. These predicted flows are used only to provide a better estimate of the donor momentum flux distribution to the momentum equations at the next axial level.

The predicted average axial velocity in the gap, used in Eq. (4.5), is calculated as

$$\bar{U}_j = \left( \frac{m_{ji}}{\bar{\rho}_{ii}} + \frac{m_{jj}}{\bar{\rho}_{jj}} \right) \frac{1}{(A_{ii} + A_{jj})}$$

For forced convection problems, the convergence is substantially improved if the predicted value for the axial momentum flux is used. For problems with negative flows or buoyancy-dominated flows, however, the use of predicted momentum flux can be counterproductive. For this reason, the calculation of a predicted momentum flux is automatically skipped in the solution procedure if negative flows occur in an assembly.

#### 4.1.2 Pressure and Linearized Flow Solution

After all tentative axial flows and crossflows have been computed at all axial levels in all assemblies, the pressures and flows are adjusted to satisfy continuity. For a given boundary condition, the flow and pressure field in each assembly are locally independent of all the other assemblies. Therefore, the equations for flow and pressure are solved on an assembly-by-assembly basis. The first step in this process is to compute the continuity error,  $E_j$ , in all cells, using the expression

$$E_j = \bar{A}_j \frac{\Delta X_j}{\Delta t} (\rho - \rho^n)_j + \tilde{m}_j - \tilde{m}_{j-1} + \Delta X_j \sum_{k \in i} e_{ik} \tilde{w}_j \quad (4.9)$$

The adjustments in axial flows and crossflows necessary to satisfy continuity are defined as

$$\Delta m_j = m_j - \tilde{m}_j$$

$$\Delta w_j = w_j - \tilde{w}_j$$

$$\Delta m_{j-1} = m_{j-1} - \tilde{m}_{j-1}$$

Substituting these expressions into Eq. (4.9) and subtracting Eq. (2.20), it can be seen that the flow adjustments must satisfy the expression

$$\Delta m_j - \Delta m_{j-1} + \Delta X_j \sum_{k \in I} e_{ik} \Delta w_j = -E_j \quad (4.10)$$

Assuming that  $m$  and  $w$  are functions of pressure only, the flow corrections can be expressed as

$$\Delta m_j = \frac{\partial \tilde{m}_j}{\partial P_{j-1}} \delta P_{j-1} + \frac{\partial \tilde{m}_j}{\partial P_j} \delta P_j$$

$$\Delta w_j = \frac{\partial \tilde{w}_j}{\partial P_{ij-1}} \delta P_{ij-1} + \frac{\partial \tilde{w}_j}{\partial P_{nj-1}} \delta P_{nj-1}$$

$$\Delta m_{j-1} = \frac{\partial \tilde{m}_{j-1}}{\partial P_{j-1}} \delta P_{j-1} + \frac{\partial \tilde{m}_{j-1}}{\partial P_{j-2}} \delta P_{j-2}$$

The subscript  $n$  on the pressure terms in the equation for  $\Delta w_j$  above refers to the channel connected to channel  $I$  through gap  $k$ . The expressions for the flow and crossflow derivatives are derived from the linearized form of the axial and transverse momentum equations (Eq. [4.1] and [4.5]) and are presented in Table 4.1.

The term  $A_{ij}$  in the definitions in Table 4.1 is from Eq. (4.2), and the  $C_j$  term stands for the denominator of the right-hand side of Eq. (4.5).

**Table 4.1.** Flow Derivatives with Respect to Pressure

Flow Derivative	Definition
$\frac{\partial \tilde{m}_j}{\partial P_{j-1}} = - \frac{\partial \tilde{m}_j}{\partial P_j}$	$\frac{\bar{g}_c \bar{A}_j}{A_{ij}}$
$\frac{\partial \tilde{w}_j}{\partial P_{ij-1}} = - \frac{\partial \tilde{w}_j}{\partial P_{jj-1}}$	$\left[ \frac{S}{\ell} \right] \frac{\Delta X_j \bar{g}_c}{C_j}$

After substituting these definitions for the flow derivatives into the above equations for the flow corrections and rearranging, the general form for Eq. (4.10) becomes

$$\begin{aligned}
 & -\frac{\partial \tilde{m}_{j-1}}{\partial P_{j-2}} \delta P_{j-1} + \left[ \frac{\partial \tilde{m}_j}{\partial P_{j-1}} - \frac{\partial \tilde{m}_{j-1}}{\partial P_{j-1}} + \Delta X_j \sum_{k \in i} e_{ik} \frac{\partial \tilde{w}_j}{\partial P_{ij-1}} \right] \delta P_{j-1} \\
 & + \frac{\partial \tilde{m}_j}{\partial P_j} \delta P_j + \Delta X_j \sum_{k \in i} e_{ik} \frac{\partial \tilde{w}_j}{\partial P_{nj-1}} \delta P_{nj-1} = -E_j
 \end{aligned}
 \tag{4.11}$$

This is equivalent to

$$\begin{aligned}
 & \frac{\partial E_j}{\partial P_{j-2}} \delta P_{j-2} + \frac{\partial E_j}{\partial P_{j-1}} \delta P_{j-1} + \frac{\partial E_j}{\partial P_j} \delta P_j \\
 & + \sum_{k \in i} \frac{\partial E_j}{\partial P_{nj-1}} \delta P_{nj-1} = -E_j
 \end{aligned}
 \tag{4.12}$$

The partial derivatives of the continuity error  $E_j$  with respect to pressure are defined in terms of the computational variables, as shown in Table 4.2.

**Table 4.2.** Continuity Error Derivatives with Respect to Pressure

Error Derivative	Definition
$\frac{\partial E_j}{\partial P_{j-1}}$	$\frac{\partial \tilde{m}_j}{\partial P_{j-1}} - \frac{\partial \tilde{m}_{j-1}}{\partial P_{j-1}} + \Delta X_j \sum_{k \in i} e_{ik} \frac{\partial \tilde{w}_j}{\partial P_{ij-1}}$
$\frac{\partial E_j}{\partial P_{j-2}}$	$-\frac{\partial \tilde{m}_{j-1}}{\partial P_{j-2}}$
$\frac{\partial E_j}{\partial P_j}$	$\frac{\partial \tilde{m}_j}{\partial P_j}$
$\sum_{k \in i} \frac{\partial E_j}{\partial P_{nj-1}}$	$-\Delta X_j \sum_{k \in i} e_{ik} \frac{\partial \tilde{w}_j}{\partial P_{nj-1}}$

When the expression presented in Eq. (4.12) is evaluated for all axial nodes and all flow channels for a given assembly, the coefficient matrix consists of M equations, where M is the total number of cells in the assembly (M = number of axial nodes times the number of channels). The matrix equation has the form

$$\begin{bmatrix} \frac{\partial E_1}{\partial P_1} & \dots & \frac{\partial E_1}{\partial P_M} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \frac{\partial E_M}{\partial P_1} & \dots & \frac{\partial E_M}{\partial P_M} \end{bmatrix} \begin{Bmatrix} \delta P_1 \\ \cdot \\ \cdot \\ \delta P_M \end{Bmatrix} = \begin{Bmatrix} E_1 \\ \cdot \\ \cdot \\ E_M \end{Bmatrix} \quad (4.13)$$

Since it is generally not possible to solve a matrix of this size directly, the matrix is solved in pieces. An alternating direction iterative solution is implemented that performs a level-by-level sweep followed by a channel-by-channel sweep for each iteration in the pressure loop. The solution proceeds as described below.

First, the pressure change in each channel in the assembly is determined at each axial level, using current values of pressure change in adjacent axial levels. The equation for continuity error becomes

$$\begin{aligned} & \frac{\partial E_j}{\partial P_{i,j-1}} \delta P_{i,j-1} + \sum_{kei} \frac{\partial E_j}{\partial P_{nj-1}} \delta P_{nj-1} \\ & = -E_j - \frac{\partial E_j}{\partial P_{i,j-2}} \delta P_{i,j-2}^o - \frac{\partial E_j}{\partial P_{i,j}} \delta P_{i,j}^o \end{aligned} \quad (4.14)$$

Note that the most recent values of the pressure change, denoted by the superscript o, are used at the axial levels at the current level, j, and at the previous level, j-1.



For all N channels in the assembly at axial level j, Eq. (4.14) produces a system of N equations having the form

$$\begin{bmatrix} \frac{\partial E_{1,j}}{\partial P_{1,j-1}} & \dots & \frac{\partial E_{1,j}}{\partial P_{N,j-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E_{N,j}}{\partial P_{1,j-1}} & \dots & \frac{\partial E_{N,j}}{\partial P_{N,j-1}} \end{bmatrix} \begin{bmatrix} \delta P_{1,j-1} \\ \vdots \\ \delta P_{N,j-1} \end{bmatrix} = \begin{bmatrix} -E_{1,j} - \frac{\partial E_{1,j}}{\partial P_{j-2}} \delta P_{j-2}^{\circ} - \frac{\partial E_{1,j}}{\partial P_j} \delta P_j^{\circ} \\ \vdots \\ -E_{N,j} - \frac{\partial E_{N,j}}{\partial P_{j-2}} \delta P_{j-2}^{\circ} - \frac{\partial E_{N,j}}{\partial P_j} \delta P_j^{\circ} \end{bmatrix} \quad (4.15)$$

This matrix is solved by direct inversion to obtain new  $\delta P$  values for the current level j.

On the second sweep of the assembly, the pressure changes at all axial levels are determined for each channel of the assembly, using current values for the pressure changes in adjacent channels. This equation for continuity error is

$$\begin{aligned} \frac{\partial E_j}{\partial P_{j-2}} \delta P_{j-2} + \frac{\partial E_j}{\partial P_{i,j-1}} \delta P_{i,j-1} + \frac{\partial E_j}{\partial P_j} \delta P_j \\ = -E_j - \sum_{k \neq i} \frac{\partial E_j}{\partial P_{n,j-1}} \delta P_{n,j-1}^{\circ} \end{aligned} \quad (4.16)$$

Considering all J axial levels in a given channel i, Eq. (4.16) forms a tridiagonal system of J equations for channel i, of the form

$$\begin{bmatrix} \frac{\delta E_{1,2}}{\delta P_{1,1}} & \frac{\delta E_{1,2}}{\delta P_{1,2}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \frac{\partial E_{i,J+1}}{\partial P_{i,j-1}} & \frac{\partial E_{i,J+1}}{\partial P_{i,j}} \end{bmatrix} \begin{bmatrix} \delta P_{1,1} \\ \vdots \\ \delta P_{i,1} \end{bmatrix} = \begin{bmatrix} -E_{1,2} + \sum_{k \neq i} \frac{\partial E_{1,2}}{\partial P_{n,1}} \delta P_{n,1}^{\circ} \\ \vdots \\ -E_{i,J+1} + \sum_{k \neq i} \frac{\partial E_{i,J+1}}{\partial P_{n,J}} \delta P_{n,J}^{\circ} \end{bmatrix} \quad (4.17)$$

This matrix equation is solved to obtain new  $\delta P$  values for the current channel.

The pressure solution is a potentially time-consuming process for large problems. To solve the equation more efficiently, an axial rebalancing step is performed every third iteration, which consists of a one-dimensional axial approximation of the problem. To form the one-dimensional tridiagonal matrix equation, the total continuity error at each level  $j$  is calculated by summing over all  $N$  channels, as

$$\bar{E}_j = \sum_{i=1}^N E_{ij}$$

A tridiagonal matrix is then formed as follows:

$$\left[ \frac{\partial \bar{E}}{\partial \bar{P}_j} \right] \{\delta \bar{P}\} = \bar{E}$$

This equation is then solved to obtain an average pressure correction at each level, which is then applied to the individual channel pressures at that level. During the pressure loop, the computational mesh is repeatedly swept, level-by-level and channel-by-channel, solving for  $\delta P$ , until the change in all of the  $\delta P$  values falls below a specified convergence criteria.

After the solution to the pressure equation has converged, the resulting pressure changes, the  $\delta P$  values, represent adjustments to the pressure field that make the flow field satisfy continuity. The pressures and flows are then updated using

$$P_{i,j} = P_{i,j}^o + \delta P_{i,j}$$

$$m_j = \tilde{m}_j + \frac{\partial \tilde{m}_j}{\partial P_{i,j-1}} (\delta P_{i,j-1} - \delta P_{i,j})$$

$$w_j = \tilde{w}_j + \frac{\partial \tilde{w}_j}{\partial P_{i,j-1}} (\delta P_{i,j-1} - \delta P_{i,j-1})$$

At this point, the flow solution satisfies both momentum and continuity for the current set of properties and boundary conditions for the given assembly. The procedure is repeated for each assembly in the problem domain. This new flow field is then used in the energy solution to determine the convective heat transfer to the fluid, as described in detail in Section 4.2.

## 4.2 Energy Solution

The energy solution for the fluid and solid structures has been divided into the following steps:

1. The coefficient matrices representing the individual rod equations are decomposed by lower-upper (LU) triangular factorization to determine  $T_{\text{clad}}$  as a function of  $T_{\text{fluid}}$ .
2. The fluid and slab equations are solved simultaneously using successive over-relaxation to obtain a fluid enthalpy and slab temperature distribution.
3. The rod temperature distribution is determined by back substitution of the new fluid temperatures.

The simultaneous solution of the fluid and slab energy equations is described in detail in Section 4.2.1. The rod energy solution, which consists of steps 1 and 3 above, is described in Section 4.2.2.

### 4.2.1 Fluid and Slab Energy Solution

The fluid energy equation (Eq. [2.24]) and the slab energy equation (Eq. [2.31]) are generally so tightly coupled in COBRA-SFS applications that they must be solved simultaneously. The construction of the matrix equation describing each fluid and slab node at a given axial level is described below, followed by a description of the solution of both sets of equations.

The fluid energy equation (Eq. [2.24]) is written in terms of fluid enthalpy. However, certain terms require the fluid temperature to calculate the heat transfer due to fluid conduction and convection. Therefore, the fluid temperatures are converted to enthalpies by the following approximation:

$$T = T_o + \frac{h - h_o}{c_p}$$

In this definition, the reference temperature and enthalpy are chosen arbitrarily as the current values of  $T$  and  $h$  from the previous iteration. The cladding temperature  $T_{\text{clad}}$  can be represented by solving Eq. (2.26) as

$$T_{\text{clad}} = K_1 + K_2 + K_3 H_{\text{surf}} T_{\text{fluid}} \quad (4.18)$$

When these two expressions are substituted into the fluid energy equation (Eq. [2.24]), an equation is obtained that is linear in enthalpy. This system of  $N$  equations, (one for each subchannel  $i$  at level  $j$ ), forms the matrix equation,

$$[L]\{h\} = \{S_j\} \quad (4.19)$$

The matrix  $[L]$  is an  $N \times N$  coefficient matrix,  $N$  being the number of channels. The source vector is  $\{S_h\}$ , and  $\{h\}_j$  consists of the fluid enthalpies at level  $j$ . For large problems, the matrix  $[L]$  is sparse, and in the interests of computational efficiency, only the non-zero elements are stored.

The only temperature terms in the fluid energy equation that are not expressed in terms of enthalpy are the slab temperatures  $T_w$ . These temperatures are obtained from the solution of the solid structure energy equation and are included in the source vector  $\{S_h\}$ . After each energy iteration, the terms containing slab temperatures are updated to reflect the values for the current iteration.

The slab energy equation presented in Eq. (2.31) is written in terms of temperature. However, the terms describing radiative heat transfer are a function of temperature to the fourth power. Rather than treat these terms explicitly, they are written for slab node  $i$  as

$$\begin{aligned} Q_R &= A_{WF} \sigma \left[ \sum_{m \in i} F_{im} (T_w^4 - T_{w_m}^4) + \sum_{n \in i} F_{in} (T_w^4 - T_{c_n}^4) \right] \\ &= A_{WF} \sigma \left[ T_w^4 \left( \sum_{m \in i} F_{im} + \sum_{n \in i} F_{in} \right) - \left( \sum_{m \in i} F_{im} T_{w_m}^4 + \sum_{n \in i} F_{in} T_{c_n}^4 \right) \right] \\ &= A_{WF} \sigma H_{RAD} (T_w - \bar{T}) \end{aligned} \quad (4.20)$$

where  $H_{RAD} = \epsilon_w T_w^3$

$$\bar{T} = \left( \sum_{m \in i} F_{im} T_{w_m}^4 + \sum_{n \in i} F_{in} T_{c_n}^4 \right) / \epsilon_w T_w^3$$

$$\epsilon_w = \left( \sum_{m \in i} F_{im} + \sum_{n \in i} F_{in} \right)$$

The terms  $H_{RAD}$  and  $\bar{T}$  are evaluated using current iterate values and the resulting linear expression is substituted into the slab energy equation (Eq. [2.31]). This yields an equation that is linear in temperature at level  $j$ . The equations for every slab at level  $j$  are combined to form the matrix equation

$$[U] \{T_w\}_j = \{S_w\}_j$$

The matrix  $[U]$  is an  $M \times M$  coefficient matrix, where  $M$  is the number of slabs  $\{S_w\}$  is the source vector, and  $\{T_w\}_j$  contains the slab temperatures at level  $j$ . For large problems, the matrix  $[U]$  is sparse, and in the interest of computational efficiency, only the nonzero elements are stored.

The slab energy equation consists of terms containing not only slab temperatures, but also fluid temperatures,  $T_i$ , and composite radiation temperatures,  $\bar{T}$ . Fluid temperatures are obtained from the fluid energy equation and are represented in the source vector  $\{S_w\}$ . After each fluid energy iteration, the source terms containing fluid temperatures are updated to reflect the current iteration values. The

composite radiation temperature  $\bar{T}$  is derived as a summation of terms containing slab and cladding temperatures to the fourth power. The cladding portion of this composite temperature is not updated during the fluid and slab solution, but the slab portion is updated after every slab energy iteration.

Both the fluid and slab matrix equations are solved by successive overrelaxation, with a default relaxation factor of 1.2. A single iteration consists of one sweep through the fluid equations, followed by one sweep through the slab equations. The iteration is repeated until the changes in fluid enthalpy and slab temperature are both less than user-specified convergence criteria. Alternatively, convergence may be defined as obtaining an asymptotic limit for the temperature field, but this requires some insight on the part of the user in examining the overall results calculated for the problem.

Once the fluid and slab energy solutions have both converged, the fluid temperatures are obtained using the fluid equation of state. The fluid temperatures are substituted into Eq. (4.18) to obtain the rod cladding temperatures. If a detailed rod model is used, the fuel temperatures are also back-calculated, using Eq. (4.21), as described in the solution of the fuel rod energy equation in Section 4.2.2.

For problems with several nodes and significant heat transfer in the radial direction, the energy solution may take hundreds of iterations to converge. To speed up the convergence rate, energy rebalancing is performed on both the fluid and slab matrix equations.

#### 4.2.2 Fuel Rod Energy Solution

To expedite the energy solution, the rod surface temperature is described as a function of fluid temperature in the form

$$T_{\text{clad}} = K_1 + K_2 + K_3 H_{\text{surf}} T_{\text{fluid}}$$

The constants  $K_1$ ,  $K_2$ , and  $K_3$  are generated by performing an LU decomposition on the fuel rod finite-volume equations. If radiative heat transfer is not considered, the constants are generated only once. If it is considered, the cladding temperature of a particular rod is dependent on the temperatures of other rods, and the rod equations are solved iteratively using the fluid and slab temperature distribution from the previous fluid temperature iteration. When the constants  $K_1$ ,  $K_2$ , and  $K_3$  have been determined, the above expression for cladding temperature is substituted into the fluid energy equation (Eq. [2.24]) before solving the fluid and slab energy equations.

In order to perform the decomposition, the fuel rod finite-volume equations (Eqs. [2.27] through [2.30]) are cast into a tridiagonal matrix of the form shown in Eq. (4.21). Here a rod model with one cladding node and four fuel nodes is used in the following example:

$$\begin{bmatrix} A_{12} & A_{13} & 0 & 0 & 0 \\ A_{21} & A_{22} & A_{23} & 0 & 0 \\ 0 & A_{31} & A_{32} & A_{33} & 0 \\ 0 & 0 & A_{41} & A_{42} & A_{43} \\ 0 & 0 & 0 & A_{51} & A_{52} \end{bmatrix} \begin{Bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_{\text{clad}} \end{Bmatrix} = \begin{Bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 + f(T_f) \end{Bmatrix} \quad (4.21)$$

The elements are ordered in a manner that allows storage of the coefficients in a 3 x 5 array. The subscripts in Eq. (4-21) are the index values of the storage array in COBRA-SFS rather than the matrix indices.

The terms in the coefficient matrix **A** are defined in Table 4.3 and the terms in the source vector  $\gamma$  are defined in Table 4.4. If the coolant temperature distribution  $T_f$  were known, Eq. (4.21) could be solved directly for the rod temperature distribution. However, since  $T_f$  is unknown, matrix **A** is factored into an upper and lower triangular matrix, **B** and **C**, respectively, as shown in Eq. (4.22). As in Eq. (4.21), the subscripts in Eq. (4.22) represent storage locations rather than matrix indices.

$$\bar{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ C_{21} & 1 & 0 & 0 & 0 \\ 0 & C_{31} & 1 & 0 & 0 \\ 0 & 0 & C_{41} & 1 & 0 \\ 0 & 0 & 0 & C_{51} & 1 \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & 0 & 0 & 0 \\ 0 & B_{21} & B_{22} & 0 & 0 \\ 0 & 0 & B_{31} & B_{32} & 0 \\ 0 & 0 & 0 & B_{41} & B_{42} \\ 0 & 0 & 0 & 0 & B_{51} \end{bmatrix} \quad (4.22)$$

Performing the indicated matrix multiplication gives

$$\overline{CB} = \begin{bmatrix} B_{11} & B_{12} & 0 & 0 & 0 \\ C_{21}B_{11} & C_{21}B_{12} + B_{21} & B_{22} & 0 & 0 \\ 0 & C_{31}B_{21} & C_{31}B_{22} + B_{31} & B_{32} & 0 \\ 0 & 0 & C_{41}B_{31} & C_{41}B_{32} + B_{41} & B_{42} \\ 0 & 0 & 0 & C_{51}B_{41} & C_{51}B_{42} + B_{51} \end{bmatrix} \quad (4.23)$$

By inspection, the terms of the stored **A** matrix can be defined in terms of the **B** and **C** matrices as

$$\begin{array}{lll} A_{12} = B_{11} & A_{13} = B_{12} \\ A_{21} = C_{21}B_{11} & A_{22} = C_{21}B_{12} + B_{21} & A_{23} = B_{22} \\ A_{31} = C_{31}B_{21} & A_{32} = C_{31}B_{22} + B_{31} & A_{33} = B_{32} \\ A_{41} = C_{41}B_{31} & A_{42} = C_{41}B_{32} + B_{41} & A_{43} = B_{42} \\ A_{51} = C_{51}B_{41} & A_{52} = C_{51}B_{42} + B_{51} & A_{53} = 0 \end{array}$$

Table 4.3. Definitions of Coefficients in the A Matrix

Element	Definition
$A_{12}$	$\frac{(\rho c_p)_f}{\Delta t} \left( \frac{d_2^2 - d_{\text{void}}^2}{4N_\theta} \right) + \frac{d_2}{\Delta r N_\theta} k_f + \frac{4\Delta r N_\theta k_f}{\pi^2(d_1 + d_2)}$
$A_{13}$	$- \frac{d_2}{\Delta r N_\theta} k_f$
$A_{21}$ and $A_{31}$	$- \frac{d_i}{\Delta r N_\theta} k_f$
$A_{22}$ and $A_{32}$	$\frac{(\rho c_p)_f}{\Delta t} \left( \frac{d_{i+1}^2 - d_i^2}{4N_\theta} \right) + \frac{k_c}{\Delta r N_\theta} (d_i + d_{i+1}) + \frac{4\Delta r N_\theta k_f}{\pi^2(d_i + d_{i+1})}$
$A_{23}$ and $A_{33}$	$- \frac{d_{i+1}}{\Delta r N_\theta} k_f$
$A_{41}$	$- \frac{d_N}{\Delta r N_\theta} k_f$
$A_{42}$	$\frac{(\rho c_p)_f}{\Delta t} \left( \frac{d_{\text{fuel}}^2 - d_N^2}{4N_\theta} \right) + \frac{d_N}{\Delta r N_\theta} k_f + \frac{d_{\text{fuel}} H_g}{N_\theta} + \frac{4\Delta r N_\theta k_f}{\pi^2(d_{\text{fuel}} + d_n)}$
$A_{43}$ and $A_{51}$	$- \frac{d_{\text{fuel}} H_g}{N_\theta}$
$A_{52}$	$\frac{(\rho c_p)_{\text{clad}}}{\Delta t} \left[ \frac{d_{\text{clad}}^2 - (d_{\text{clad}} - 2t_{\text{clad}})^2}{N_\theta} \right] + \frac{d_{\text{fuel}} H_g}{N_\theta} + \frac{d_{\text{clad}} H_{\text{surf}}}{N_\theta} + \frac{2\Delta r N_\theta k_c}{\pi^2(d_{\text{clad}} - t_{\text{clad}})}$

Using elementary matrix algebra, an algebraic expression relating  $T_{\text{clad}}$  and  $T_{\text{fluid}}$  can be determined, as

$$\begin{aligned} \text{let } \bar{A}T &= Y \\ \text{and } Y &= \bar{C}Z \\ \text{then } \bar{C}BT &= Y \\ \bar{C}BT &= \bar{C}Z \\ \text{thus } \bar{B}T &= Z \end{aligned}$$

Table 4.4. Definitions of Terms in the Y Matrix

Element	Definition
$Y_1$	$\left[ \frac{(\rho c_p)_f T_1^n}{\Delta t} + q_f''' \right] \left( \frac{d_1^2 - d_{\text{void}}^2}{4N_\theta} \right) + \frac{d_{\text{void}}}{N_\theta} q_{\text{void}}''$ $+ \frac{2\Delta r N_\theta k_f}{\pi^2(d_1 + d_2)} (T_{1,m-1} + T_{1,m+1})$
$Y_2, Y_3$ and $Y_4$	$\left[ \frac{(\rho c_p)_f T_i^n}{\Delta t} + q_f''' \right] \left( \frac{d_i^2 - d_{i-1}^2}{4N_\theta} \right)$ $+ \frac{2\Delta r N_\theta k_f}{\pi^2(d_i + d_{i-1})} (T_{n,m-1} + T_{n,m+1})$
$Y_5$	$\left[ \frac{(\rho c_p)_f T_c^n}{\Delta t} \right] \left[ \frac{d_{\text{clad}}^2 - (d_{\text{clad}} - 2t_{\text{clad}})^2}{4N_\theta} \right]^2$ $+ \frac{\Delta r N_\theta k_c}{\pi^2(d_{\text{clad}} - t_{\text{clad}})} (T_{c,m+1} + T_{c,m-1})$
$f(T_f)$	$(d_5 H_{\text{surf}} T_{\text{fluid}})$

Since **B** is upper triangular, only **Z** must be found to solve for **T**. The elements of **B** and **C** are given as follows:

$$\begin{aligned}
 B_{11} &= A_{12} & C_{21} &= A_{21}/B_{11} \\
 B_{12} &= A_{13} & B_{21} &= A_{21}/C_{21} \\
 C_{31} &= A_{31}/B_{21} & B_{22} &= A_{23} \\
 B_{31} &= A_{32} - C_{31} B_{22} & C_{41} &= A_{41}/B_{31} \\
 & & B_{41} &= A_{42} - C_{41} B_{32}
 \end{aligned}$$



Since  $\gamma = CZ$ ,  $Z$  can be evaluated in terms of  $C$  and  $Y$  as

$$Z_1 = Y_1$$

$$Z_2 = Y_2 - Z_1 C_{21}$$

$$Z_3 = Y_3 - Z_2 C_{31}$$

$$Z_4 = Y_4 - Z_3 C_{41}$$

$$Z_5 = Y_5 - Z_4 C_{51} + f(T_p)$$

Since  $BT = Z$ , the expression for the clad temperature is determined as

$$B_{51} T_5 = Z_5$$

$$T_5 = Z_5/B_{51}$$

$$T_5 = \frac{Y_5}{B_{51}} + \frac{Z_4 C_{51}}{B_{51}} + \frac{d_5 H_{\text{surf}}}{B_{51}} T_{\text{fluid}}$$

In more general terms of  $N$  fuel nodes, this is

$$T_{N+1} = \frac{Y_{N+1}}{B_{N+1}} + \frac{Z_N C_{N+1}}{B_{N+1}} + \frac{d_{\text{rod}} H_{\text{surf}}}{B_{N+1}} T_{\text{fluid}}$$

This is equivalent to the cladding temperature expression

$$T_{\text{clad}} = K_1 + K_2 + K_3 H_{\text{surf}} T_{\text{fluid}}$$

This decomposition allows the cladding and fuel temperatures to be expressed as functions of fluid temperature. If there is no heat transfer by thermal radiation or by circumferential conduction, and if there is no rod-to-rod contact, then the rod solution is complete. However, if any of these effects are present, a change in temperature results in a change in terms of the source matrix  $Y$ , which requires iteration on the rod temperatures. When the constants  $K_1$ ,  $K_2$ , and  $K_3$  have been determined, the expression for cladding temperature is substituted into the fluid energy equation before solving the fluid and slab energy equations.

## 5.0 Boundary Conditions

The COBRA-SFS code has great flexibility in the manner in which boundary conditions may be applied to a given problem. This reflects the wide range of possible applications of the code, and the complex geometries and heat transfer systems that it can be used to analyze. This flexibility, although necessary and desirable, places something of a burden on the user, however. The user must make logical and consistent selections in the boundary condition input, or risk falling prey to the merciless "gigo" law of computer programming; that is, "garbage in; garbage out." This section presents an overview of the boundary condition specification options in the code, and explains how they can be used to model different sorts of problems. It is highly recommended that the user study this section carefully before attempting to set up any problems for COBRA-SFS analysis.

The boundary conditions for the code fall rather neatly into two classes: flow boundary conditions and thermal boundary conditions. The distinction between the two types becomes a bit blurred, however, when the optional plenum model is used. The plenum regions (top or bottom or both) provide a boundary on the flow, and also on the heat transfer to the environment beyond the boundary of the system as represented in the COBRA-SFS input model. The following subsections present these two types of boundary conditions separately, and also discuss in some detail how they interact and how they can be used in conjunction with one another: Section 5.1 discusses the fluid flow boundary condition options. Section 5.2 presents the thermal boundary condition options, including the optional plenum model. Transient forcing functions, by means of which the various boundary conditions can be varied over time, are described in Section 5.3.

### 5.1 Fluid Flow Boundary Conditions

COBRA-SFS derives from the COBRA family of subchannel codes, and in these codes, boundary condition specification is a relatively straightforward affair. They were designed to model flow in a fuel assembly or reactor core. Their main purpose is to determine what enters at the bottom and exits at the top of the core. The side boundary is assumed impermeable, isothermal, and generally ignorable. The boundary conditions are specified with a uniform exit pressure seen by all subchannels, and either a specified inlet flow (in which case the code calculates the resulting inlet pressure), or a specified inlet pressure (in which case the code determines the required inlet flow). This happy state of affairs can still be modeled using COBRA-SFS, but in general it is not adequate for typical applications of the code, even neglecting for the moment the added complication of the top and bottom plenum model option.

COBRA-SFS still operates under the assumption that the channels all see a uniform exit pressure. The user has considerable flexibility, however, in specifying what happens at the inlet. Basically, there are two options, which can be summarized as follows:

1. The total inlet mass flow is specified, and the code calculates the resulting inlet pressure distribution.

2. The inlet pressure is specified (by specifying the exit pressure and the desired pressure drop), and the code calculates the resulting inlet flow.

The first option is basically the same option as in the original COBRA code, but as implemented in COBRA-SFS, the user has considerable flexibility in specifying the distribution of the inlet flow. These variations are discussed in Section 5.1.1. The variations on the specified pressure drop option are discussed in Section 5.1.2.

### 5.1.1 Inlet Mass Flow Rate Boundary Condition Options

The user has the option of specifying different boundary conditions for different assemblies. Whether a given assembly will be treated with an inlet flow or a pressure drop boundary condition is determined by user input for each assembly (with input variable ITDPA(NASS) on CHAN.6 (see Section 6.2.3). When specifying the inlet mass flow rate as a boundary condition for a given assembly, the user enters the global average inlet mass flux (in  $\text{Mlbm/hr-ft}^2$ , input variable GIN on line OPER.2; refer to Section 6.2.11). This must be the overall average inlet mass flux for all channels in all assemblies with flow boundary conditions, and is assumed to be the sum of all subchannel flows divided by the sum of their inlet flow areas.

In order to specify how this global average mass flux is distributed among the subchannels, however, the user has the following four different options:

1. GIN is the average inlet mass flux, and local subchannel inlet flows are calculated based on the subchannel inlet area (specified by  $IG=0$  on OPER.1).
2. GIN is the average inlet mass flux, but the user will specify by input the relative fraction of flow for each assembly. Within each assembly, the local subchannel flow will be calculated on the basis of the subchannel areas, assuming a uniform mass flux over the assembly inlet (specified by  $IG=2$  on OPER.1).
3. GIN is the average inlet mass flux, but the user will specify by input the relative fraction of flow in each subchannel, for specified assemblies (specified by  $IG=3$  on OPER.1).
4. GIN is the average inlet mass flux, but the user will specify by input the relative fraction of flow to each assembly, and within specified assemblies will also specify the fraction of flow in each subchannel (specified by  $IG=4$  on OPER.1).

It should be noted that these options do not really represent different boundary conditions. They are simply variations on the inlet flow boundary condition, which allow the user some control over the inlet flow distribution.

Another variation on the inlet flow boundary condition option can be obtained by requiring the code to calculate the appropriate inlet flow distribution that will give a uniform mass flux in all subchannels of the assembly. This is obtained by specifying  $IDTPA(NASS)=2$ , and defining the

desired uniform mass flux in GIN, on OPER.2. The code will use the values specified for the total system pressure drop DPS to obtain a first guess at the inlet pressure, but will then solve for inlet pressure values that give the desired uniform mass flux in the channels by adding DPS to the exit pressure PREF, (DPS and PREF are specified by input on OPER.2; see Section 6.2.11).

The user also has the option of specifying the total inlet flow rate, using input variable FTOTAL on OPER.2. The code will calculate the required inlet pressure for each channel to produce a uniform pressure drop across all channels at the specified total flow rate, FTOTAL. When this option is used, all assemblies must have the same boundary condition option, for obvious reasons. It is specified by setting ITDPA(NASS)=3 on CHAN.6 for all assemblies, or setting ITDP=3 on OPER.1 (NOTE: this option has been tested only with FTOTAL=0.0; see Sections 6.2.3 and 6.2.11).

In addition to the above, there is one more flow boundary condition option that represents something of a departure from the standard assumption of a specified inlet flow and calculated exit flow. The user may specify a zero flow boundary condition at the top and bottom of the assembly (This is accomplished by setting ITDPA(NASS) to 4 on CHAN.6 for that assembly; see Section 6.2.3). This option allows modeling of a sealed enclosure in which there are no plena at top or bottom. A number of electrically heated test sections simulating spent fuel rod arrays are of this configuration. Obviously, the uniform exit pressure assumption cannot be strictly adhered to in applying this boundary condition, or the problem would be over-specified. In this option, the specified exit pressure is used as the reference pressure for the fluid properties calculations, but the actual exit pressure seen by a given channel is whatever is required to give zero flow across the boundary.

### 5.1.2 Inlet Pressure Boundary Condition Options

The options to specify an inlet pressure in COBRA-SFS are implemented in terms of specifying a pressure drop for an assembly or group of assemblies. These are the options for a uniform pressure drop and the optional network model, which are described in detail below, in Sections 5.1.2.1 and 5.1.2.2, respectively.

#### 5.1.2.1 Uniform Pressure Drop Boundary Option

The inlet pressure boundary condition option in COBRA-SFS is implemented as a specified uniform pressure drop (which is specified in variable DPS on OPER.2; see Section 6.2.11). This boundary condition can be applied to the entire problem, or to selected assemblies within the problem, by specifying input variable ITDPA(NASS) = 1 on CHAN.6. It will be applied to all assemblies if ITDP=1 on OPER.1. When this boundary condition is selected for an assembly, the inlet pressure for each subchannel is defined as

$$P_1 = \text{PREF} + \text{DPS}$$

Note that the reference pressure PREF and total pressure drop DPS are specified by input on OPER.2.

When this option is used, the inlet flow for each subchannel in the assembly is calculated by the code. The value specified in GIN is used to determine the 'first guess' at the inlet flow rate, but the final converged value will in general be different.

### 5.1.2.2 Network Model

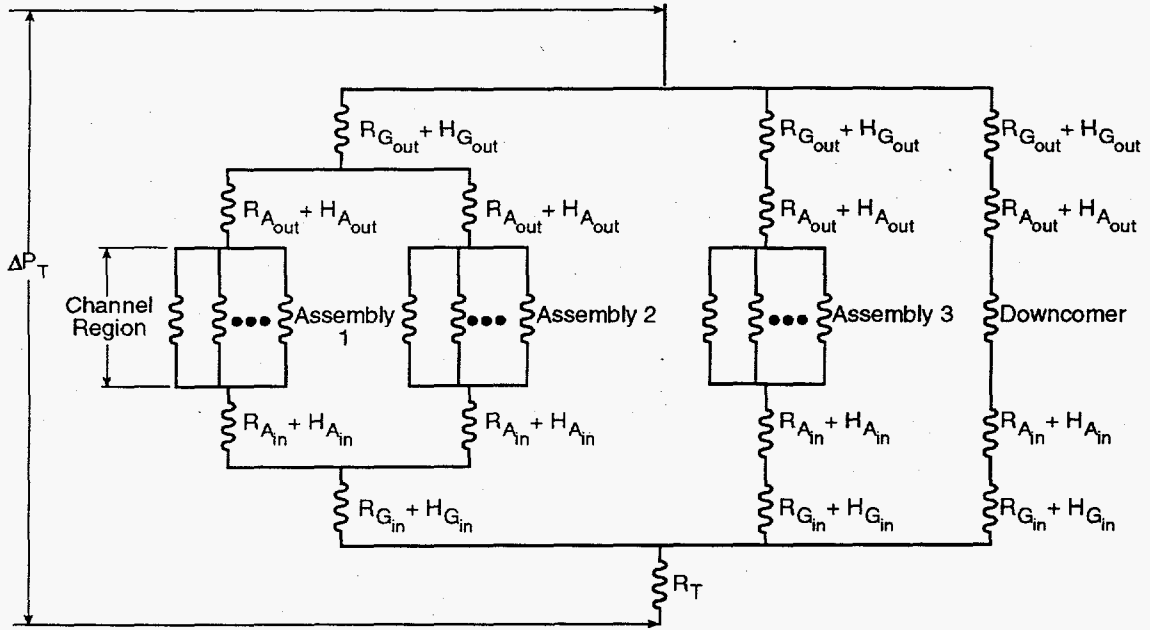
The specified total pressure drop boundary condition is generally applied uniformly to all channels. However, for some unusual modeling situations in which the COBRA code is applied to a reactor core, it may be desirable to allow the code to calculate the flow distribution between assemblies based on the orificing and assembly arrangement. The network model was developed for the COBRA code to model the pressure losses above and below the channel region in the Fast Flux Test Facility. This model has been retained in COBRA-SFS, and could be used to model a storage cask system, if it were configured as arrays of parallel assemblies with plena above and below. However, the code has not been validated for such an application.

In most cases, it would probably be more efficient to use the plenum model in COBRA-SFS, rather than the network model. If the plenum model is used, the network model is superfluous. However, the network model is available in the code, and could be used if the user wished to undertake an appropriate validation effort. When the network model is used, a single pressure drop is specified (which may vary as a function of time), and the subchannel flow rates are adjusted so that the pressure drop through each flow path matches the specified uniform pressure drop (DPS on OPER.2).

Figure 5.1 is a schematic description of the network model for a three-assembly problem with a bypass channel. The conservation equations described in the previous section are solved only for the region containing the three assemblies, as noted in Figure 5.1. This region represents the assemblies, consisting of rods and subchannels, where flow resistance is determined by wall friction factors and local form loss coefficients. The gravitational head is also accounted for in this region. Along the additional portions of the flow paths, a reduced momentum equation is solved, which takes into account only the flow resistance due to friction and form losses and the gravitational head. No inertia terms are included, and it is assumed that the transport time through the network model is essentially zero.

In the example described by Figure 5.1, the resistances marked  $R_{AIN}$  and  $R_{AOUT}$  are the flow resistances associated with the assembly inlet orifices and the outlet hardware (handling socket, etc.), respectively. The loss coefficients can be made dependent on Reynolds number by means of input option selections. A gravitational pressure drop can also be modeled by supplying head lengths at the inlet and outlet. (These are noted as  $H_A$  in Figure 5.1.) The assembly inlet gravitational head is calculated using the inlet temperature to define the density. The outlet gravitational head is calculated using the mixed mean assembly outlet temperature to determine the density.

The assembly group dynamic loss coefficients,  $R_{GIN}$  and  $R_{GOUT}$ , represent the flow resistances from a common plenum to the assembly group plena or to the downcomer region. The group dynamic loss coefficients are assumed to be independent of Reynolds number. Here too, gravitational losses may be modeled by supplying head lengths. One additional term,  $R_T$ , may be used to represent the dynamic loss coefficient for flow from the inlet nozzle to the common plenum.



S9503052.10

**Figure 5.1.** Schematic Description of the Network Model for Pressure Drop Through the Reactor Vessel

For most problems, the actual loss coefficients for each of these resistances will not be known, but the flow and corresponding pressure drop across each resistance should be available. The effective loss coefficients may then be calculated as

$$R = \frac{\Delta P g_c}{m^2}$$

For  $R$ , which may be dependent on Reynolds number, it is necessary to supply a wetted perimeter so the Reynolds number can be calculated from the flow rate. The wetted perimeter need not have any physical significance, but should be chosen so that the correct loss is obtained for a given flow rate when using the specified resistance versus Reynolds number curve.

## 5.2 Thermal Boundary Conditions

In addition to the fluid boundary conditions described above, the COBRA-SFS code also contains options for defining heat transfer boundary conditions. In many applications, there is significant heat loss radially out the sides of the canister or cask being modeled, and often out the top

and bottom of the structure, as well. The assumption of adiabatic boundaries for heat transfer is generally not adequate for spent fuel cask analysis. The user must be able to specify heat transfer boundaries on the sides, top, and bottom. The manner in which this capability is implemented in COBRA-SFS is described in Section 5.2.1 for the side boundary.

Top and bottom boundary conditions must be specified when a shipping cask or multi-assembly storage container has a lower or upper plenum within which the coolant mixes due to natural circulation. The physical structures of the cask above and below the fuel assemblies can also provide important heat transfer paths to the environment, in addition to the heat lost radially out the sides or transported by the working fluid. There also may be heat loading on the cask from the environment, due to solar insolation, external fires, etc. The top and bottom structures are modeled in COBRA-SFS using the plenum model. Boundary conditions for the plenum regions can be specified for the axial and radial directions in both plena. The expressions for heat transfer used in these boundary conditions can include conduction, thermal radiation, and natural convection, or a combination of all three. It is also possible to specify heat sources by defining local heat fluxes at the boundary. The thermal boundary conditions for the plenum regions are described in Section 5.2.2.

### 5.2.1 Side Thermal Boundary Conditions

The default in the code is to assume that there are no side thermal boundary conditions. The outside boundary at each axial level is considered as adiabatic. The user has the option, however, of specifying a heat transfer boundary on the outermost wall node at each axial level. This is done by defining a one-dimensional thermal connection, or series of connections from the node to some boundary temperature, that will permit heat transfer radially from the node. The user specifies a boundary temperature for the connection, along with a set of empirical coefficients for a heat flux correlation of the form

$$q'' = C_1 [C_2 (T_i - T_{i+1})]^{C_3} (T_i - T_{i+1}) + \sigma \frac{T_i^4 - T_{i+1}^4}{\frac{1}{\epsilon_i} + \frac{1}{\epsilon_{i+1}} - 1.0} \quad (5.1)$$

The empirical coefficients,  $C_1$ ,  $C_2$ , and  $C_3$  are specified by input for each connection type. If the effect of radiative heat transfer is to be included, the surface emissivities must also be specified. The outermost boundary temperatures are also specified by input, and the user has the option of defining variations in the boundary temperatures, both axially and circumferentially. The expression in Eq. (5.1) allows for modeling of heat transfer modes that are not linear in temperature (such as natural convection, which is expressed in terms of a Grashof or Rayleigh number).

The user has a great deal of flexibility in specifying the side thermal boundary nodes, and the input has been made as general as possible to permit modeling of a wide range of cask and canister geometries. Examples of how this option can be used are illustrated by the validation calculations presented in Section 7.0. These examples include the relatively straightforward case of a single assembly with a specified temperature on the outside of the test section wall, as well as the more

complicated case of large multi-assembly storage casks with complex heat transfer paths through support baskets, shielding, insulation layers and cask shell structures. The user would be well-advised to study these examples carefully before attempting to set up the input to specify the thermal boundary conditions for a new application of the code.

Heat transfer through the specified side boundary connections is calculated as part of the energy solution. At each axial level, the temperatures in the boundary regions are estimated using slab temperatures from the previous iteration, before the rod, fluid, and slab energy equations are solved. The one-dimensional series of equations for each boundary connection are solved iteratively for a radial temperature distribution in the boundary region. After the boundary solution has converged, a linear composite resistance is constructed to represent the total connection between the outermost slab node and the specified boundary temperature. This resistance and the boundary temperature are then implemented in the solid structure energy equation (Eq. [2.31]). After all boundary resistances at an axial level have been computed, the energy equations for that level are solved. The iteration proceeds in this manner until convergence is achieved, or until the user-specified iteration limit is reached.

### 5.2.2 Plenum Boundary Conditions

In an ordinary COBRA subchannel model, the axial flow can convect fluid energy out the top of the assembly, but it is assumed that there is no axial transport by conduction from slab and rod nodes at the top boundary. If flow reverses at the exit, it is assumed that the incoming fluid is drawn from an essentially infinite reservoir above the channel exit, at an enthalpy and pressure defined by input. Similar assumptions are made for the inlet conditions. There is no conduction heat loss out the bottom, and the inlet flow enthalpy is defined by input. But in many applications the storage canister or cask is in some way sealed at the top and bottom. It may have an open plenum at the top and bottom, where the flow streams from different channels or assemblies can communicate with each other, providing flow paths for natural circulation. In addition, there may be structures that can conduct heat from the flow channels and internal structure nodes through the end caps to the surrounding environment.

In COBRA-SFS, these top and bottom structures can be represented using the plenum model. The user can model both the top and bottom plenum, or the bottom plenum only, or the top plenum only. When both top and bottom plenum are modeled, the channel inlet conditions and outlet flow are calculated in the code as required to satisfy overall mass conservation in the system and any specified assembly pressure drop boundary conditions. When only the bottom plenum is modeled, with no top plenum, the channel outlet conditions are determined from the flow calculated for the uniform exit pressure specified in the input. When only the top plenum is modeled, without a bottom plenum, the channel inlet flow is specified according to the selected boundary condition option, as discussed in Section 5.1. The outlet conditions are calculated to satisfy mass continuity at the exit, and for heat conduction through the end cap to the ambient boundary temperature.

Specific examples of COBRA-SFS models of shipping/storage casks are presented in Section 7.0 as part of the validation of Cycle 2. In order to properly represent the heat transfer paths through the plena in these cask designs, the models are quite detailed, but the basic idea is relatively simple. This is illustrated by the 'typical' cask model shown in Figure 5.2. This model consists of an upper plenum, a channel region, and a lower plenum, with natural circulation paths established through the



channel region between the plena. The plenum model represents the fluid with a single node. It is assumed that the flow entering the plenum mixes instantaneously and completely. The resulting pressure and mixed temperature provides the fluid transport properties for flow from the plenum into the channels, as required to complete the natural circulation loop shown in Figure 5.3.

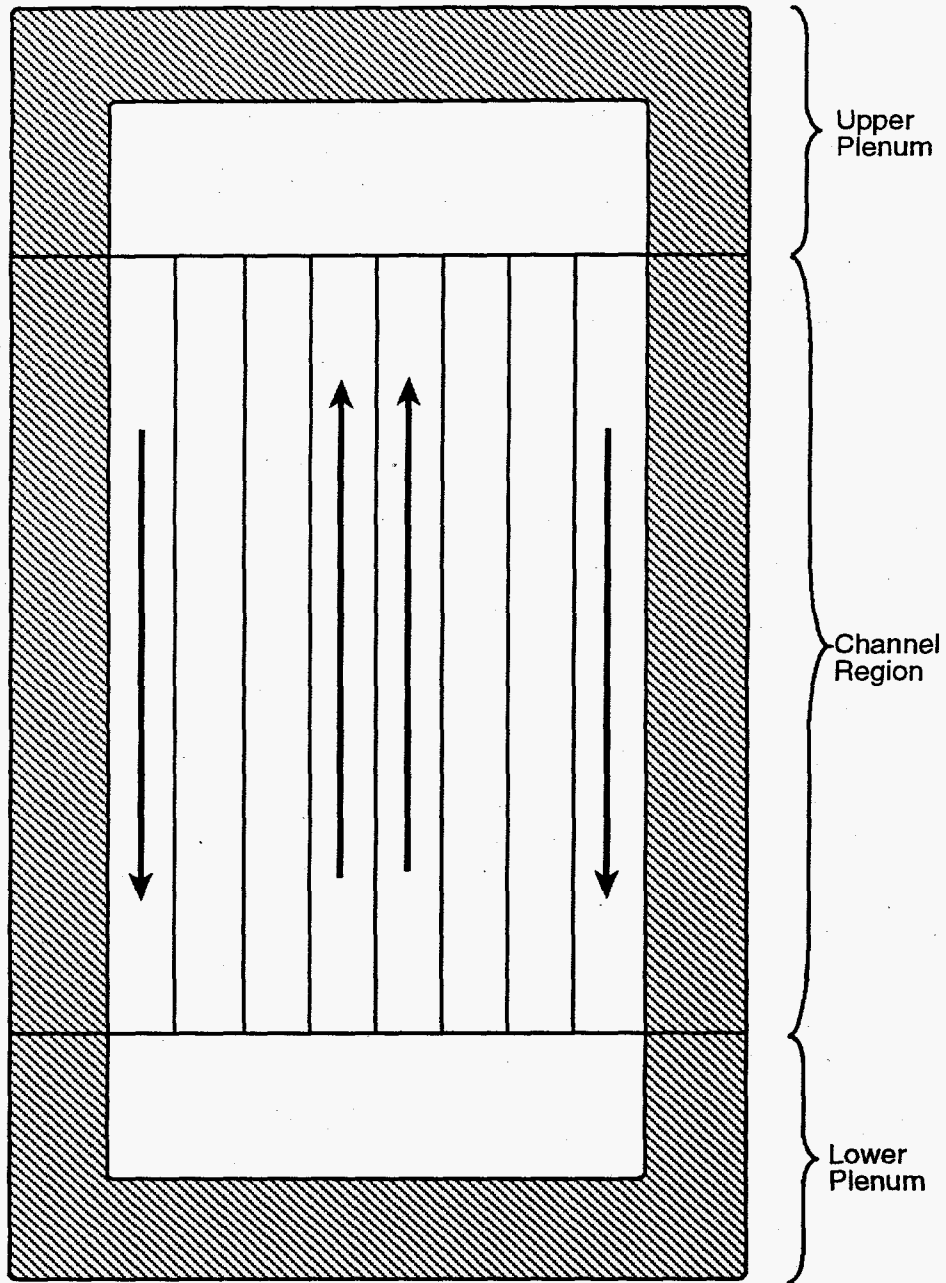
The plenum model assumes a one-dimensional node for the fluid, but the conduction heat transfer paths in the plenum regions can be radially and axially nonuniform. It is possible to include heat losses through the sides of the plenum region, as well as through the top or bottom. Fluid enters the top plenum as positive flow at the channel exit, and enters the lower plenum as negative flow at the channel inlet. Because the flow is assumed to mix completely and instantaneously in each plenum, positive flow into the channels from the lower plenum convects the mixed enthalpy into the channels, and negative flow into the channels from the upper plenum convects the mixed enthalpy of that plenum. The plenum mixture temperature is also the temperature that is used to determine the rate of heat loss to the environment through the top and sides of the plenum region, through the axial and radial conduction regions specified by user input.

The conduction heat transfer from the fluid in each plenum is modeled using one-dimensional thermal connections in the axial and radial directions. These are defined by input using the relation in Eq. (5.1), and may consist of a series of regions, with each region having a different expression for heat transfer (see the input instructions for group BDRY, in Section 6.2.10). Different boundary temperatures may also be specified for the radial and axial directions. In addition, conduction heat transfer paths through each plenum region can be modeled with thermal connections to solid structure nodes of the channel region in the axial level adjacent to the plenum. These connections represent axial heat transfer from the cask body, or heat transfer from the basket to the bottom or top of the cask. Radial and axial conduction paths through the plena can have different boundary temperatures, to appropriately represent the external environment seen by the cask. For example, a cask sitting on a solid surface would probably see different temperatures on the bottom and sides of the lower plenum.

As with the side boundary specification, this option for the top and bottom plenum has been designed to permit the user great flexibility in modeling these structures. Examples of plenum models are included in some of the validation cases presented in Section 7.0, and the user should study these closely before setting up the input to describe a specific application.

The fluid enthalpies and flow rates entering the plenum, and the temperatures of the solid structure nodes connected to the plenum, are computed in the energy solution for the axial level adjacent to the plenum. These values are then held fixed while the plenum equations are solved. The resulting plenum structure temperatures and plenum fluid enthalpy are then used to define the boundary conditions for the next iteration of the rod, fluid and slab energy solution.

If a detailed representation of the flow field in the plenum is required, however, this one-dimensional plenum model is not appropriate, since it represents the fluid in the plenum with a single temperature and pressure. An alternative to the plenum model is simply to extend the region modeled by the channels to include the plenum. This is illustrated in Figure 5.4, in which the one-dimensional plenum region of the example in Figure 5.3 is represented with four channels and five axial levels.



S9503052.11

Figure 5.2. Simple Illustration of a COBRA-SFS Cask Model

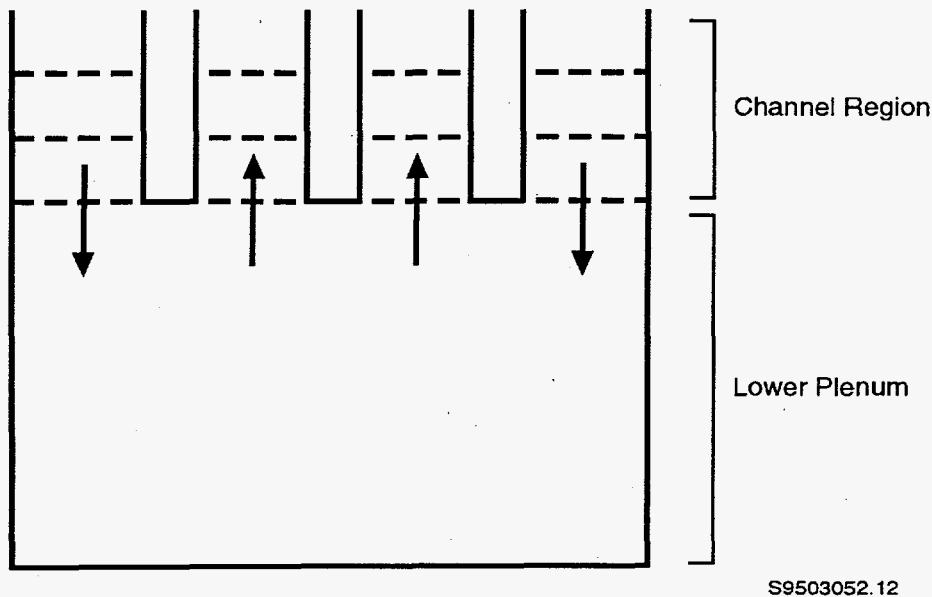
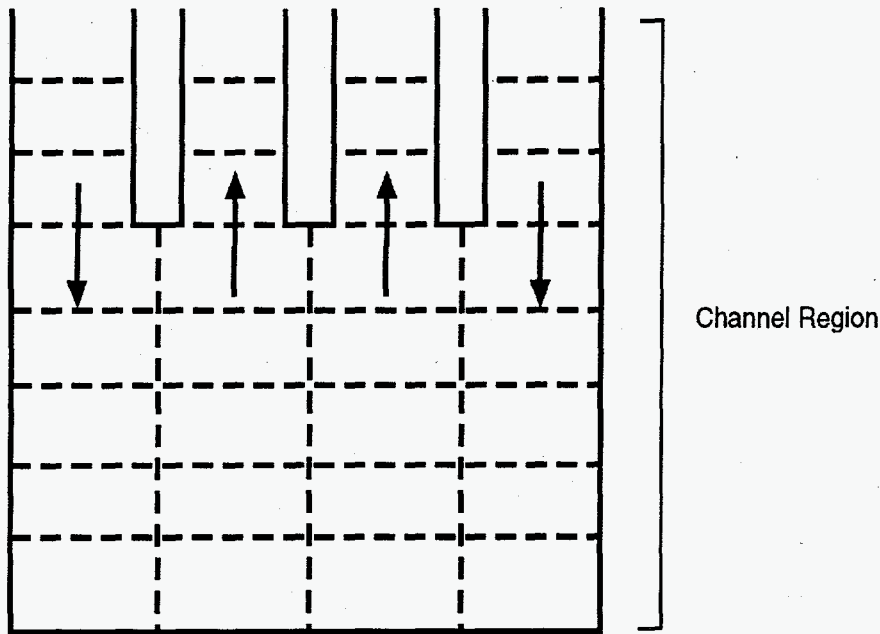


Figure 5.3. Example of COBRA-SFS Plenum Model

The necessary zero-flow boundary at each channel inlet (for the lower plenum), and at each channel outlet (for the upper plenum), is obtained with the boundary conditions specification in group OPER (the value of ITDP is specified as 4; see Section 6.2.11). If the plenum region is an open reservoir, the option for fluid-fluid shear can be used for the lateral flow connections between channels. This makes it possible to properly account for momentum losses in the open region of the plenum when it is modeled with several channels.

### 5.3 Transient Forcing Functions

The conservation equations in the COBRA-SFS code include the time-dependent terms, and are therefore applicable to transient analysis. The numerical solution method in the code is fully implicit in time, and has no time-step size limitation for stability. However, the constitutive models in the code for flow and heat transfer (described in Section 3.0) are based almost exclusively on steady-state data and on models that were derived assuming steady conditions. The code has not been validated for transient applications. If a user wishes to apply the code to transient analysis, it is recommended that appropriate validation be performed for the constitutive models that will be employed in the calculation, to verify their applicability to the problem.



S9503052.13

Figure 5.4. Example of Channels Used to Model a Plenum in COBRA-SFS

This caveat notwithstanding, the COBRA-SFS code can simulate transients in which the inlet enthalpy, total heat generation rate, system pressure, or inlet flow rate vary as a function of time. The transient forcing functions on these boundary conditions are implemented in the code by reading input tables of time versus relative (or actual) value for each parameter to be varied. Linear interpolation is used to obtain the current forcing function value at a time step between specified table entries. The value of parameter  $P$  at time  $t$  is computed as

$$P(t) = F(t)P(o)$$

where  $F(t)$  is the factor interpolated from the input transient table at time  $t$  and  $P(o)$  is the steady-state value of parameter  $P$ . Alternatively, the actual value of the parameter may be used in the table, so that

$$P(t) = F(t)$$

where  $F(t)$  is the value interpolated from the table at time  $t$ .

The factors entered in the forcing functions tables apply to any of the input options available for the given parameter. For example, the inlet enthalpy forcing function table applies to average inlet enthalpy, average inlet temperature, or to individual channel inlet enthalpy or temperature, depending on the input option selected.

Flow forcing functions may be used with any inlet flow option, including the uniform inlet pressure gradient option. Alternatively, a forcing function may be specified for the uniform pressure drop option. If the pressure drop is specified, a flow forcing function becomes the pressure drop forcing function and the inlet flows are computed in response to the pressure boundary condition. For obvious reasons, the forcing functions on flow cannot be applied when the total flow, zero flow, or plenum boundary condition options are used.

## 6.0 User Manual

This section is intended to serve as the basic user's manual for COBRA-SFS, Cycle 2. It contains guidance for the user concerning the important parameters to consider in developing an input model for the code, and a summary of validated applications. A listing of all documented applications of the code is also provided. This user guidance is presented in Section 6.1. The input instructions for the Cycle 2 version of the code are presented in Section 6.2. The auxiliary code RADGEN, used to generate view factor input for the radiative heat transfer model, is described in Section 6.3. This section includes the input instructions for RADGEN.

### 6.1 User Guidance on Code Applications

The COBRA code uses the subchannel modeling approach, and was originally designed for thermal-hydraulic analysis of reactor fuel bundles. Subsequent generations of the code retained this modeling philosophy, but have been tailored to specific applications. The COBRA-SFS code is designed to analyze multi-assembly spent fuel storage systems, such as shipping casks and storage canisters. The subchannel modeling approach is intrinsically flexible, and as a result the code is not tailored to a specific cask geometry. It has been applied to a wide variety of cask and canister designs, and has successfully accommodated a wide range of geometric configurations.

The flexibility of the subchannel modeling approach and the deliberate generality of the input structure in the code, however, require that the user know what he is about in setting up the input model for a given analysis. The validation test cases for Cycle 2 (see Section 7.0) serve as examples of how the code can be applied to various problems, from single assembly test sections to multiple assembly shipping/storage casks. In addition to these handy examples, Section 6.1.1 discusses cask modeling optimization, and illustrates some of the main considerations in setting up a cask model. A summary of all previously published reports on COBRA-SFS applications is given in Section 6.1.2. Guidance on how to deal with a calculation in which the code fails is presented in Section 6.1.3.

#### 6.1.1 COBRA-SFS Cask Model Optimization

A detailed study of COBRA-SFS modeling optimization has been published elsewhere (Rector and Michener 1989). This section summarizes the results and recommendations of that study, and presents parts of the discussion that might be of interest to a new user of the code. In general, the modeling flexibility and extensive capabilities of the code often tempt the inexperienced user into setting up a more complicated and detailed model than is actually required for the application. By developing a basic understanding of the modeling features in the code, the user can learn to construct a model that will give the best results for the least effort, both in setup time and computation time.

The accuracy of an analysis with a COBRA-SFS model of a spent fuel storage system is highly dependent on noding distribution and the number and locations of temperature points in the system. Although the code can handle very large problems consisting of hundreds of channels, rods, and axial levels, there is a practical upper limit on model size, generally dictated by computer memory requirements. Also, the larger the problem, the longer it will take to set up and verify the input and the

longer it will take to run; these are constraints in both time and memory costs. Experience has shown that the resolution afforded by detailed rod-and-subchannel modeling is not always necessary for an accurate representation of the flow and temperature field in a fuel assembly. Acceptable results can, in many cases, be obtained with much coarser representations of the assembly geometry (Rector et al. 1986).

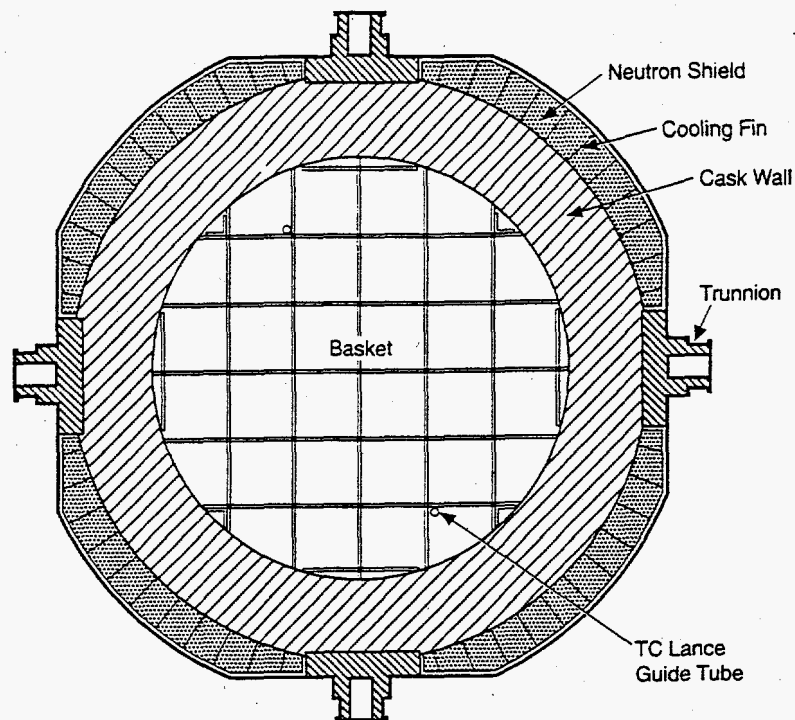
There are basically two areas in model development where the user has significant choices that affect computational efficiency. These are in the geometry modeling for assembly, channel, slab, and axial nodalization, and in the optimization of the numerical solution. The following subsections discuss the considerations and trade-offs to be made in model optimization for these two areas.

#### **6.1.1.1 Cask Model Optimization: Geometry Considerations**

The accuracy of the results depends on having enough nodes to adequately resolve the problem, but cost and memory requirements increase in direct proportion to the number of nodes. Therefore, the objective of model optimization as far as the geometry is concerned is to obtain the desired level of accuracy with the minimum number of nodes. There are basically two ways to reduce problem size: 1) reduce the actual problem to be solved by modeling only a section of symmetry rather than the whole system, and 2) select the noding such that a given node represents the largest possible region that can be characterized with a single temperature (or in the case of a fluid node, a single temperature, velocity, and pressure).

A COBRA-SFS cask model is constructed by dividing the cask into a set of finite volumes or nodes, consisting of assemblies which are made up of channels to represent the fluid flow paths and rods to represent the fuel. The channels are bounded by slabs that represent the solid material of the cask, such as assembly canisters, support baskets, shielding layers, and inner and outer structural shells. Most cask designs can be described as basically consisting of a large right circular cylinder containing an array of baskets to hold spent fuel assemblies or the fuel rods from disassembled bundles. The number of nodes required for adequate resolution of the system is highly problem dependent, but in many cases one may take advantage of inherent symmetries in the cross-section to reduce the volume of system that must be represented.

There are three conditions that must be satisfied for symmetry in the model. The geometry must be symmetric, the decay heat generation must be the same in geometrically symmetric regions, and the boundary conditions must be the same on geometrically symmetric regions. Geometric symmetry is readily verified by inspection of a cross-sectional diagram of the system. Figure 6.1 shows a diagram of the TN24P cask, which has geometric symmetry on four different axes. Examples of noding patterns that take advantage of this symmetry are shown for a half-section of the cask in Figure 6.2, and for an eighth-section in Figure 6.3. All other things being equal, the eighth-section model would give the same results as a full cask model for one-eighth the number of nodes. Alternatively, a more detailed model could be constructed for the one-eighth section using the same number of nodes as would be required for a coarser full cask model.



S9505005.5

**Figure 6.1. TN24P Cask Cross Section**

Symmetry in the distribution of the decay heat generation rates throughout the cask is usually more problematical than geometric symmetry. In general, the fuel loading will not be symmetric, but it will probably be done with an eye to making it as uniform as possible in order to minimize hot spots in the heat loading of the cask. If the distribution is relatively uniform, it can be treated as if it were symmetric, and an average heat generation rate used for assemblies in the same relative position in each of the sections of symmetry. A more conservative approach would be to use the heat generation rate of the hottest of the assemblies in the same relative position at each location in the selected section of symmetry.

In general, the external boundary conditions on a cask can be treated as symmetric, especially if the cask is simply sitting alone in a relatively benign environment. The resistance to thermal conduction around the circumference of the cask will usually be significantly less than the resistance to the ambient temperature. If local boundary conditions are imposed on the exterior of the cask, however, such as a fire or exposure to the sun on one side only, this assumed symmetry can be destroyed. Even with a uniform ambient temperature, the boundary conditions may not be symmetrical if the cask is horizontal rather than vertical. Contact with the ground or other supporting structures can significantly affect the boundary conditions, and must be considered when determining the appropriate section of symmetry to represent the cask in the COBRA-SFS model.

Once an appropriate section of symmetry has been selected for the model, the detail required for resolution of the flow field and temperatures within the assemblies and cask structure must be determined. The level of detail needed in the model depends on the type of information to be



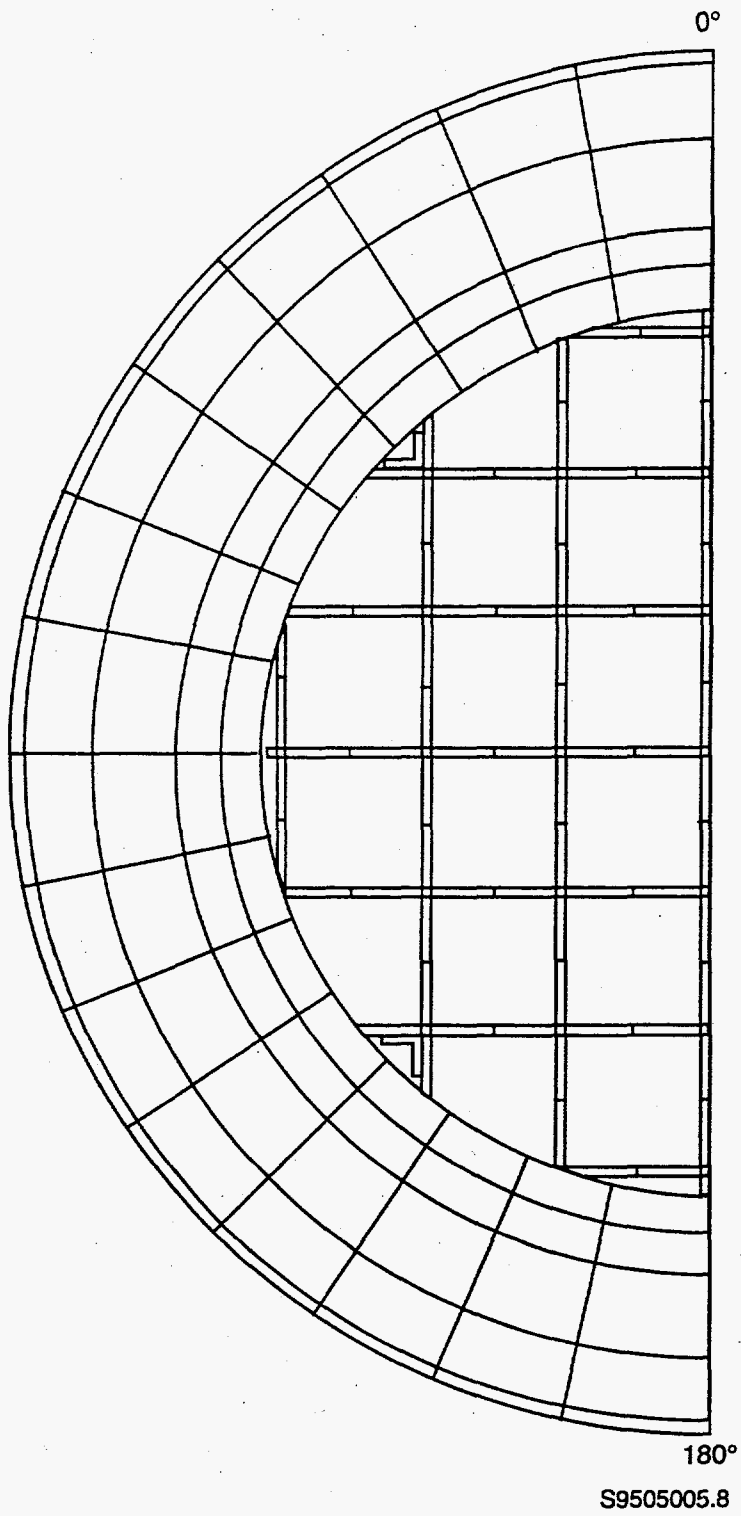


Figure 6.2. TN24P One-Half Section of Symmetry Cask Model

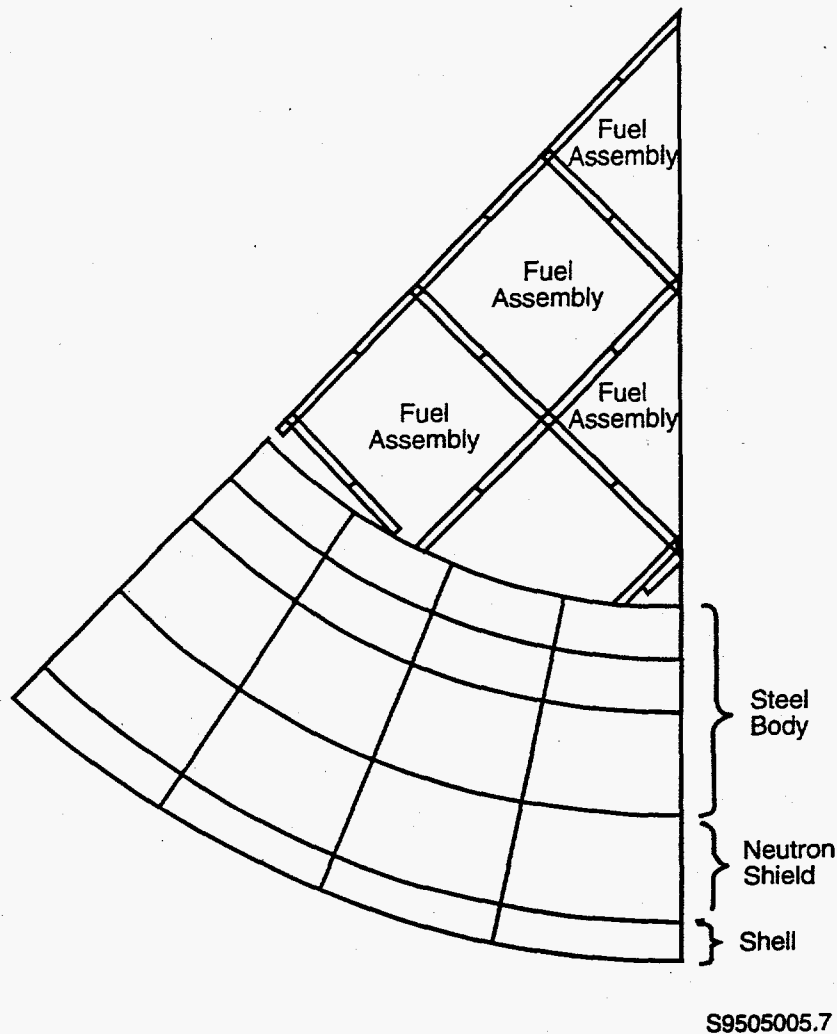
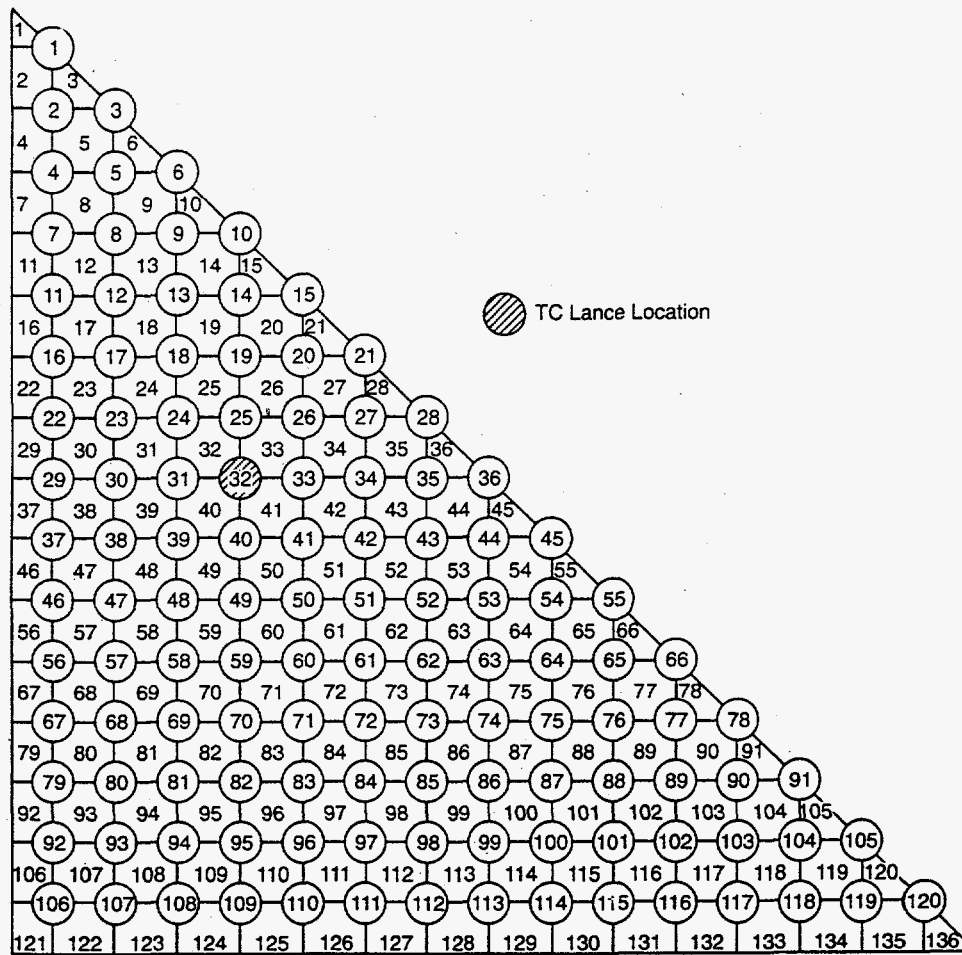


Figure 6.3. TN24P One-Eighth Section of Symmetry Cask Model

obtained from the analysis. Typically, the user wants to determine the peak clad temperature, and often the radial and axial distributions of temperature. To obtain an accurate peak clad temperature, it is usually necessary to model the hot assembly in detail, as illustrated in Figure 6.4 for a half-section of a full bundle.

For regions where the temperature is likely to be lower because of lower heat generation rates, it may be sufficient to model the assembly using lumped rods and channels, as illustrated in Figure 6.5. This sort of lumped noding may give adequate answers for peak clad temperatures even in the hot assembly, if the decay heat generation rates are relatively uniform. Sensitivity studies are advisable for very large models, to determine the minimum level of detail that will still give acceptable results. In general, so long as the assembly noding is sufficient to provide reasonably correct average channel flows and temperatures to the plena, an accurate modeling of the resistances to radial heat flow in the slab noding is more important in obtaining accurate radial temperature profiles than is excessive detail in the assembly noding.



S9503052.14

**Figure 6.4.** Rod-and-Subchannel Model for Fuel Assembly, (Half-Section of Symmetry)

The importance of accurate representation of the conduction paths requires that sufficient slab nodes be included in the model. Heat transfer between slab nodes is modeled by assuming a constant heat flux between node points, which implies a linear temperature gradient over that length, if the nodes have the same thermal conductivity. If the heat flux or temperature gradient in a region is relatively constant, then a single node-to-node connection may be sufficient, but in general the slab noding must be detailed enough to resolve any significant non-uniformity in the temperature gradient through the material. The thermal properties of the material being modeled are the best guide to determining the appropriate slab noding resolution, but in some cases sensitivity studies may be useful as well.

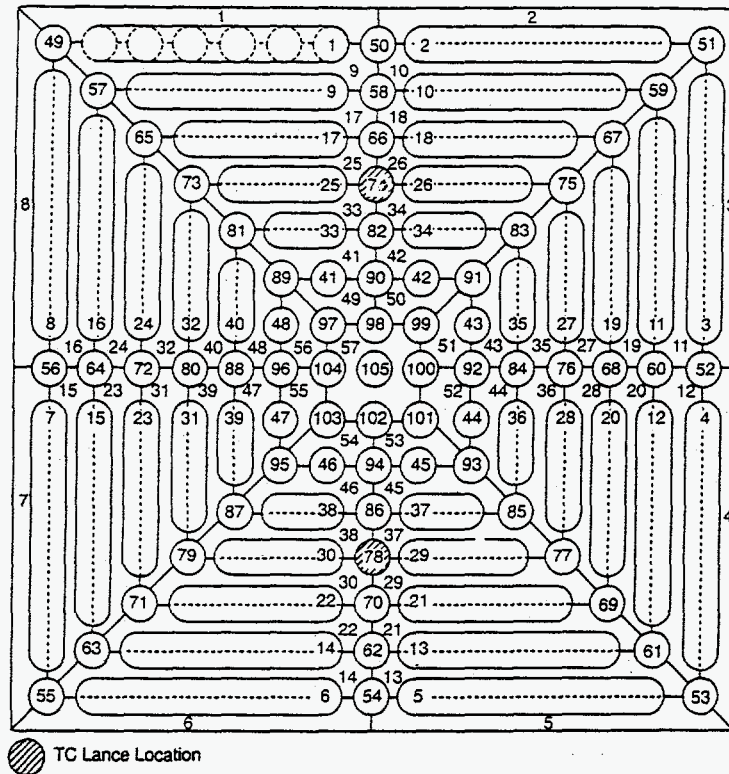


Figure 6.5. Lumped Rod and Channel Model for Fuel Assembly

### 6.1.1.2 Cask Model Optimization: Solution Parameters

The calculation time required for the numerical solution can be affected by several user-specified parameters. The most important of these are the acceleration parameters for the rod and slab energy equations and the axial rebalancing of the fluid energy solution. The COBRA-SFS numerical solution spends about 80% of the total calculation time in the energy solution, so even a small change in the rate of convergence of this part of the solution can have a significant impact on the overall cost of the simulation.

The acceleration parameters are applied to calculational variables as

$$X^n = \alpha X^n + (1.0 - \alpha)X^{n-1}$$

where

- X = calculational variable
- $\alpha$  = acceleration factor, (input)
- n = current iteration number.

The fluid and solid structure energy equations, which are solved simultaneously for all nodes at each axial level, use acceleration factors ACCELH and ACCELW (both defined on CALC.2). The default value for these two acceleration factors is 1.2, and for most problems this is a reasonable

value. However, the optimum value can be as high as 1.5 to 1.6, and may decrease the computation time by as much as 20%. For very large cask models, or if a large number of simulations will be run with a given model, it may be wise to run a few sensitivity cases to determine an optimum value for these factors.

The rod energy equation is solved for all nodes at an axial level, with the new fluid enthalpies and slab temperatures held fixed. If the rods are not thermally coupled, either by direct contact conductance or radiative heat transfer, the solution of the rod energy equation requires only one iteration, and the value of the rod solution acceleration factor is irrelevant. If the rods are coupled primarily by contact conductance, the rod energy solution may converge faster if the solution is accelerated by specifying a value for ACCROD that is greater than 1.0. If the rods are coupled by radiative heat transfer, however, it may be necessary to damp the solution because of the highly non-linear  $T^4$  terms.

Experience has shown that the value of ACCROD should probably be less than 0.8 for most problems with radiative heat transfer, and it might need to be as low as 0.5 or 0.4 for stability, in some cases. It is recommended that a conservatively low value be selected for ACCROD. The solution computation time does not seem to be readily decreased by adjustments to this parameter, but if it is too large, the solution can become unstable.

Axial rebalancing is done to accelerate the axial propagation of thermal disturbances in the flow field. It is sometimes useful in speeding up convergence, but since axial transport of energy is not generally a dominant heat transfer mechanism in cask thermal-hydraulics, the effect of such acceleration on overall convergence is often very minor. The default value of 1.0 for ACCAX is usually sufficient, and the factor has very little effect on convergence rates for most problems.

### **6.1.2 Summary of Applications of COBRA-SFS**

The basic conservation equations in COBRA-SFS are applicable to any thermal-hydraulic problem that can be adequately described using the assumptions of the subchannel modeling philosophy. The constitutive relations required to achieve closure of the equation set, however, are for the most part far less general, and usually have a limited range of applicability. As a result, the question of whether or not COBRA-SFS can be used to analyze a particular problem is one that should be approached with some caution. As a general rule, the code should not be applied without first ascertaining its suitability for the analysis by appropriate data comparisons.

The COBRA-SFS code has been extensively validated for application to single assembly analysis, by means of data comparisons with electrically heated test sections. Compared to shipping casks and storage canisters, these are relatively simple geometries, and as such provide a check on the basic solution method and the formulation of the conservation equations in the code. The code's ability to accurately predict temperature distributions for these tests shows that the conservation equations are in the proper form in the code, and that they correctly model the physical processes of heat transfer and momentum exchange. Documentation of this validation work includes the reports listed in Table 6.1.

**Table 6.1. Single-Assembly Validation of the COBRA-SFS Code**

ID number	Date	Title
PNL-6049; Vol. III	1986	COBRA-SFS: A Thermal-Hydraulic Analysis Computer Code; Vol. III: Validation Assessments
EPRI-NP-3764	1984	Thermal-Hydraulic Analysis of Consolidated Spent PWR Fuel Rods
PNL-5781	1986	COBRA-SFS Predictions of Single-Assembly Spent Fuel Heat Transfer Data
EPRI-NP-4593	1986	Comparisons of COBRA-SFS Calculations to Data from Electrically Heated Test Sections Simulating Unconsolidated and Consolidated BWR Spent Fuel

Validation of multi-assembly casks and storage systems is rather more difficult than simulation of single-assembly tests. Due to the greater complexity of the geometry, and the sheer number of nodes that must be included, the possibility of compensating errors is vastly increased. As part of the code validation for multi-assembly applications, COBRA-SFS has been used to obtain pre-test predictions for a number of cask test programs, such as the CASTOR-V/21, the TN24P, the Westinghouse MC-10, and the VSC-17 casks.

These calculations were made essentially 'blind', in that in many cases the COBRA-SFS runs were made before the tests were conducted. In most cases, discrepancies between the pre-test predictions produced by the code, and the actual test results could be traced to deviations of the experimental design or procedure from the test plan. When the code input was modified to make the model more representative of the actual test conditions, the predictions were generally in excellent agreement with the data. Table 6.2 lists the documentation of the multi-assembly validation work performed on COBRA-SFS Cycle 1, and on various subsequent interim developmental versions of the code.

As a result of the validation work that has been performed with COBRA-SFS, the code can be applied with confidence to analyses of unconsolidated or consolidated rod arrays, in single assembly canisters or multi-assembly storage casks, oriented vertically or horizontally, with gas or vacuum back-fill. Boundary conditions describing external conditions can be defined on the top, bottom and sides of the system, including source terms due to external heating.

Capabilities exist in the code to model heat transfer due to forced or free convection, conduction, and thermal radiation in unconsolidated and consolidated spent fuel rods in single assemblies or casks. The code can also model the effects of inlet and outlet plena, the structures of the cask walls and support structures, and the external environment. The user is assured of a reasonable expectation that the code will be able to calculate such problems successfully, but the worth of those results can only be judged in light of the results of an appropriate validation of the code for such an application.

**Table 6.2. Multi-Assembly Validation of COBRA-SFS Code**

ID number	Date	Title
PNL-5917 EPRI-NP-4887	1986	CASTOR-V/21 PWR Spent Fuel Storage Cask Performance Testing and Analyses
PNL-6054 EPRI-NP-5128	1986	TN-24P PWR Spent-Fuel Storage Cask: Testing and Analyses
PNL-6631 EPRI-NP-6191	1989	Testing and Analyses of the TN24P PWR Spent Fuel Dry Storage Cask Loaded with Consolidated Fuel
PNL-5777, Vol. II	1986	BWR Spent Fuel Storage Cask Performance Test, Vol. II: Pre- and Post-Test Decay Heat, Heat Transfer, and Shielding Analyses
PNL-5974	1986	CASTOR-1C Spent Fuel Storage Cask Decay Heat, Heat Transfer, and Shielding Analysis
PNL-5802	1986	COBRA-SFS Thermal-Hydraulic Analysis of the CASTOR-1C and REA 2023 BWR Storage Casks Containing Consolidated Spent Fuel
PNL-7839 EPRI-TP-100305	1992	Performance Testing and Analyses of the VSC-17 Ventilated Concrete Cask

### 6.1.3 What To Do When the Code Fails

It may be humbling to have to admit it, but it is usually the user's own fault when the code fails. The COBRA-SFS code has seen a great deal of use in a wide variety of applications. Most of the more obvious bugs and errors have been worked out of the code. When it fails in a particular calculation, the problem can usually be traced to something incorrect in the input (e.g., inadvertent errors in the input values, misapplication of particular models, or inappropriate boundary conditions). Therefore, the first step in investigating any problem with the code's performance should be a cold-eyed, relentless scrutiny of the input file.

Numerous examples of code applications are presented in Section 7.0 with the validation test cases. Most of the models in the code are exercised in one way or another in these problems. In addition, the input instructions in Section 6.2 give detailed explanations of the form and usage of each input line in the code. Where necessary, interrelationships between different parts of the input stream are pointed out, and any incompatibilities are noted. There is also some internal checking for errors within the code. But because the input is so extremely flexible, there is a definite limit to how effective this can be. The code checks for violations of the dimension parameter specifications, and for inconsistencies in the view factor input for radiative heat transfer. Some checking is done on the

specification of the channel and slab node interconnections, to make certain that the input conventions have not been violated. But the code has no means of divining what the geometry *should* look like, as compared to how the input describes it.

If careful examination of the input uncovers no obvious errors, incompatible modeling specifications, or inappropriate boundary conditions, the next step is to see how far the solution behavior managed to get before failing. The flow and heat transfer problems solved by the code are generally quite complex, and in many cases highly nonlinear. There are two basic ways the solution can get into trouble: 1) by driving toward a solution far outside the expected reasonable range of the answer, or 2) by being unable to converge on an answer even after many iterations. If there are no input errors, the underlying cause of the failure is similar in both cases; non-linearities in the problem are interfering with the orderly procession of the solution toward convergence. How to deal with the matter, however, is slightly different in each case, as described below.

#### 6.1.3.1 Code Fails with a Cryptic Error Message

The error message is "some variable is not being interpolated properly by subroutine curve." In most cases, the variable is the fluid enthalpy, and the problem is not with the interpolation algorithm in the code, as the message seems to imply. The interpolation algorithm is probably working fine; the message simply means that the variable has exceeded the range of the properties table for the fluid, as specified in input group PROP (see Section 6.2.2). The user should first check the input to determine if the specified range of the table is large enough to accommodate the full range of conditions expected in the problem. If the range seems a little narrow, it might be a good idea to add a line or two to the properties table and try running the case again.

If the problem occurs on the very first iteration, there is only a slight possibility that increasing the range of the fluid properties table will take care of matters and let the iteration proceed to a proper solution. For a problem with a specified pressure drop boundary condition (with or without the plenum model), this error message on the first iteration might simply mean that the initial conditions are a very bad guess. If the value specified in GIN on OPER.2 for the flow rate is significantly less than the final value, unrealistically high fluid temperatures may be calculated in the first iteration. In such a case, it may help to increase the specified flow rate to a value closer to the expected final value.

If, however, the problem occurs after a number of iterations, the trouble is probably more serious, and may indicate an instability in the solution of either the conductive or radiative heat transfer energy exchange. The thermal connections between different slab nodes, and between the slab nodes and the fluid, are implicit within a given axial level. They are connected only semi-implicitly for radiative heat transfer within a given axial level, and for conduction to the axial levels above and below a given level. The heat transfer solution for a case in which slab nodes have strong semi-implicit thermal connections, in comparison to their implicit connections, may experience an instability.

This instability can be diagnosed by running the case again with the maximum number of iterations set to one less than the value at which the code printed the error message. The resulting output for a slab node that is experiencing this instability will show a saw-tooth temperature profile in the axial direction. The problem can usually be dealt with by specifying some degree of damping for



the solution of the slab node or nodes that exhibit the sawtooth pattern. This is accomplished by specifying a slab damping factor with the variable SLDAMP on CALC.2, then identifying the slab node(s) to be damped, using the input on lines CALC.5 and CALC.6.

The error message may also appear after some number of iterations if there are improperly specified radiative or conductive thermal connections between nodes in the problem. This sort of error has the net effect of creating or destroying energy in the model, so that in some node the enthalpy or temperature may be increasing or decreasing unnaturally with every iteration. The problem is due to an input error of some kind, but it may be rather difficult to find the node that is in error for a case with a large number of nodes and complex interconnections. One relatively simple way to check for this possibility is to perform an isothermal test case.

To run an isothermal case, the power generation rate is set to zero and all boundary temperatures are set to some uniform value. If all thermal connections have been specified properly, the temperature of all fluid, rod and slab nodes should be at exactly the boundary value. If they are not, the region with the largest discrepancy should be examined to determine the source of the problem. If an isothermal case is not sufficient to uncover the problem, selective removal of different radiative heat transfer groups, slab connections, or plenum connections may also be of help in isolating the problem.

### 6.1.3.2 Code Fails to Converge within the Specified Number of Iterations

The first and most obvious question to ask is how many iterations should be sufficient to achieve convergence. The default of 20 iterations in the code is a hold-over from the halcyon days of reactor core thermal-hydraulic analysis, in which it was not uncommon for the code to be able to converge in less than a dozen iterations, even for very complex problems. The applications to cask analysis generally involve much more challenging problems, and it is not unusual for several hundred iterations to be required to achieve convergence. However, more than 300 iterations is getting a bit excessive, and more than 500 is a signal that the code is probably working too hard. More than a thousand iterations is definitely too many, and something probably should be done to find out what is wrong.

In such a case, the user should examine carefully the iteration summary produced at the head of the output of results. It lists the main solution errors against which the convergence criteria are tested, in the following format.

iteration no.	sweep no.	peak clad				Total flow (lbm/s)	pressure drop (psi)	error			
		temp (F)	level	rod	assm.			total energy	flow	fluid energy	rod energy
1	1	381.9	12	13	4	0.338E+01	-0.2353495	-2.3474	56.2054	0.0035	0.0002
	2	382.6	11	13	4			-2.0857	56.2054	0.0233	0.0001
2	1	449.6	9	13	4	0.253E+01	0.0054958	-1.0128	3.0014	0.0780	0.0038
	2	474.4	9	13	4			-0.8751	3.0014	0.1611	0.0007

This example shows the first two iterations for a calculation that took 214 iterations to converge (see Section 7.3.2, PSN concrete cask with nitrogen back-fill). In examining the convergence behavior of a given case, the significant items of interest in the iteration summary are the four error terms, which are tested against the convergence criteria specified by the user in the input for group CALC, on line CALC.2. The total energy error is tested against the input variable QERROR. The flow error is tested against the input variable FERROR. The fluid energy error is tested against the input variable HERROR. The fluid continuity solution is also tested against the input variable WERRY during the inner iteration, but no information is printed on this in the iteration summary (see Section 6.2.12, input group CALC, line CALC.2 for a complete discussion of this input).

In general, these error terms should show a monotonically decreasing trend. If the errors are oscillating (i.e., jumping between nearly the same values over the course of two or three iterations), the behavior may be due to the stability problem described above. Strong semi-implicit coupling along with relatively weak implicit coupling between nodes for the heat transfer solution may not be severe enough in some cases to cause the code to actually fail. It can, however, result in very slow rates of convergence, or oscillation of the solution about an asymptote that may or may not be approaching the correct solution. In such a case, the problem can be diagnosed in the same way as described above, and much the same action needs to be taken to fix the problem. That is, by looking at the output for saw-tooth patterns in the axial temperature profiles, and damping any nodes that exhibit such behavior.

If the total energy error is not oscillating, but appears to be stuck on some value that does not change from iteration to iteration, the problem may be improperly specified thermal connections. This sort of input error can cause the code to fail, as noted above in Section 6.1.3.1, if the resulting energy error is large enough. But alternatively, it may simply create a source of energy error that cannot be iterated away. The problem can usually be diagnosed using the same procedures as outlined above, starting with running an isothermal case to determine the existence of an energy source or sink somewhere in the problem.

If the error summary shows steadily decreasing values, but they are going down at an irritatingly slow rate, there are a number of possible causes of the problem and different actions to be taken. Some of the more likely possibilities are discussed below, with suggestions for appropriate action.

If the problem is starting with an initial guess that is too far from the final solution, it may take an abnormally long time to reach convergence. In such a case, the best thing to do is to start with a better initial guess, as defined with the operating conditions in group OPER on line OPER.2. If the case uses a uniform pressure drop boundary condition for a specified total flow (with or without the plenum model), it may be helpful to start with an initial specified flow rate (entered as GIN on OPER.2) that is closer to the expected final value. It may also be helpful to adjust the initial enthalpy (entered as HIN on OPER.2).

Another possible cause of a slow convergence rate is that the specified damping factors on CALC.2 are too conservative for a particular case. These can be reduced or eliminated by appropriate changes in the input for CALC.2. If the damping is not needed, the code will almost always converge more rapidly without the damping factors. However, removing the damping factors could in some cases cause the code to fail due to instabilities, as described above. As disastrous as this

result might seem to the user, it is actually a useful improvement over very slow convergence rates. If the failure is due to instabilities of the type described above, the user will be motivated to search out the specific nodes that are hindering convergence. Appropriate damping factors can then be applied specifically to the troublesome nodes, using the damping factor input variable SLDAMP on CALC.2 and identifying the specific nodes to be damped on CALC.5 and CALC.6. This allows damping to be applied where it is needed, without unnecessarily impacting the global convergence rate for the problem.

The optional axial energy rebalancing can sometimes result in a dramatic improvement in the convergence rate. This feature is specified by setting the variable ACCAX to a non-zero value on CALC.2. Axial energy rebalancing consists of performing a one-dimensional energy solution for the system every three iterations, and using the results to make a global adjustment in all nodes at all axial levels. In many cases this will propagate changes through the system much more rapidly than the node-by-node solution, and can result in much faster convergence. It can, however, destabilize the solution, and so should not be used unless it can be shown to be a definite help. The results of the axial rebalancing calculation can be seen directly by setting the variable IPREP on OUP.1 in group OUP. However, this yields a very large amount of output even for problems of moderate size, and probably would not be useful except for debugging purposes.

One last option to consider if the code fails to converge is to examine the values selected for the convergence criteria. If they are unnecessarily stringent, they could be forcing the code to calculate many iterations with no real change in the solution. A quick and easy check to determine if the code is doing more iterations than it really needs is to look at the peak clad temperature in the iteration summary. If it is not changing between iterations, this indicates that no substantial changes are occurring in the solution. In such a case, the additional iterations are probably a waste of time, and it would be appropriate to relax the convergence criteria slightly.

The appropriateness of relaxed convergence criteria can be evaluated by comparing the results obtained with different values for the convergence criteria. If more relaxed convergence criteria yield essentially the same results as the same case with tighter convergence criteria, and does so in fewer iterations, then the relaxed convergence criteria are probably appropriate for the problem. Some degree of judgement is required from the user to determine what constitutes the appropriate standard of convergence for a given application. In general, it is sufficient for a calculated answer to be approximately as accurate as the measured value with which it is validated.

## 6.2 Input Instructions for COBRA-SFS

The input instructions for COBRA-SFS are provided in this section. The code input is listed line by line, with the definition of each input variable. Each set of instructions is preceded by an explanatory paragraph. However, users unfamiliar with COBRA-SFS should refer to Section 6.1 for a discussion of the modeling capabilities and limitations of the code, and the list of references documenting the various applications and validation calculations performed with the code. These reports are intended in part to provide guidance and examples for the code user in developing new input files and applications.

COBRA-SFS input is organized into groups identified by unique four-character flags. The groups are defined by function and reflect a generalized logical approach to setting up input for a problem. COBRA-SFS input falls into the following five basic categories:

1. the physical properties of the solid materials and working fluid
2. the flow channel geometry and solid node structure
3. the constitutive models for the flow and heat transfer solutions
4. the boundary conditions
5. solution control parameters and output options.

The various input groups under these five categories are listed below. Note that many of the input groups are optional.

#### **Case Control Data**

Problem identification and initialization

#### **Thermodynamic and Material Properties**

Group PROP - Fluid and Solid Material Properties

#### **Geometry Description**

Group CHAN - Flow Field Geometry

Group VARY - Geometry Variations (optional)

Group RODS - Fuel Rod Geometry (optional)

Group SLAB - Solid Structure Geometry (optional)

Group RADG - Thermal Radiation Exchange Factors (optional)

#### **Constitutive Models**

Group HEAT - Heat Transfer Correlations

Group DRAG - Friction Factors and Pressure Loss Coefficients

#### **Boundary Conditions**

Group BDRY - Thermal Boundary Conditions (optional)

Group OPER - Operating Conditions

#### **Code Control**

Group CALC - Calculational Parameters

Group OUTP - Output Options

The information required to initiate a COBRA-SFS calculation is described in Section 6.2.1. The fluid and solid material properties are read in group PROP, which is described in Section 6.2.2. Note that input group PROP is not optional; properties of the fluid must be supplied in all cases.

The geometry of the problem is described in group CHAN, which is a required input group, and in optional groups VARY, RODS, SLAB, and RADG. The flow field geometry, described in input group CHAN, defines the flow regions within the problem. This input group is described in Section 6.2.3. Group VARY can be used to describe variations of flow channel areas and connections with respect to axial location. This optional input group is described in Section 6.2.4.

The fuel rod geometry input in group RODS is required only if there are heat generating rods in the problem. The input for this group is described in Section 6.2.5. Section 6.2.6 describes the input for optional group SLAB, which is used to describe heat-conducting solid structures in the system. When thermal radiation is modeled, the geometric viewfactor information in group RADG is required input. This group is described in Section 6.2.7.

The constitutive models required by the energy and flow field solutions are described in groups HEAT and DRAG, respectively. The input for group HEAT, described in Section 6.2.8, specifies the heat transfer correlations used in the energy equations. The input for group DRAG, described in Section 6.2.9, specifies the friction factor correlations and pressure loss coefficients used in the momentum equations.

The boundary conditions for the problem are described in groups BDRY and OPER. The input for group BDRY, described in Section 6.2.10, defines the thermal boundary conditions applied to the problem. The input for group OPER, described in Section 6.2.11, defines the boundary operating conditions for the flow solution and the transient forcing functions.

The parameters controlling various functions of the code are read in groups CALC and OUTP. The input for group CALC, described in Section 6.2.12, defines the calculational parameters controlling the numerical solution procedure. The input for group OUTP, described in Section 6.2.13, defines the parameters controlling the various output options.

Input is terminated by entering ENDD for the group flag. This signals the code to stop looking for additional input groups.

The new user would be well advised to read through the input instructions entirely before attempting to set up the input for a particular problem. It is important for the user to have a reasonably complete picture of the overall structure of the COBRA-SFS input, as it will give order to the large array of options available.

The general format of the input instructions is to give a complete description of all variables in each line of input, including the format for reading the data. Some input lines are repeated, and some groups of lines are repeated in sequence as a set. These repetitive patterns are noted in the instructions, both in the format for the line and in the descriptive text accompanying the input. In many instances, later input will depend on values specified in earlier input lines. These flags are noted on the input line, with a reference to the line on which the flag was defined. For example, the variable defining the number of channels in an assembly, NCHANA, is read on input line CHAN.5. In all subsequent instructions that refer to NCHANA, the origin of this flag is denoted by specifying the variable as NCHANA[CHAN.5].

### 6.2.1 Problem Initiation Input

Input records COBRA.1 and COBRA.4 are required at the beginning of every COBRA-SFS input file. COBRA.1 defines the execution time allotted for the problem, and offers the user the opportunity to restart a previously executed problem from the TAPE8 restart dump. The case title is specified on COBRA.4. This title is printed on the output file, and serves as the case identifier, for easy reference. The date and time of the calculation are also written to the output file along with the title. If the restart option is flagged, COBRA.2 must also be read, to define the restarted calculation.

**COBRA.1** MAXT,IECHO  
FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	MAXT	CPU time limit (sec.) for problem execution. For restart cases, enter MAXT as a negative number (ABS(MAXT) will be used to define the time limit). (NOTE: The value entered for MAXT cannot supersede any execution time limit specified in the job control language in a batch submittal. The user must ascertain that any such time limits are compatible with the input value of MAXT.)
6-10	IECHO	Flag for printout of input file;  = 0; the input file will be printed on the output file (default).  = 1; the input file will not be printed on the output file.

**COBRA.2** NJUMP,NA,IT,NTT,TTT Read only if MAXT < 0 on COBRA.1  
FORMAT(4I5,F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NJUMP	Restart option flag;  = 1; continue calculation from a previous steady state or transient solution; new input may be read in to define a new case, with the previous calculation as the first guess.  = 2; not a restart; print output information from a previous calculation on tape8.

6-10	NA	Number of iterations for restarted case; <ul style="list-style-type: none"> <li>- for a steady-state calculation, number of additional iterations.</li> <li>- for a transient calculation, number of iterations per time step              (NOTE: If new input is read for group CALC, the value for NA will be superseded by the new value for NTRIES on CALC.3.)</li> </ul>
11-15	IT	Flag to define type of restart: <ul style="list-style-type: none"> <li>= 0; restart and continue the previous solution;             <ul style="list-style-type: none"> <li>- continue iterating in a steady-state case</li> <li>- continue time steps in a transient case</li> </ul> </li> <li>= 1; restart a steady-state case and initiate a transient              (NOTE: Input to define the transient must be specified by new input for group CALC. Forcing functions may be specified in group OPER, either with new input or in the original input file.)</li> </ul>
16-20	NTT	Number of transient time steps for the restarted calculation.
21-25	TTT	Total transient time (seconds) for the restarted calculation.

All cases, including a restart case, must have a title, which is read on line COBRA.3. This is an alphanumeric identifier that will be printed on each page of the output file, along with the date and time of the run.

**COBRA.3** J1,(TEXT(I),I=1,17)  
 FORMAT(5X,I5,17A4)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5		Blank
6-10	J1	flag for option controlling output of input data; <ul style="list-style-type: none"> <li>≤ 1; print all processed input data from the current input file.</li> <li>= 10; print all processed input data from the current input file, then stop execution.</li> </ul>
11-78	TEXT(I)	Title for problem identification, for label printed at top of each page in the output file (maximum of 68 characters).

When doing a restart calculation, (i.e., NJUMP=1), the user has the opportunity to modify the input by specifying new data for the input groups PROP, VARY, RADG, HEAT, DRAG, BDRY, OPER, and OUTP. The problem geometry, however, which is specified in groups CHAN, RODS, and SLAB, cannot be changed on a restart. Changes in these groups constitute a whole new problem, and a restart is neither useful nor appropriate. After all modified groups have been entered, or if no changes or modifications are needed in the input on a restart, the user must enter the ENDD group flag to terminate the search for new input for the restart.

**COBRA.4** AGROUP Read only if NJUMP > 0  
 FORMAT(A4)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter ENDD, to signal end of input stream for this restart case.

### 6.2.2 Group PROP--Fluid and Material Properties

This group is used to define thermodynamic and heat transfer properties for the fluid and solid materials included in the problem. Properties can be specified for only two different fluids, but any number of solid material properties can be specified in a problem. Properties for fluids are defined on PROP.2 and PROP.3; solid material properties are defined on PROP.4. At least one fluid must be specified to define the coolant, but properties for solid materials are optional.

**PROP.1** AGROUP,NPROP,NSPROP  
 FORMAT(A4,1X,2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter PROP.
6-10	NPROP	Number of elements in fluid properties table.  Enter 0 to use option for two sets of fluid properties. (The number of elements for each table will be specified on PROP.2.)
11-15	NSPROP	Number of solid materials for which properties will be entered on PROP.3.

#### Fluid Properties

If the coolant is a liquid, properties are entered as saturated liquid values for the pressure PLIQ(I). If the coolant is a gas, properties are entered for temperatures at the system pressure (which is specified on OPER.2), and variable PLIQ(I) is not used. Fluid properties for at least two reference temperatures are required to allow interpolation.

Different fluids can be specified for different parts of a given case. When this option is used, NPROP[PROP.1] is set equal to zero, and the number of elements for each of the two fluid properties



tables is defined on PROP.2. PROP.2 and PROP.3 are read twice in sequence to specify two sets of fluid properties. If NPROP[PROP.1] is greater than zero, only one fluid properties table will be supplied, and PROP.2 is not read. The value of NPROP is automatically assigned to NFPROP(1), and only one fluid property table is read on PROP.3.

**PROP.2** NFPROP(J),FNAME(J) Read only if NPROP[PROP.1] = 0  
 FORMAT (I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NFPROP(J)	Number of fluid property table elements to be read on PROP.3 for the Jth table.
6-10		blank
11-20	FNAME(J)	alphanumeric identifier for the Jth fluid properties table

**PROP.3** (PLIQ(I,J),TEMLIQ(I,J),HLIQ(I,J),CONLIQ(I,J),CPLIQ(I,J),  
 VLIQ(I,J),VISLIQ(I,J),BLIQ(I,J),I=1,NFPROP(J))  
 FORMAT(8E10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	PLIQ(I,J)	Saturation pressure for liquid coolant (psia). (Note: PLIQ(I) is not used when coolant is a gas.)
11-20	TEMLIQ(I,J)	Temperature (°F).
21-30	HLIQ(I,J)	Enthalpy (Btu/lbm).
31-40	CONLIQ(I,J)	Thermal conductivity (Btu/h-ft-°F).
41-50	CPLIQ(I,J)	Specific heat (Btu/lbm-°F).
51-60	VLIQ(I,J)	Specific volume (ft <sup>3</sup> /lbm).
61-70	VISLIQ(I,J)	Viscosity (lbm/ft-h).
71-80	BLIQ(I,J)	Coefficient of Thermal expansion (/°F)

\*\*\* If NPROP[PROP.1] > 0, PROP.3 is read NPROP times. \*\*\*

\*\*\* If NPROP[PROP.1] = 0, PROP.3 is read NPROP(J) times. \*\*\*

\*\*\* PROP.2 and PROP.3 are read twice in sequence if NPROP[PROP.1] = 0 \*\*\*

## Solid Material Properties

Solid material properties are specified for NSPROP[PROP.1] materials. This input is optional, and is read only if NSPROP on PROP.1 is greater than zero. Properties are assumed constant for all temperatures. The material type number is used to identify the material properties of the various solid structure nodes input in groups SLAB and BDRY. Material property types must be numbered sequentially from 1 to NSPROP.

**PROP.4** (IMAT(I), ANAME(I), CPSOL(I), RHOSOL(I), CON0(I), CON1(I), CON2(I), CON3(I), NQVT(I), I=1, NSPROP[PROP.1])  
 Read only if NSPROP > 0 on PROP.1  
 FORMAT (I5, A5, 6E10.0, I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IMAT(I)	Solid material type identification number.
6-10	ANAME(I)	Solid material name (for user convenience).
11-20	CPSOL(I)	Specific heat (Btu/lbm-°F) (for transient calculations).
21-30	RHOSOL(I)	Density (lbm/ft <sup>3</sup> ) (for transient calculations).
31-40	CON0(I)	Coefficients of an expression for solid thermal conductivity (Btu/h-ft-°F) that has the form:
41-50	CON1(I)	
51-60	CON2(I)	
61-70	CON3(I)	

$$k = \text{CON0(I)} + \text{CON1(I)}*T + \text{CON2(I)}*T^2 + \text{CON3(I)}*T^3$$

where T is the temperature in °R.

71-75	NQVT(I)	Transient heating rate forcing function. Corresponds to profiles read in OPER.14 and 15.
-------	---------	--

\*\*\* PROP.4 is read NSPROP[PROP.1] times. \*\*\*

### 6.2.3 Group CHAN--Flow Field Geometry

This group is read to define the flow field geometry. The basis of the problem geometry is subchannel modeling, in which the flow field is represented as an array of channels that can communicate laterally by crossflow. For convenience in modeling fuel bundles and casks, the channels are grouped into assemblies; a group of channels that can communicate laterally comprise an assembly. Channels of one assembly cannot communicate by crossflow with the channels of other assemblies.

## Axial Geometry

The axial geometry of the channel region is defined by the axial length,  $Z$ , entered on CHAN.23 and the number of axial nodes,  $NDX$ , entered on CHAN.1. All channels in all assemblies have the same axial length, and the same axial noding. The default in the code is a uniform axial node size, which is calculated as  $DX = Z/NDX$ . The user has the option, however, of specifying axial nodes of varying sizes, by setting  $NAZONE$  on CHAN.1 and reading in a variable axial node size table on CHAN.4.

**CHAN.1** AGROUP,NASSEM,NDX,NAZONE,ISHEAR,NSHEAR,NANGLT  
 FORMAT(A4,1X,6I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter CHAN.
6-10	NASSEM	Number of assemblies.
11-15	NDX	Number of axial nodes.
16-20	NAZONE	Flag for variable axial node sizes; = 0; uniform axial nodes (default). > 0; number of regions in variable axial node size table (read on CHAN.4).
21-25	ISHEAR	Flag for fluid shear stress (NOTE: used only when modeling plenum regions with multiple channels); = 0; fluid shear stress not considered. = 1; fluid shear stress included.
26-30	NSHEAR	Number of pairs of gaps that are connected by fluid-fluid shear in the lateral direction.
31-35	NANGLT	Number of lateral control volume orientation angles (read on CHAN.2 below) (NOTE: this input is needed for the fluid-fluid shear option, but it is also used to define the direction of the gravity vector for lateral momentum transport in a horizontal or tilted geometries.)

CHAN.2 (ANGLE(I),I=1,NANGLT(CHAN.1))  
 Read only if NANGLT[CHAN.1] > 0  
 FORMAT (8E10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	ANGLE(I)	Orientation angle for lateral flow, in degrees from vertical; to be assigned to specific gaps on CHAN.9.
11-20		
etc.		

(NOTE: this input is used only if fluid-fluid shear is included in the model (i.e., ISHEAR=1 on CAN.1), or if modeling lateral gravity terms in a horizontal or tilted geometry.)

CHAN.3 Z,THETA  
 FORMAT(2F10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	Z	Axial length of the channel region (in.).
11-20	THETA	Channel orientation, in degrees from vertical (Default is 0.0. Used to calculate gravity term in the momentum equation).

CHAN.4 (NSTEPS(I),VDX(I),I=1,NAZONE) Read only if NAZONE > 0 on CAN.1  
 FORMAT(8(I5,E5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	NSTEPS(I)	Number of nodes in zone I. (Note: the total number of nodes, $\sum$ NSTEP(I), must be equal to NDX, specified on CAN.1.)
6-10, 16-20, etc.	VDX(I)	Axial node length (in.) in zone I. (Note: the sum of all nodes must be equal to Z on CHAN.2.)

\*\*\*\* up to four pairs of NSTEPS,VDX data may be entered per record. \*\*\*\*  
 \*\*\*\* If NAZONE is greater than 4, CHAN.4 is read repeatedly until \*\*\*\*  
 \*\*\*\* all NAZONE pairs of data have been specified. \*\*\*\*

### Assembly Geometry Description

The input lines CHAN.5 through CHAN.7 are read in sequence as a set NASSEM[CHAN.1] times, once for each assembly in the problem. All channels are identified in this manner, assembly by assembly. An assembly may contain any number of channels. The assembly number and the assembly type number are read on CHAN.5. Each assembly must have a unique number (1 through NASSEM), but different assemblies can have the same assembly type number, if they have the same geometry. An assembly type is defined by the number of channels and their interconnections, as

specified by input on CHAN.7. It is necessary to enter the geometry data for an assembly type only once, with the CHAN.7 input for the first assembly of that type. For subsequent assemblies of that type, the value entered for ITYPA on CHAN.5 is recognized as a previously defined type, and CHAN.7 is not read. The geometry input for any subsequent assembly of the same type is automatically copied into the appropriate arrays.

The flow and heat transfer correlations to be used must be defined separately for each assembly, however. This information is supplied on CHAN.6, along with the index of the axial heat generation profile table (if the assembly contains heated rods).

**CHAN.5** NASS,ITYPA,NCHANA,INTAPE,IFREE,TMNCVL(NASS)  
 FORMAT(4I5,E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NASS	Assembly number.
6-10	ITYPA	Assembly type number (corresponding to a channel geometry description entered on CHAN.7).
11-15	NCHANA	Number of channels in assembly NASS.
16-20	INTAPE	Flag for I/O unit source of the channel geometry input on CHAN.6 for assembly NASS:  = 0; read from the input file (default).  = N; read from I/O unit N.
21-25	IFREE	Flag for specifying wall shear boundary condition option for assembly NASS:  = 0; wall shear specified by friction factor correlation (default).  = 1; laminar zero-slip wall boundary condition. NOTE: this option is used only when modeling large open plenum regions where fluid-fluid shear must be taken into account.)
26-30	TMNCVL(NASS)	Factor to adjust transverse momentum control volume length in assembly NASS so that the length is defined as $L_{NASS} = \ell_v / TMNCVL(NASS)$ in the calculation of the turbulent crossflow, $w'$ . (Default for TMNCVL(NASS) is 0.0, so that $L_{NASS} = 1.0$ for $w'$ .)

**CHAN.6** NAFLX(NASS),NFLMC(NASS),NHFVT(NASS),NPFVT(NASS),MDFLT,  
NFASS(NASS),ITDPA(NASS)  
FORMAT(7I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NAFLX(NASS)	Identification number of the axial heat flux profile table [specified on OPER.16 and OPER.17] for assembly NASS (default is 1).
6-10	NFLMC(NASS)	Identification number of the heat transfer coefficient correlation [specified on HEAT.2] for assembly NASS (default is 1). (NOTE: If NTHEAT > 0 on HEAT.1, this is also the identification number of the heat transfer coefficient correlation for lateral flow in assembly NASS. IF NFHEAT > 0 on HEAT.1, this is also the identification number of the heat transfer coefficient correlation for free convection in assembly NASS.)
11-15	NHFVT(NASS)	Identification number of the heat generation versus time forcing function [specified on OPER.13 and OPER.14] to use in assembly NASS during a transient (default is 0; initial value is used throughout the transient.)
16-20	NPFVT(NASS)	Identification number of the pressure drop or flow versus time forcing function [specified on OPER.11 and OPER.12] for assembly NASS (default is 0; initial value is used throughout the transient).
21-25	MDFLT	Identification number for the friction factor correlation(s) [specified on DRAG.2 for axial flow and optionally on DRAG.13 for lateral flow], to be used in assembly NASS (default is 1). (NOTE: This value can be superseded in a given channel by specifying a non-zero value for N on CHAN.7 for an individual channel of this assembly type.)
26-30	NFASS(NASS)	Identification number of the fluid properties table [specified on PROP.3] for assembly NASS. (Default is table 1.)
31-35	ITDPA(NASS)	Flag for flow or pressure drop boundary condition for assembly NASS:  = 0; specified flow boundary condition. = 1; uniform pressure drop (specified in OPER). = 2; pressure drop resulting in uniform inlet mass flux. = 3; uniform pressure drop calculated for total mass flow rate (FTOTAL on OPER.2). = 4; zero flow at both top and bottom boundaries.  (NOTE: ITDPA(NASS) is needed only if different assemblies have different boundary conditions. If all assemblies see the same boundary condition, ITDPA(NASS) may be entered as 0 for each assembly, and the appropriate boundary condition option is then specified with variable ITDP on OPER.1 for all assemblies.)

CHAN.7 (N,I,AC,PW,PH,(NANGLE(K),LC(K),GAPS(K),DIST(K),K=1,4),  
 I=1,NCHANA[CHAN.5])  
 Read only for a new value of assembly type in ITYPA[CHAN.5].  
 FORMAT(I1,I4,3E5.3,4(I1,I4,2E5.2))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1	N	Index number of friction factor correlation to be applied to channel I; must correspond to a correlation entered on DRAG.2. Default is MDFLT, entered on CHAN.6, for all channels in the assembly. (NOTE: if the option for lateral friction factor correlations is used, N also must correspond to the index number of a lateral friction factor correlation entered on DRAG.13. See group DRAG input instructions.)
2-5	I	Channel identification number (must be entered sequentially, 1 through NCHANA[CHAN.5]).
6-10	AC	Channel nominal area (in. <sup>2</sup> ).
11-15	PW	Channel nominal wetted perimeter (in.).
16-20	PH	Channel nominal heated perimeter (in.).
21, 36, 51, 66	NANGLE(K)	Identification number of a lateral control volume orientation angle read on CHAN.2. (NOTE: this input is optional, and is needed only when the option for fluid-fluid shear or gravity forces in the transverse direction are modeled; see Section 6.1.2.)
22-25 37-40, 52-55, 67-70	LC(K)	Identification number of adjacent channel for the Lth connection to channel I, where LC(K) > I (each connection should be identified only once, as a connection from the lower-numbered channel to the higher-numbered channel).
26-30, 41-45, 56-60, 71-75	GAPS(K)	Width of flow connection between channel I and channel LC(K), (in.).
31-35, 46-50, 61-65, 76-80	DIST(K)	Transverse control volume length (in.) between channel I and channel LC(K). (NOTE: This input is optional. It defines the transverse length of the momentum cell and also the conduction length in the transverse direction. A default value for this distance can be defined for all flow connections by specifying the variable SL on CHAN.8. If this option is used, DIST(K) can be entered as zero.)

\*\*\* CHAN.7 is read NCHANA[CHAN.5] times for a given assembly type. \*\*\*

\*\*\* CHAN.5 through CHAN.7 are read sequentially NASSEM[CHAN.1] times. \*\*\*

## Lateral Flow Connection Parameters

All lateral flow connections between the channels within each assembly are defined in the input on CHAN.7. The flow solution for these connections requires an empirical term to define the cross-flow resistance (refer to the description of the lateral momentum equation in Section 2.3.3). For analysis of rod bundle arrays using subchannel modeling, a single number is usually sufficient to characterize all lateral connections. This input is read on CHAN.8. The user also has the option of specifying different loss coefficients for different gaps, as appropriate to model problems in which the lateral control volumes are not all essentially identical. This input is supplied in group DRAG. However, a value still must be entered for KIJ(1) on CHAN.8, for use in gaps where the lateral resistance is not defined in subsequent input. Even if lateral resistance values will be defined for all gaps in subsequent input, CHAN.8 must be read in. The code cannot look ahead in the input stream to see whether or not this input will be required.

A value must also be entered for the width-to-length ratio SL, to be used to define the centroid length for any gap that is not specified with a non-zero value for DIST(K) on CHAN.7. If all gaps have a non-zero DIST(K) value specified, SL is not used, and therefore can be entered as zero.

The index numbers of pairs of gaps that exchange momentum due to fluid-fluid shear must be identified by input on CHAN.9, if this option has been specified by input on CHAN.1. The user must understand the gap numbering convention in COBRA-SFS in order to correctly define this input. The gaps are numbered automatically in the code, in the order in which they are specified by input on CHAN.7. For example, in an array of subchannels such as that shown in Figure 6.4 for a fuel assembly, the input for CHAN.7 for subchannels 1 through 6 is as follows;

		NANGLE LC				NANGLE LC						
N	I	AC	PW	PH	↓	↓	GAPS	DIST	↓	↓	GAPS	DIST
1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	12345
0	1.0663	.5750	.16570		2.1984	.4860						
0	2.3004	1.226	.66290		3.1410	.4860	4.1984	.563				
0	3.0885	.6630	.66300		5.1410	.5630						
0	4.3004	1.226	.66290		5.1410	.4860	7.1984	.563				
0	5.1770	1.326	1.3260		6.1410	.5630	8.1410	.563				
0	6.0768	.7115	.49720		9.0790	.5630						



By the COBRA-SFS gap number convention, the gaps are automatically numbered as follows:

This pattern is continued until all subchannels have been identified by input on CHAN.7.

<u>CHAN. 7 line</u>	<u>subchannel II</u>	<u>subchannel JJ</u>	<u>gap number</u>
1	1	2	1
2	2	3	2
	2	4	3
3	3	5	4
4	4	5	5
	4	7	6
5	5	6	7
	5	8	8
6	6	9	9

**CHAN.8** KIJ(1),SL,FTM  
FORMAT(3E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	KIJ(1)	Nominal flow resistance in lateral connections between channels (default is 0.5).
6-10	SL	Width-to-length ratio for transverse momentum control volume. (NOTE: This is used only for gaps where DIST(K) is entered as zero on CHAN.7 default is 0.5).
11-15	FTM	Factor for turbulent mixing of momentum and energy (default is 0.0).

**CHAN.9** (III(L),JJJ(L),L=1,NSHEAR[CHAN.1])  
Read only if NSHEAR > 0  
FORMAT (12I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15 etc.	III(L)	Identification numbers of one gap in a pair that are connected by lateral fluid shear. (NOTE: refer to Section 6.1.2 for an explanation of the gap numbering convention in the code.)
6-10, 16-20 etc.	JJJ(L)	Identification numbers of the other gap in a pair that are connected by lateral fluid shear.

\*\*\*\* Up to six pairs of III,JJJ channel numbers may be entered per record. \*\*\*\*  
\*\*\*\* If NSHEAR is greater than six, repeat this line until all NSHEAR \*\*\*\*  
\*\*\*\* pairs of data have been specified. \*\*\*\*

## 6.2.4 Group VARY--Geometry Variations

This group is used to specify axial variations in channel flow area and lateral flow connection width. It is optional. If this group is not used, all channel areas and lateral flow connection widths are assumed constant over the entire axial length.

**VARY.1** AGROUP,NAFACT,NAXL,NARAMP,NGAPS,NGXL  
FORMAT(A4,1X,5I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter VARY.
6-10	NAFACT	Number of channels with area variations.
11-15	NAXL	Number of axial locations for channel area variations. (NOTE: All area variation tables use the same table of axial locations.)
16-20	NARAMP	Number of iterations for gradual insertion of area variations into the calculation (default is 1).
21-25	NGAPS	Number of lateral flow connections for which the width varies axially.
26-30	NGXL	Number of axial locations for lateral flow connection width variations. (NOTE: All gap width variation tables use the same axial locations.)

### Channel Area Variations

The channel area variations are read only if NAFAC and NAXL on VARY.1 are both greater than zero. The channel area variations are specified on VARY.2 through VARY.4. VARY.2 is read once to define the axial locations of the area variations, then the input lines VARY.3 and VARY.4 are read sequentially, NAFAC times, to define the area variations and the channels affected. Line VARY.3 is read to specify the assembly number and index number of a channel in the assembly that has area variations, then VARY.4 is read to define the table of area variation factors in that channel, at the locations defined by input on VARY.2.

**VARY.2** (AXL(J),J=1,NAXL[VARY.1]) Read only if NFACT > 0 and NAXL > 0 on VARY.1  
 FORMAT(12F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	AXL(J)	Relative axial location (x/L) of the Jth channel area variation.

**VARY.3** NASS,I Read only if NAFAC T > 0 and NAXL > 0 on VARY.1  
 FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NASS	Identification number of assembly containing channel I.
6-10	I	Identification number of channel in assembly NASS for which area variations are being specified.

**VARY.4** (AFACT(L,J),J=1,NAXL[VARY.1]) Read only if NAFAC T > 0 and NAXL > 0 on VARY.1  
 FORMAT(12F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10 etc.	AFACT(L,J)	Channel area variation factor ( $A_j/AC_j$ ) for the Lth area variation table at axial level AXL(J), as defined on VARY.2. (NOTE: Area variation factor tables are numbered sequentially, in the order they are read in on VARY.4. Index L of AFACT is 1 to NAFAC T.)

\*\*\* VARY.3 and VARY.4 are read sequentially NAFAC T[VARY.1] times. \*\*\*

### Lateral Flow Connection Width Variations

The flow connection width variations specified on VARY.5 through VARY.7 are read only if NGAPS and NGXL on VARY.1 are both greater than zero. This input follows the same pattern used in defining the channel area variations. VARY.5 is read once to define the axial locations where gap width variations occur. Then VARY.6 and VARY.7 are read sequentially NGAPS times. VARY.6 is read to define the assembly number and the index number of a gap that has varying width. Then VARY.7 is read to define the table of width variation factors for that gap.

**VARY.5** GAPXL(J),J=1,NGXL[VARY.1]) Read only if NGXL > 0 and NGAPS > 0 on VARY.1  
 FORMAT(12F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	GAPXL(J)	Relative axial location (x/L) of the Jth gap width variation.

**VARY.6** NASS,K Read only if NGXL > 0 and NGAPS > 0 on VARY.1  
 FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NASS	Identification number of assembly containing lateral flow connection K.
6-10	K	Identification number of gap with width variation.

**VARY.7** (GFACT(L,J),J=1,NGXL[CHAN.1]) Read only if NGXL > 0 and NGAPS > 0 on VARY.1  
 FORMAT(12F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	GFACT(L,J)	Flow connection width variation factor $(GAP_{ij}/GAPS_k)$ for the Lth variation table at axial level GAPXL(J), as specified on VARY.5. (NOTE: Gap width variation tables are numbered sequentially in the order in which they are read on VARY.7. Index L of GFACT(L,J) is 1 to NGAPS.)

\*\*\* VARY.6 and VARY.7 are read sequentially NGAPS[VARY.1] times. \*\*\*

### 6.2.5 Group RODS--Fuel Rod Geometry

This group is read to define the geometry and material properties for the fuel rods that are contained in the assemblies defined in group CHAN. If a given problem does not contain fuel rods or heater rods simulating fuel rods, the input for this group is not needed. The fuel rod model is defined by the value specified for variable NC on RODS.1. If NC=0, there is no fuel rod model; the heat flux at the surface of the fuel rods is treated as a boundary condition in the calculation. In such a case, radiative heat transfer cannot be included in the solution. If NC=1, the conduction solution is performed for the rod cladding only. This provides a rod wall surface temperature for the heat transfer calculation. If NC>4, the conduction equation is solved within the fuel rod for internal node temperatures, as well as the cladding surface temperature. Radiative heat transfer can also be considered if NC is greater than zero on RODS.1.

The rod geometry input must be specified for each assembly, even if a given assembly does not contain fuel rods. A rod configuration type is specified for each assembly; assemblies that do not contain any rods are given a configuration type of zero. A nonzero configuration type is defined by the number of rods in the assembly, the nominal rod diameter, radial power distribution, fuel type, and channel connections for heat transfer.

**RODS.1** AGROUP,NC,NFUEL,NQAX,NRODTP,NTHETA,NRCON,NSCON  
 FORMAT(A4,1X,7I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter RODS.
6-10	NC	flag for fuel model option; = 0; no fuel model. = 1; only the rod cladding is modeled (NOTE: this option consider all modes of surface heat transfer, but not internal fuel conduction. It is applicable to steady state calculations only.) > 4; fuel model with (NC-1) finite-difference nodes in the fuel, and one node in the cladding. (NOTE: this option considers all modes of surface heat transfer, plus internal fuel conduction. It can be used in steady state calculations, if needed, and is required for transient calculations.)
11-15	NFUEL	Number of fuel types for which thermal properties are to be specified on RODS.4. (NOTE: this parameter is not used if NC = 0.)
16-20	NQAX	Flag for temperature dependent fuel properties; = 0; constant fuel properties. = 1; temperature-dependent fuel properties for fuel type 1 only. (NOTE: this option is available only if NC > 4).
21-25	NRODTP	Number of tables of axial fuel type variations. (Applicable only if NC > 4.)
26-30	NTHETA	Number of circumferential nodes in a fuel rod (default is 1). (WARNING: the viewfactors input in group RADG must be consistent with this input variable.)
31-35	NRCON	Number of rod-to-rod thermal connections for conduction heat transfer.

36-40            NSCON            Number of rod-to-slab thermal connections for conduction heat transfer.

CAUTION:    The options for multiple circumferential nodes in the fuel rods and rod-to-rod or rod-to-wall contact for thermal conduction are still under development and need additional validation. The feature is included in this release, but is not recommended for general application in Cycle 2.)

**Rod Configuration Description**

A rod configuration type is defined by the number of rods it contains, the nominal rod diameter, the radial power distribution, and the connections for heat transfer between the rods and channels in an assembly. Each assembly described in group CHAN must be assigned a rod configuration type on RODS.2. If an assembly contains no fuel rods, the rod configuration type must be specified as zero. The rod configuration description for a given non-zero type is entered on RODS.3. For efficiency, a rod configuration description is entered only once. The rod geometry arrays for subsequent assemblies with the same rod type are automatically filled with the appropriate input. The rod configuration description input is read one assembly at a time for all NASSEM[CHAN.1] assemblies.

RODS.2    NOA,ITYPA,NORODS,INTAPE  
          FORMAT(4I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NOA	Assembly number; (Note: must be entered sequentially, 1 through NASSEM[CHAN.1].)
6-10	ITYPA	Rod configuration type number for assembly NOA. (NOTE: if there are no rods in assembly NOA, ITYPA must be entered as zero.)
11-15	NORODS	Number of rods in configuration type ITYPA. (NOTE: if ITYPA is zero, NORODS must be zero.)
16-20	INTAPE	I/O unit number from which the rod input for rod configuration type ITYPA is to be read (default is the input file).

**RODS.3** (N,I,DIA(I),RADIAL(I),(LR(I,L),PHI(I,L),L=1,6),I=1,NORODS[RODS.2])  
 Read only if NORODS > 0 on RODS.2.  
 FORMAT(1X,I1,I3,2E5.2,6(I5,E5.2))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-2	N	Rod material properties flag; if NRODTP = 0 on RODS.1, N is identification number of fuel material type for rod I, as defined by input on RODS.4 (default is N=1).  if NRODTP > 0 on RODS.1, N is identification number of a table of axially varying fuel type for rod I, as defined on RODS.7 and RODS.8.
3-5	I	Rod identification number. (NOTE: must be entered in sequence, from 1 to NORODS[RODS.2].)
6-10	DIA(I)	Rod outside diameter (in.).
11-15	RADIAL(I)	Radial heat generation factor for rod I, normalized to the average rod heat generation rate.
16-20, 26-30 etc.	LR(I,L)	Identification number of the Lth channel connected to rod I (up to 6 channels can be connected to rod I).
21-25, 31-35, etc.	PHI(I,L)	Fraction of the perimeter of rod I connected to adjacent channel LR(I,L). (Note: multiple rods can be modeled by one average rod of diameter DR(I) by specifying PHI = X, where X is the number of rods being modeled by rod I.)

\*\*\* RODS.3 is read NORODS times for each new rod configuration type. \*\*\*  
 \*\*\* RODS.2 and (optionally) RODS.3 are read sequentially as an input \*\*\*  
 \*\*\* set NASSEM[CHAN.1] times. \*\*\*

### Fuel Material Properties

Fuel material properties must be entered if a rod model is used (i.e., NC > 0 on RODS.1). The fuel material properties are defined by fuel type, and are numbered sequentially in the order they are read in on RODS.4. If the rod model is used, at least one fuel type must be specified using the input on RODS.4. If the axial fuel type variations option has been flagged (i.e., NRODTP > 0 on RODS.1), at least two fuel types must be defined.

**RODS.4** (KFUEL(I),CFUEL(I),RFUEL(I),DFUEL(I),KCLAD(I),CCLAD(I),RCLAD(I),  
TCLAD(I),HGAP(I),DROD(I),GEOMF(I),DFUELI(I),QVOID(I),  
I=1,NFUEL[RODS.1])  
Read only if NFUEL > 0 on RODS.1

FORMAT(12E5.0,E10.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	KFUEL(I)	Thermal conductivity (Btu/h-ft-°F) of fuel type I.
6-10	CFUEL(I)	Specific heat (Btu/lbm-°F) of fuel type I.
11-15	RFUEL(I)	Density (lbm/ft <sup>3</sup> ) of fuel type I.
16-20	DFUEL(I)	Pellet diameter (in.) for fuel type I.
21-25	KCLAD(I)	Cladding thermal conductivity (Btu/h-ft-°F) for fuel type I.
26-30	CCLAD(I)	Cladding specific heat (Btu/lb-°F) for fuel type I.
31-35	RCLAD(I)	Cladding density (lbm/ft <sup>3</sup> ) for fuel type I.
36-40	TCLAD(I)	Cladding thickness (in.) for fuel type I.
41-45	HGAP(I)	Fuel-clad gap conductance coefficient (Btu/h-ft <sup>2</sup> -°F) for fuel type I.
46-50	DROD(I)	Outside diameter of the fuel rod, including the cladding (in.), for fuel type I.
51-55	GEOMF(I)	Fuel rod geometry flag; = 0; solid cylindrical fuel rod with internal heat generation. = 1; annular cylindrical fuel rod with internal heat generation. = 2; annular cylindrical fuel rod with a heat flux boundary condition on the inner fuel surface.
56-60	DFUELI(I)	Inner diameter (in.) of an annular fuel rod of fuel type I (used only if GEOMF(I) > 0).
61-70	QVOID(I)	Heat generation rate (Btu/sec-ft <sup>3</sup> ) in the central void of an annular fuel rod of type I (used only if GEOMF(I) = 1).

\*\*\* RODS.4 is read NFUEL[RODS.1] times. \*\*\*



## Temperature-Dependent Fuel Properties

The user has the option of specifying temperature-dependent material properties for fuel type 1, using the input on RODS.5 and RODS.6. This option is selected by setting NQAX to 1 on RODS.1. However, all other fuel types will still have material properties that are constant for all temperatures.

**RODS.5**    NTNODE   Read only if NQAX = 1 on RODS.1  
              FORMAT(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NTNODE	Number of entries in the temperature-dependent material properties table for fuel type 1.

**RODS.6**   (TVARY(I),VARYK(I),VARYCP(I),VARYR(I),I=1,NTNODE[RODS.5])  
              Read only if NQAX = 1 on RODS.1  
              FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 21-25, 41-45	TVARY(I)	Temperature (°F).
6-10, 26-30, 46-50	VARYK(I)	Thermal conductivity (Btu/h-ft-°F) of fuel type 1 at temperature TVARY(I).
11-15, 31-35, 51-55	VARYCP(I)	Specific heat (Btu/lb-°F) of fuel type 1 at TVARY(I).
16-20, 36-40, 56-60	VARYR(I)	Density (lb/ft <sup>3</sup> ) of fuel type 1 at TVARY(I).

The temperature-varying material properties must be entered as a monotonically increasing table, with TVARY(1) the lowest temperature in the table, and TVARY(NTNODE) the highest temperature.

## Axial Fuel Variation

This input is optional, and is read only if NRODTP is greater than zero on RODS.1. It allows the user to account for axial variations in materials using the fuel types defined by input on RODS.4. The fuel types are assigned to axial ranges in tables of axial location versus fuel type, using the input lines RODS.7 and RODS.8. The entries must be monotonically increasing on axial location. The tables are numbered 1 through NRODTP, in the order they are entered on RODS.7 and RODS.8. When this option is used, the value specified for N on RODS.3 for each rod must correspond to the index of the table containing the appropriate fuel types, as entered on RODS.7 and RODS.8.

**RODS.7** KNZ Read only if NRODTP > 0 on RODS.1  
 FORMAT(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	KNZ	Number of axial zones in the Ith table of axial distance versus fuel type, where I is from 1 to NRODTP[RODS.1].

**RODS.8** ((ZEND(I,K),IZTYP(1K),K=1,KNZ[RODS.7]),I=1,NRODTP[RODS.1])  
 Read only if NRODTP > 0 on RODS.1.  
 FORMAT(6(E5.0,I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	ZEND(I,K)	Relative axial location (x/L) of the end of the Kth fuel zone of the Ith axially varying fuel type table.
6-10, 16-20,	IZTYP(I,K)	Index of material type (from the order of entry on RODS.4), in fuel zone ending at ZEND(I,K) in the Ith etc. table of axial distance versus fuel type.

\*\*\* RODS.7 and RODS.8 are read in sequence, NRODTP times. \*\*\*

**Rod-to-Rod Thermal Connections**

This input is optional, and is read only if NRCON[RODS.1] is greater than zero. It is used to specify heat transfer paths due to direct connection between rods. It is assumed that the contact conductance and geometry factor are the same for all rods in contact with each other. This input is entered on RODS.9. The specific rods that are in contact with other rods must be identified on RODS.10. This input is read for all NRCON[RODS.1] rods. Up to 6 rods may be in contact with a given rod. All rods in contact with a given rod must be identified; there is not automatic reciprocity in this input.

**CAUTION:** This option is still under development, and needs additional validation and verification. It is not recommended for general use. If it is absolutely necessary for a specific problem, the user is advised to perform appropriate validation by comparison to experimental data, to evaluate the reliability and accuracy of the code predictions for the conditions.

**RODS.9** RCON,RRDIM Read only if NRCON > 0 on RODS.1  
 FORMAT(2E10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	RCON	Contact resistance between rods, (h-ft- o F/Btu).
11-20	RRDIM	Geometry factor of thermal connections for rod-to-rod contact, defined as the ratio of one-half the cladding node thickness over the perimeter of the rod circumferential node. That is,

$$RRDIM = \frac{\frac{1}{2} T_{CLAD}}{\frac{\pi d_{rod}}{NTHETA}}$$

**RODS.10** NTH,NRR(NTH),(NCLAD(NTH,I),I=1,6) Read only if NRCON[RODS.1] > 0  
 FORMAT(8I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NTH	Index number of connection for rod-to-rod contact reesistance (from 1 to NRCON).
6-10	NRR(NTH)	Index number of the NTH rod in contact with other rods.
11-15, 16-20, 21-25, 26-30, 31-35, 36-40	NCLAD(NTH,I)	Index numbers of the NTHETA rods that are in contact with rod NRR(NTH).

\*\*\*\* RODS.10 is read NRCON[RODS.1} times. \*\*\*\*

### Rod-to-Slab Thermal Connections

This input is optional, and is read only if NSCON[RODS.1] is greater than zero. It is used to specify heat transfer paths due to direct contact between rods and structures modeled with slabs.

**\*\*\*CAUTION:** As with the rod-to-rod contact conductance, this option is still under development, and needs additional validation and verification. It should be used only when absolutely necessary to model special problems. In such cases, the user is advised to perform appropriate validation by comparison to experimental data, to evaluate the reliability and accuracy of the code predictions for the conditions.

**RODS.11** WCON,WRDIM Read only if NSCON > 0 on RODS.1  
 FORMAT(2E10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	WCON	Contact resistance between rods and walls, (h-ft-°F/Btu)
11-20	WRDIM	Geometry factor of thermal connections for rod-to-wall contact, defined as the ratio of one-half the wall surface node thickness over the perimeter of the wall mode.

**RODS.12** NTH,NRW(NTH),(NWALL(NTH,I),I=1,6) Read only if NSCON[RODS.1] > 0  
 FORMAT(8I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NTH	Index number of connection for rod-to-wall contact resistance (from 1 to NSCON).
6-10	NRW(NTH)	Index number of the NTH wall node in contact with rod(s).
11-15, 16-20, 21-25, 26-30, 31-35, 36-40	NWALL(NTH,I)	Index numbers of the rods that are in contact with wall node NRW(NTH).

\*\*\*\* RODS.12 is read NSCON[RODS.1] times. \*\*\*\*

### 6.2.6 Group SLAB--Solid Structure Geometry

This group defines the geometry, material properties, and heat transfer connections for the solid structures that comprise a cask or other physical system being analyzed. It can be used to model such things as the basket that holds the assemblies, the various shells comprising the cask body, and other internal structures of the cask. Conduction heat transfer is modeled within and between slab nodes, and slab nodes can also exchange energy with the fluid by conduction and convection. Radiative heat transfer with the fuel rods and with other slab nodes can also be modeled.

Solid structure nodes can have thermal connections to the fluid in the channels, and to other solid structure nodes. Slab nodes may have connections only to other solid nodes, or only to channels, or to both solid nodes and channels, depending on the geometry of the system. This input is extremely flexible, allowing the user to model almost any reasonable geometry configuration. A number of examples of the use of this input are illustrated in Section 7.0.

The user must specify the total number of slab nodes with the value for NWK on SLAB.1. The thermal connections to the fluid or other slab nodes are indexed by type, where a type is defined by the effective thermal resistance of the connection. The number of types of solid-to-solid

connections, (NKSS), and the number of types of solid-to-fluid connections (NKSF), are specified by the input on SLAB.1. Connections of the same type have the same thermal resistance, where the thermal resistance of a connection is calculated as

$$R = \frac{F_{G,A} + F_{G,B}}{k} \quad (6.1)$$

where  $F_{G,A}$  = geometry factor for node A of the pair  
 $F_{G,B}$  = geometry factor for node B of the pair  
 $k$  = thermal conductivity (Btu/s-ft-°F) of the material in the conduction path between the centers of nodes A and B

For connections between adjacent slab nodes, the geometry factor is defined as

$$F_G = \frac{W}{L} \quad (6.2)$$

where  $W$  = distance from the solid node center to the edge facing the adjacent node  
 $L$  = length of the solid node at the face in contact with the adjacent node (NOTE: this length is perpendicular to the axial direction)

For thermal connections between slab nodes and fluid nodes (i.e., channels), the geometry factor is

$$F_G = \frac{W}{P} \quad (6.3)$$

where  $W$  = distance from the solid node center to the node edge facing the channel  
 $P$  = perimeter of the slab node face that sees the channel

Material properties for a slab node are normally considered to be axially uniform. The user does have the option, however, of specifying axially varying material properties for specific nodes. The number of nodes affected when this option is used is defined by the value entered for NMAT on SLAB.1.

**SLAB.1** AGROUP,NKSS,NKSF,NWK,NMAT  
 FORMAT(A4,1X,13I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter SLAB.
6-10	NKSS	Number of solid-to-solid thermal connection types.
11-15	NKSF	Number of solid-to-fluid thermal connection types.
16-20	NWK	Number of solid structure nodes.
21-25	NMAT	Number of slab nodes with axially varying material types.

## Solid-to-Solid Thermal Connection Types

A total of NKSS[SLAB.1] thermal connection types must be defined by the input on SLAB.2. The user has two options for specifying the thermal resistance of a connection type. In the first option, the user can define the individual geometry factors,  $F_{G,A}$  and  $F_{G,B}$ , for the connection, using Eq. (6.2). These geometry factors for the two adjacent nodes are entered in RDIMA(IC) and RDIMB(IC) on SLAB.2. The convention for this input is that RDIMA(IC) is the geometry factor for the lower-numbered node of any pair of nodes connected by thermal connection type IC, and RDIMB(IC) is the geometry factor for the higher-numbered node of the pair.

When the geometry factors are specified using RDIMA(IC) and RDIMB(IC), the code automatically calculates the resistance for a given node of that type using the specified geometry factors and the thermal conductivities of the materials comprising the adjacent nodes. Thermal conductivity is determined from each node's material property type number, specified by input on SLAB.3.

Alternatively, under the second option, the resistance can be specified directly as the value of R calculated using Eq. (6.1). This value is entered in variable RSER(IC) on SLAB.2. In most cases, it will be more convenient to enter the individual geometry factors for a connection type, using RDIMA(IC) and RDIMB(IC), than to calculate RSER(IC). In order to enter RSER(IC), the user must first calculate both geometry factors for each node, then sum them and divide by the average thermal conductivity of the connection path. If the adjacent nodes are of different materials, the user must calculate an appropriate average thermal conductivity for the combined materials. In general, the option for RSER(IC) should be used only for connections between uniform nodes that are composed of the same material.

The heat transfer between slab nodes is calculated assuming perfect thermal contact, unless the user specifies otherwise. In a situation where there is significant contact resistance, the gap resistance for the connection type must be defined by the user. This is done on SLAB.2 by entering a gap resistance in RPAR(IC) and a heat transfer area for the gap in ARAD(IC). The gap resistance specified in RPAR(IC) is defined as

$$R_{\text{gap}} = \frac{W}{L} \left( \frac{1}{k} \right) \quad (6.4)$$

where  $W$  = width of gap between the two slab nodes  
 $L$  = length of the node face that sees the adjacent slab node, (NOTE: this length is perpendicular to the axial direction)  
 $k$  = thermal conductivity (Btu/s-ft-°F) of the material in the gap

The area for heat transfer across the gap is defined as follows;

$$A_{\text{rad}} = L \Delta X \quad (6.5)$$

where  $L$  = length of the node face that sees the adjacent slab node (NOTE: this length is perpendicular to the axial direction)  
 $\Delta X$  = axial node length (as determined by input in group CHAN, see CHAN.1, CHAN.3, and CHAN.4)

Since  $\Delta X$  is determined by other input, and may be axially varying, the value entered for ARAD(IC) consists only of the  $L$  term in Eq. (6.5) above. The code calculates the correct area for a given thermal connection between nodes, using the value of ARAD(IC) and the axial node length.

If radiative heat transfer across the gap is important, it can be included by defining appropriate emissivities in EMA(IC) and EMB(IC) for the two surfaces that see each other across the gap. As with the input for RDIMA and RDIMB, the convention is that EMA is the emissivity of the lower-numbered node, and EMB is the emissivity of the higher-numbered node of any pair of nodes of this type.

**SLAB.2** (IC,RDIMA(IC),RDIMB(IC),RSER(IC),RPAR(IC),EMA(IC),EMB(IC),ARAD(IC),  
 IC=1,NKSS[SLAB.1]) Read only if NKSS > 0 SLAB.1  
 FORMAT(I5,4E10.0,3F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IC	Identification number of a solid-to-solid thermal connection type. (NOTE: must be entered in sequence from 1 to NKSS.)
6-15	RDIMA(IC)	Geometry factor, (as defined in Eq. (6.2)), for the lower-numbered node of any pair of nodes connected by thermal connection type IC. (NOTE: not used if RSER(IC) is specified.)
16-25	RDIMB(IC)	Geometry factor, (as defined in Eq. (6.2)), for the higher-numbered node of any pair of nodes connected by thermal connection type IC. (NOTE: not used if RSER(IC) is specified.)
26-35	RSER(IC)	Thermal resistance (s-ft-°F/Btu), (as defined in Eq. (6.1)), between the centers of adjacent nodes connected by thermal connection type IC. (NOTE: Not used if RDIMA(IC) and RDIMB(IC) are specified.)
36-45	RPAR(IC)	Thermal resistance (s-ft-°F/Btu) of the gap between adjacent solid structure nodes connected by thermal connection type IC, as defined in Eq. (6.4). (NOTE: this is optional input. Enter zero if there is no significant gap resistance between nodes of this type.)

46-50	EMA(IC)	Surface emissivity of node A for radiative heat transfer across the gap between any pair of nodes connected by thermal connection type IC. (Enter zero if RPAR(IC) is zero.)
51-55	EMB(IC)	Surface emissivity of node B for radiative heat transfer across the gap between any pair of nodes connected by thermal connection type IC. (Enter zero if RPAR(IC) is zero.)
56-60	ARAD(IC)	Length of node (in.) on the face looking at the gap, for calculation of heat transfer area for energy exchange across the gap between adjacent nodes connected by thermal connection type IC. (Enter zero if RPAR(IC) is zero.)

\*\*\* SLAB.2 is read in NKSS[SLAB.1] times. \*\*\*

### Solid Structure Node Description

This input describes each of the NWK solid nodes in the problem, identifies its material type, and specifies all interconnections for heat transfer with other solid nodes. The specification of the heat transfer connections between slabs follows the same convention as that for specifying connections between channels for lateral flow. A connection is identified only once by defining it from the lower-numbered node to the higher-numbered node of the pair. For example, if node 2 connects to node 7, the connection is identified only on the input line for node 2. This connection should *not* be repeated on the input line for node 7.

**SLAB.3** (KW,MATTYP(KW),WALLXC(KW),QSLAB(KW),NAXK(KW),NSLAB,(KWAL(L,KW),  
ICON(L,KW),L=1,NSLAB)) Read only if NWK > 0 on SLAB.1.  
FORMAT(2I5,2F5.0,12I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	KW	Identification number of a solid structure node. (NOTE: must be entered in sequence, from 1 to NWK.)
6-10	MATTYP(KW)	Material type identification for node KW:  If NMAT = 0 on SLAB.1; MATTYP(KW) must correspond to a material type specified on PROP.3. If NMAT > 0 on SLAB.1, and; if slab KW has constant material properties; MATTYP(KW) is specified as a material type defined on PROP.3. if slab KW has axially varying properties; MATTYP(KW) is specified as the index of an axially varying material property



profile, to be defined on SLAB.8 and SLAB.9. (NOTE: in this case, MATTYP(KW) must be greater than NSPROP[PROP.1].)

11-15	WALLXC(KW)	Solid node cross-sectional area (in. <sup>2</sup> ) in the axial direction, for axial conduction (enter zero if axial conduction will not be calculated for this problem).
16-20	QSLAB(KW)	Solid node volumetric heat generation rate (Btu/h-ft <sup>3</sup> ).
21-25	NAXK(KW)	Identification number of the axial heat generation profile table [specified on OPER.16] to be applied to QSLAB(KW) (default is 1).
26-30	NSLAB	Number of thermal connections to adjacent solid nodes that have index numbers greater than KW.
31-35, 41-45, etc.	KWAL(L,KW)	Identification number of the Lth solid node adjacent to node KW, where KWAL(L,KW) > KW.
36-40, 46-50, etc.	ICON(L,KW)	Identification number of the thermal connection type for the connection between node KW and node KWAL(L,KW). (NOTE: This must correspond to a thermal connection type identified on SLAB.2.)

A maximum of five pairs of KWAL,ICON values may be specified on SLAB.3 for given slab node. If NSLAB is greater than five, continue the connection input on SLAB.4 to define all connections to node KW before going on to the input for the next slab node.)

**SLAB.4** (KWAL(L,KW),ICON(L,KW),L=6,NSLAB)) Read only if NSLAB > 6 on SLAB.3.  
FORMAT(30X,10I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-30		BLANK
31-35, 41-45, etc.	KWAL(L,KW)	Identification number of the Lth solid node adjacent to node KW, where KWAL(L,KW) > KW.
36-40, 46-50, etc.	ICON(L,KW)	Identification number of the thermal connection type for the connection between node KW and node KWAL(L,KW). (NOTE: This must correspond to a thermal connection type identified on SLAB.2.)

\*\*\* SLAB.4 is read with five pairs of entries per line. \*\*\*

\*\*\* SLAB.3 (with SLAB.4, if needed) is read NWK[SLAB.1] times. \*\*\*

## Solid-to-Fluid Thermal Connection Types

This input is required only if solid structure nodes have heat transfer connections to flow channels. If NKSF[SLAB.1] is zero, this input is not read. As with the solid-to-solid thermal connections, the solid-to-fluid connections are defined by type. Connections of the same type have the same thermal resistance, where the thermal resistance of a connection is calculated as

$$R = \frac{F_G}{k} \quad (6.6)$$

where  $F_G$  = geometry factor for the solid node facing the channel (as defined in Eq. (6-3) above)  
 $k$  = thermal conductivity (Btu/s-ft-°F) of the solid node.

The user has two options for specifying the thermal resistance of a connection type. In the first option, the user can define the individual geometry factor,  $F_G$ , for the connection, using Eq. (6.3). This geometry factor is entered in RDIMF(IC) on SLAB.5. Alternatively, using the second option, the resistance can be specified directly as the value of R calculated using Eq. (6.6). This value is entered in variable RWAL(IC) on SLAB.2.

**SLAB.5** (IC,RDIMF(IC),RWAL(IC),WID(IC),IC=1,NSKF[SLAB.1])  
 Read only if NKSF > 0 on SLAB.1  
 FORMAT(I5,2E5.0,E10.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IC	Solid-to-fluid thermal connection type identification number. (NOTE: must be entered in sequence, from 1 to NKSF.)
6-10	RDIMF(IC)	Geometry factor, as defined in Eq. (6-3), from the center of the solid node to the surface that faces the channel (not used if RWAL(IC) is specified).
11-15	RWAL(IC)	Thermal resistance (s-ft-°F/Btu) (as defined in Eq. (6-6)), from the center of the solid node to the node surface that faces the channel (not used if RDIMF(IC) is specified).
16-25	WID(IC)	Perimeter of solid node that faces the channel (in.).

\*\*\* SLAB.5 is read NKSF[SLAB.1] times. \*\*\*

## Solid-to-Fluid Connection Description

This input specifies the connections between solid nodes and fluid channels. It is read only if NKSF on SLAB.1 is greater than zero. If there are no connections between solid nodes and the fluid, this input is not needed. Connections are read only for nodes that see fluid nodes. In many cases,

the number of nodes with fluid connections will be less than NWK, the total number of solid nodes. To signal the end of this input, zero must be entered for KW on SLAB.6, after all nodes with fluid connections have been identified.

**SLAB.6** (KW,NCHN,(IASSM(L,KW),IKW(L,KW),ICF(L,N),L=1,NCHN))  
 Read only if NSKF > 0 on SLAB.1  
 FORMAT(2I5,4(3I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	KW	Identification number of solid node with thermal connection(s) to channel(s). NOTE: enter zero for KW to terminate this input when all solid-to-fluid connections have been identified.
6-10	NCHN	Number of thermal connections to adjacent channels for node KW.
11-15, 26-30, etc.	IASSM(L,KW)	Index number of assembly containing the Lth channel with thermal connection to node KW.
16-20, 31-35 etc.	IKW(L,KW)	Identification number of Lth channel with thermal connection to node KW.
21-25, 36-40 etc.	ICF(L,KW)	Identification number of solid-to-fluid thermal connection type for the connection between node KW and channel IKW(L,KW). (NOTE: This must correspond to a thermal connection type specified on SLAB.5.)

A maximum of four sets of values of IASSM, IKW, and ICF may be specified on SLAB.6. If NCHN is greater than four, continue the connection input on SLAB.7 to define all connections to node KW before going on to the input for the next node with connections to the fluid.)

**SLAB.7** (IASSM(L,KW),IKW(L,KW),ICF(L,N),L=5,NCHN))  
 Read only if NCHN > 4 on SLAB.5  
 FORMAT(15X,4(3I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10		BLANK
11-15, 26-30, etc.	IASSM(L,KW)	Index number of assembly containing the Lth channel with thermal connection to node KW.
16-20, 31-35 etc.	IKW(L,KW)	Identification number of Lth channel with thermal connection to node KW.

21-25, ICF(L,KW) Identification number of solid-to-fluid thermal  
 36-40 connection type for the connection between node KW  
 etc. and channel IKW(L,KW).  
 (NOTE: This must correspond to a thermal connection type  
 specified on SLAB.5.)

\*\*\* SLAB.7 is read with 4 sets of (IASSM, IKW, ICF) entries per line. \*\*\*  
 \*\*\* SLAB.6 (with SLAB.7, if needed), is read as many times as necessary \*\*\*  
 \*\*\* to define the connections between solid nodes and the fluid nodes. \*\*\*

The input on SLAB.6 is terminated by entering a line with zero entered for KW, after all nodes with connections to flow channels have been identified.

### Axial Variation of Slab Material

This option allows the user to specify different material properties in a slab node at different axial locations. The material type flag MATTYP(KN) on SLAB.3 is set to the profile index number for any nodes that are to use the axially varying profile. The input on SLAB.8 and SLAB.9 is used to define the profiles, by specifying which material properties (defined by input in group PROP) will be used at which axial locations for each profile. The user must specify NMAT profiles.

**SLAB.8** MAT,NSET Read only if NMAT > 0 on SLAB.1 FORMAT (2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	MAT	Axially varying material type profile identification number. (Used in SLAB.3 to identify axially varying material type profile for specific nodes. Must be greater than NSPROP on PROP.1.)
6-10	NSET	Number of regions with different material properties in this profile.

**SLAB.9** (K1,K2,MATJ,N=1,NSET[SLAB.6]) Read only if NMAT > 0 on SLAB.1  
 FORMAT (4(3I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 16-20, 46-50	K1	Node number at the beginning of the Nth region of axially varying material profile MAT. (NOTE: the 31-35, first entry for K1 must be node 2.)
6-10, 21-25, 36-40,	K2	Number of axial nodes in the Nth region of profile MAT. (NOTE: The total number of nodes for all NSET regions in profile MAT must sum to the total number 51-55 of axial nodes, NDX[CHAN.1].)

11-15	MATJ	Identification number of material type for the Nth
26-30,		region of profile MAT.
		(NOTE: This must correspond to a material type specified in
41-45,		PROP.3.)
56-60		

Four sets of entries (K1,K2,MATJ) are read on each line of SLAB.9. This line is repeated until all NSET regions are read for profile MAT.

\*\*\*SLAB.8 and SLAB.9 are read NMAT[SLAB.1] times.\*\*\*

### 6.2.7 Group RADG--Radiative Heat Transfer Exchange Factors

This group describes the exchange factors for radiative heat transfer within assemblies. Exchange factors are dimensionless geometry parameters that define how well a given surface can "see" another surface for radiative heat transfer. The radiative heat transfer exchange factor groups are specified by type, where a type is defined by the geometry and emissivities of the surfaces in the assembly. A radiative heat transfer exchange factor group type consists of an array of gray body view factors for every surface in the assembly as it exchanges thermal energy by radiation with every other surface in the assembly. The emissivities of the surfaces are also included in the definition of the group type.

There are two ways the user can define radiative heat transfer exchange factor group types for a given case. The gray body view factors and emissivities for some or all groups can be read from a separate input file, on logical unit 10 (opened with the file name *tape10*). This file is generated by the auxiliary program RADX-1 or RADGEN, and must be consistent with the geometry input specified in groups RODS and CHAN for the problem (see Section 6.3 for a discussion of the Cycle 2 version of the RADGEN code, and its input instructions).

The user also has the option of specifying black body view factors for assemblies that do not have gray body view factors supplied on *tape10*. The code will automatically calculate the corresponding gray body exchange factors from the black body view factors. Both options can be used within the same problem, but in different assemblies. Typically, a *tape10* file generated by the RADGEN code will be used to obtain the view factors for the assemblies containing rods, and the option for user-specified black body view factors will be used for assemblies modeling open regions of the system, such as the basket region, which do not contain rods.

When view factors generated by RADX-1 or RADGEN are used for square assemblies containing unconsolidated rods on a square pitch, the walls of the assembly must be modeled with eight slab nodes, two on each face of the assembly enclosure. For assemblies with consolidated rods on a triangular pitch, the walls of the assembly must be modeled with four slab nodes, one for each face of the enclosure. For assemblies where black body view factors are specified by the user, however, the walls can be modeled with any number of nodes. Only one radiative heat transfer group type can be specified within a given assembly, but a given solid structure node may belong to more than one group, if it has surfaces facing more than one assembly.

**RADG.1** AGROUP,NASSR,NT10,NRADG  
 FORMAT(A4,1X,3I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter RADG.
6-10	NASSR	Number of assemblies for which radiative heat transfer groups are to be specified.
11-15	NT10	Number of radiative heat transfer groups to be read from logical unit 10 (maximum of 6; these radiative heat transfer exchange factor groups are calculated using a program such as RADGEN; see Section 6.3.)
16-20	NRADG	Number of radiative heat transfer exchange factor group types for which the gray body view factors will be calculated from input of black body view factors and emissivities.

**Black Body Radiative Exchange Factor Types**

This input is optional and is read only if NRADG on RADG.1 is greater than zero. It is used to define radiative heat transfer view factors for enclosures that cannot be defined using program RADGEN; see Section 6.3.

**RADG.2** NRAD,NSURF Read only if NRADG > 0 on RADG.1  
 FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NRAD	Radiative heat transfer exchange factor group type identification number. (NOTE: groups must be numbered sequentially, from 1 to NRADG.)
6-10	NSURF	Number of surfaces in radiative heat transfer exchange factor group type NRAD.

**Black Body Radiative Exchange Factors**

For each radiative exchange factor group type NRAD specified on RADG.2, the user must describe the geometry of the enclosure, and the black body view factors for all surfaces. These factors are area-weighted and can be calculated using the Hottel crossed-string correlation method (Cox 1977), such that

$$A_n F_{nm} = A_m F_{mn} \tag{6.7}$$

where  $A_n, A_m$  = area of surfaces n and m, respectively  
 $F_{nm}$  = gray body view factor from surface n to m  
 $F_{mn}$  = gray body view factor from surface m to n.

Refer to Section 6.3 for a discussion of the black body view factors, and the auxiliary code, RADGEN, used to calculate grey body view factors for rod arrays.

**RADG.3** IS,AREAS(NR,IS),EMX(IS),(ISX,FVW(NR,IS,ISX),ISX=1,6),NFACES  
 Read only if NRADG > 0 on RADG.1.  
 FORMAT(I5,2F5.0,6(I5,F5.0),I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IS	Identification number for a surface in radiative heat transfer group type NRAD. (NOTE: Surfaces of the group type must be numbered sequentially from 1 to NSURF for this input; the number is a generic place-holder for nodes with surfaces in assemblies that use radiative heat transfer group type NRAD; IS does not refer to a specific slab node number. Slab nodes of assemblies of group type NRAD are assigned to specific surfaces of the group type by the input on RODS.11.)
6-10	AREAS(NR,IS)	Perimeter (in.) of surface IS in group NRAD.
11-15	EMX(IS)	Emissivity of surface IS in group NRAD.
16-20, 26-30, etc.	ISX	Index number of a surface that can exchange thermal radiation with IS in group NRAD.
21-25, 31-35, etc.	FVW(NR,IS,ISX)	Black body view factor from surface IS to surface ISX.
76-80	NFACES	Total number of surfaces that can exchange energy by thermal radiation with surface IS.

Up to six pairs of values for ISX,FVW(NR,IS,ISX) can be entered on RADG.3; if NFACES is greater than six, the remaining input for surface is read on RADG.4 before going on to the input for the next surface.

**RADG.4** (ISX,FVW(NR,IS,ISX),ISX=7,NSURF[RADG.2])  
 Read only if NSURF > 6 on RADG.2.  
 FORMAT(15X,6(I5,F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15		blank
16-20, 26-30, etc.	ISX	Index number of a surface that can exchange thermal radiation with IS in group NRAD.
21-25, 31-35, etc.	FVW(NR,IS,ISX)	Black body view factor from surface IS to surface ISX.

Up to six pairs of values for ISX,FVW(NR,IS,ISX) may be entered on RADG.4. RADG.4 is read as many times as necessary to define all view factors to all surfaces seen by surface IS in group NRAD.

\*\*\* RADG.2 and RADG.3, (with RADG.4, if required), are read \*\*\*  
 \*\*\* sequentially, NRADG[RADG.1] times. \*\*\*

Input lines RADG.5 through RADG.9 are read from logical unit 10, which can be generated using the auxiliary program RADGEN. The input lines are listed here for information only. They are not supplied in the normal input stream.

**RADG.5** NROWS,PDR,EMW,EMR  
 Read from I/O unit 10, only if NT10 > 0 on RADG.1.  
 FORMAT(I5,3E5.0,I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NROWS	Number of rows of rods in the rod bundles modeled by this radiative heat transfer group.
6-10	PDR	Nominal pitch to diameter ratio for the rod bundle modeled by this radiative heat transfer group.
11-15	EMW	Emissivity of the slab nodes modeling the walls of the assembly for this radiative heat transfer group.
16-20	EMR	Emissivity of the rods for this radiative heat transfer group.



**RADG.6** MM1,MM2,MM3,MM4  
 Read from I/O unit 10, only if NT10 > 0 on RADG.1.  
 FORMAT(4I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	MM1	Total number of surfaces in radiative heat transfer group MM2.
6-10	MM2	Identification number for this radiative heat transfer group. (NOTE: $1 \leq MM2 \leq 6$ )
11-15	MM3	Flag for number of surfaces on a rod (NOTE: must correspond to the value for NTHETA entered on RODS.1).
16-20	MM4	Number of wall surfaces in this radiative heat transfer group (must be 8 for unconsolidated rod bundle, and 4 for consolidated rod bundle with triangular pitch in a square enclosure).

**RADG.7** (EMV(MM2,N),N=1,MM4)  
 Read from I/O unit 10, only if NT10 > 0 on RADG.1.  
 FORMAT(8E15.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15, 16-30, 21-45, etc.	EMV(MM2,N)	Emissivity of the Nth slab node enclosing an assembly with radiative heat transfer group MM2. (NOTE: Default is EMW[RADG.5] for all wall nodes; if wall emissivities are uniform, this input can be entered as a blank line.)

**RADG.8** (RRATIO(MM2,N),N=1,MM4)  
 Read from I/O unit 10, only if NT10 > 0 on RADG.1.  
 FORMAT(8E15.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15, 16-30, 31-45, etc.	RRATIO(MM2,N)	Ratio of the perimeter of the Nth node of the assembly wall to the rod perimeter, for radiative heat transfer group MM2.

**RADG.9** (FVn(N,M),M=1,MM1),N=1,MM1)  
 Read from I/O unit 10 only if NT10 > 0 on RADG.1.  
 FORMAT(8E15.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15, 16-30, 31-45, etc.	FVn(N,M)	Gray body exchange factor from surface N to surface M for radiative heat transfer group MM2.

\*\*\* RADG.9 is read for MM1 sets of exchange factors. \*\*\*

RADG.6 through RADG.9 are read sequentially NT10 times from logical unit 10.

This is the end of the input read from logical unit 10.

### Assembly Radiative Heat Transfer Group Information

This input line assigns the appropriate radiative heat transfer group type to each assembly that has radiative heat transfer.

**RADG.10** IASS,ITYPR(IASS),MSID(IASS),(KSIDE(IASS,L),L=1,12)  
 FORMAT(3I5,12(I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IASS	Identification number of an assembly with radiative heat transfer.
6-10	ITYPR(IASS)	Radiative heat transfer group type: <ul style="list-style-type: none"> <li>- for a radiative group type read from logical unit 10, enter a positive number (must correspond to a group type identified on RADG.6).</li> <li>- for a radiative group calculated from user specified black body view factors, enter the negative of the group identification number (must correspond to a group type identified on RADG.2).</li> </ul>
11-15	MSID(IASS)	Number of slab nodes with surfaces in radiative exchange factor group ITYPR(IASS). (NOTE: must be the same as NSURF on RODS.2 for group type ABS[ITYPR(IASS)].)
16-20, 21-25, etc.	KSIDE(IASS,L)	Identification number of the slab node corresponding to the Lth surface of radiative exchange factor group ABS[ITYPR(IASS)].

Up to 12 values for KSIDE can be entered on RADG.10; if MSID(IASS) is greater than 12, the remaining input for assembly IASS is read on RADG.11 before going on to the input for the next assembly.

**RADG.11** (KSID(L),L=13,MSID(IASS)) Read only if MSID(IASS)[RADG.10] > 12  
 FORMAT(15X,12(I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15		BLANK
16-20, 21-25, etc.	KSIDE(IASS,L)	Identification number of the slab node corresponding to the Lth surface of radiative heat transfer group ITYPR(IASS).

\*\*\* RADG.10 (with RADG.11 if MSID(IASS) > 12) is read NASSR[RADG.1] times. \*\*\*

### 6.2.8 Group HEAT--Heat Transfer Correlations

This group defines the heat transfer coefficient correlations for convection and conduction heat exchange between the fuel rods and the fluid only. Heat transfer between the fluid and the structures represented by slab nodes is modeled using thermal connections, as defined in group SLAB (see Section 6.2.6. The correlations are identified by type number, which must correspond to the values specified for NFLMC(NASS) on CHAN.6. The option for the single-phase turbulent mixing model in COBRA-SFS can also be selected by input in this group.

**HEAT.1** AGROUP,NHEAT,NSCBC,NFCON,NTHEAT,NFHEAT  
 FORMAT(A4,1X,5I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter HEAT.
6-10	NHEAT	Number of heat transfer correlation sets to be read in on HEAT.2.
11-15	NSCBC	Flag for single-phase turbulent mixing model: = 0; no turbulent mixing (default) = 1; $W_T = AMIX * (S_K \bar{G})$ = 2; $W_T = AMIX * (Re ** BMIX) * (S_K \bar{G})$ = 3; $W_T = AMIX * (Re ** BMIX) * (\bar{D} \bar{G})$ = 4; $W_T = AMIX * (Re ** BMIX) * (S_K / l_k) (\bar{D} \bar{G})$

where

- $W_T$  = turbulent crossflow.
- Re = Reynolds number (based on axial velocity)
- $\bar{G}$  = average axial mass flux of the two channels connected by lateral flow connection k
- $S_K$  = lateral gap width

$\bar{D}$  = average hydraulic diameter of the two channels connected by lateral flow connection  
 $\ell_K$  = transverse momentum control volume length for flow connection k  
 AMIX, BMIX = user-defined constants; see HEAT.3.

16-20 NFFCON  
 Flag for radial conduction in the fluid:

= 0; no conduction in the fluid.

= 1; fluid conduction between channels through the gap connections.

21-25 NTHEAT  
 Number of lateral convection heat transfer correlation sets to be read in on HEAT.6.

26-30 NFHEAT  
 Number of axial free convection heat transfer correlations to be read in on HEAT.6.

### Axial Flow Forced Convection Heat Transfer Correlations

Forced convection heat transfer correlations are specified in the code using the standard formula,

$$H = (a_1 Re^{a_2} Pr^{a_3} + a_4) \frac{k}{D_c} \quad (6.8)$$

where  $a_1, a_2, a_3, a_4$  = empirical coefficients  
 $Re$  = Reynolds number  
 $Pr$  = Prandtl number  
 $k$  = thermal conductivity of the fluid  
 $D_c$  = channel hydraulic diameter

The convention in the code is to enter heat transfer correlations in pairs, with one set of coefficients for laminar flow and one set for turbulent flow, such that Eq. (6.8) becomes

$$H_{\text{laminar}} = (a_{1l} Re^{a_{2l}} Pr^{a_{3l}} + a_{4l}) \frac{k}{D_c} \quad (6.9)$$

$$H_{\text{turbulent}} = (a_{1t} Re^{a_{2t}} Pr^{a_{3t}} + a_{4t}) \frac{k}{D_c}$$

The local heat transfer coefficient for a node is evaluated as the maximum value obtained with the turbulent and laminar equations for the local Reynolds number and Prandtl number,

$$H_{\text{axial}} = \max(H_{\text{laminar}}, H_{\text{turbulent}})$$

The heat transfer correlations are numbered sequentially in the code, from 1 to NHEAT, in the order they are read in on HEAT.2. These sequence numbers must be used when specifying the values entered for variable NFLMC(NASS) on CHAN.6, when identifying the heat transfer correlation used by each assembly type.

**HEAT.2** (AH1(I),AH2(I),AH3(I),AH4(I),AHL1(I),AHL2(I),AHL3(I),AHL4(I),  
I=1,NHEAT) Read only if NHEAT > 0 on HEAT.1  
FORMAT(12F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AH1(I)	Reynolds number multiplier coefficient of Ith axial heat transfer correlation for turbulent flow (i.e., $a_1$ in Eq. [6.9]).
6-10	AH2(I)	Reynolds number exponent of Ith axial heat transfer correlation for turbulent flow (i.e., $a_2$ in Eq. [6.9]).
11-15	AH3(I)	Prandtl number exponent of Ith axial heat transfer correlation for turbulent flow (i.e., $a_3$ in Eq. [6.9]).
16-20	AH4(I)	Additive coefficient of Ith axial heat transfer correlation for turbulent flow (i.e., $a_4$ in Eq. [6.9]).
21-25	AHL1(I)	Reynolds number multiplier coefficient of Ith axial heat transfer correlation for laminar flow, (i.e., $a_{11}$ in Eq. [6.9]).
26-30	AHL2(I)	Reynolds number exponent of Ith axial heat transfer correlation for laminar flow (i.e., $a_{12}$ in Eq. [6.9]).
31-35	AHL3(I)	Prandtl number exponent of Ith axial heat transfer correlation for laminar flow (i.e., $a_{13}$ in Eq. [6.9]).
36-40	AHL4(I)	Additive coefficient of Ith axial heat transfer correlation for laminar flow (i.e., $a_{14}$ in Eq. [6.9]).

**HEAT.3** (AMIX(N),BMIX(N),N=1,MAXTYP) Read only if NSCBC > 0 on HEAT.1  
FORMAT(2E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AMIX(N)	Multiplicative coefficient for turbulent mixing correlation, of form specified by value of NSCBC on HEAT.1, to be applied to all assemblies of type N.
6-10	BMIX(N)	Reynolds number exponent for turbulent mixing correlation of form specified by value of NSCBC on HEAT.1, to be applied to all assemblies of type N. (NOTE: BMIX(N) is not used if NSCBC = 1.)

MAXTYP is the total number of assembly types (i.e., the number of unique values specified for ITYPA on CHAN.5).

\*\*\* HEAT.3 is read MAXTYP times, once for each assembly type, \*\*\*  
\*\*\* if NSCBC > 0 on HEAT.1. \*\*\*

## Radial Fluid Conduction

If NFCON = 1, radial heat conduction in the fluid will be calculated for the problem. Heat conduction through the fluid is treated as occurring between adjacent channels through the gap connections. The conduction path is defined by the centroid length of the lateral control volume, which is specified by user input in variable DIST(K) on CHAN.7. The input on HEAT.4 can be used to modify the conduction length for the lateral connections in a given assembly, to account for variations in the connection lengths not considered in the value entered for DIST.

**HEAT.4** (GK(N),N=1,MAXTYP) Read only if NFCON = 1 on HEAT.1.  
 FORMAT(16E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	GK(N)	Multiplier on the lateral control volume length for thermal conduction between channels in assemblies of type N: <ul style="list-style-type: none"> <li>- if GK(N) is specified as 1.0, the length specified in DIST on CHAN.7 for a particular gap connection on is used for the conduction length.</li> <li>- if GK(N) is specified as 0.0, fluid conduction is zero in assemblies of type N.</li> </ul>

MAXTYP is defined for this input as for HEAT.3; it is the number of unique assembly types.

Up to 16 values can be read on a line for HEAT.4. If there are more than 16 assembly types, this line is repeated as many times as necessary to specify a GK(N) value for each of MAXTYP assembly types.

## Lateral Flow Forced Convection Heat Transfer Correlations

Heat transfer correlations for forced convection due to lateral flow follow the same convention as the correlations for axial flow. Coefficients are read in for a turbulent and a laminar formulation,

$$H_T = \frac{k}{D_r} (a_{t1} Re_D^{a_{t2}} Pr^{a_{t3}} + a_{t4}) \quad (6.10)$$

$$H_L = \frac{k}{D_r} (a_{l1} Re^{a_{l2}} Pr^{a_{l3}} + a_{l4})$$

where  $a_{t1}, a_{t2}, a_{t3}, a_{t4}$  = empirical coefficients for turbulent correlation  
 $a_{l1}, a_{l2}, a_{l3}, a_{l4}$  = empirical coefficients for laminar correlation  
 $Re_D$  = Reynolds number, based on rod diameter  
 $Pr$  = Prandtl number

$k$  = thermal conductivity of the fluid  
 $D_r$  = rod nominal diameter.

The local heat transfer coefficient is evaluated as the maximum of the two;

$$H_{\text{lateral}} = \max(H_T, H_L)$$

These heat transfer correlations are numbered sequentially in the code, from 1 to NTHEAT, in the order they are read in on HEAT.5. The heat transfer correlation to be used in a particular lateral flow connection is determined by the value specified for NFLMC(IASS) on CHAN.6 for the assembly containing the lateral flow connection. The input for NFLMC(IASS) on CHAN.6 is thus made to do double duty, since it also specifies the heat transfer correlation for axial flow in the channels of the assembly, by identifying the index of a correlation defined by input on HEAT.2). The user must make certain that the index number entered on CHAN.6 in variable NFLMC(IASS) for an assembly type corresponds to the correct axial heat transfer correlation (from HEAT.2) for the channels *and* to the correct lateral heat transfer correlation (from HEAT.5) for the lateral connections between channels in the assembly.

**HEAT.5** (AHT1(I),AHT2(I),AHT3(I),AHT4(I),AHTL1(I),AHTL2(I),AHTL3(I),  
 AHTL4(I),I=1,NTHEAT[HEAT.1]) Read only if NTHEAT > 0  
 FORMAT(8F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AHT1(I)	Reynolds number multiplier coefficient of Ith lateral heat transfer correlation for turbulent flow (i.e., $a_{1t}$ in Eq. [6.10]).
6-10	AHT2(I)	Reynolds number exponent of Ith lateral heat transfer correlation for turbulent flow (i.e., $a_{2t}$ in Eq. [6.10]).
11-15	AHT3(I)	Prandtl number exponent of Ith lateral heat transfer correlation for turbulent flow (i.e., $a_{3t}$ in Eq. [6.10]).
16-20	AHT4(I)	Additive coefficient of Ith lateral heat transfer correlation for turbulent flow (i.e., $a_{4t}$ in Eq. [6.10]).
21-25	AHTL1(I)	Reynolds number multiplier coefficient of Ith lateral heat transfer correlation for laminar flow (i.e., $a_{1l}$ in Eq. [6.10]).
26-30	AHTL2(I)	Reynolds number exponent of Ith lateral heat transfer correlation for laminar flow (i.e., $a_{2l}$ in Eq. [6.10]).
31-35	AHTL3(I)	Prandtl number exponent of Ith lateral heat transfer correlation for laminar flow (i.e., $a_{3l}$ in Eq. [6.10]).
36-40	AHTL4(I)	Additive coefficient of Ith lateral heat transfer correlation for laminar flow (i.e., $a_{4l}$ in Eq. [6.10]).

\*\*\* HEAT.5 is read NTHEAT[HEAT.1] times. \*\*\*

## Axial Flow Free Convection Heat Transfer Correlations

Heat transfer correlations for free convection due to buoyancy-driven recirculating axial flows are specified in the form,

$$H_r = \frac{k}{D_r} (a_{r1} Ra^{a_{r2}}) \quad (6.11)$$

where  $a_{r1}$ ,  $a_{r2}$  = empirical coefficients  
 $Ra$  = local Rayleigh number of the flow, (defined as the Grashof number times the Prandtl number)  
 $k$  = thermal conductivity of the fluid  
 $D_r$  = rod diameter.

These heat transfer correlations are numbered sequentially in the code, from 1 to NFHEAT, in the order they are read in on HEAT.6. The heat transfer correlation to be used for free convection in a given assembly is determined by the value specified for NFLMC(IASS) on CHAN.6. The input for NFLMC(IASS) on CHAN.6 is thus made to do triple duty, since it also specifies the heat transfer correlations for axial and lateral forced convection in the assembly. The user must make certain that the index number entered on CHAN.6 in variable NFLMC(IASS) for an assembly type corresponds to the correct axial heat transfer correlation (from HEAT.2) for the channels, *and* to the correct lateral heat transfer correlation (from HEAT.5) for the lateral connections between channels in the assembly, *and* to the correct free convection heat transfer correlation (from HEAT.6).

**HEAT.6** (AHF1(I),AHF2(I),I=1,NFHEAT[HEAT.1]) Read only if NFHEAT > 0  
FORMAT (2F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AHF1(I)	Rayleigh number multiplier coefficient of Ith axial heat transfer correlation for free convection (i.e., $a_{r1}$ in Eq. [6.11]).
6-10	AHF2(I)	Rayleigh number exponent of Ith axial heat transfer correlation for free convection (i.e., $a_{r2}$ in Eq. [6.11]).

\*\*\* HEAT.6 is read NFHEAT[HEAT.1] times \*\*\*

### 6.2.9 Group DRAG--Friction Factors and Loss Coefficients

This group describes the axial pressure losses due to wall friction and local obstructions in the flow field. The optional network model can also be specified, by setting the flag NETWK on DRAG.1 (refer to Section 5.1.2.2 for a discussion of the network model).



**DRAG.1** AGROUP,NFRICT,NOLC,NRAMP,NLCFF,NLCFP,NETWK,NOGRP,  
NBLOCK,NVISCW,NFTRAN FORMAT(A4,1X,10I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter DRAG.
6-10	NFRICT	Number of friction factor correlation sets to be read in (default is 1).
11-15	NOLC	Number of assembly types for which loss coefficients are to be specified.
16-20	NRAMP	Number of iterations over which the loss terms are to be ramped into the solution (default is 1).
21-25	NLCFF	Number of loss coefficient versus Reynolds number forcing functions.
26-30	NLCFP	Number of points in the loss coefficient versus Reynolds number forcing functions.
31-35	NETWK	Flag for optional pressure drop network model:  = 0; not used.  = 1; pressure drop will be calculated for user-specified assembly network.
36-40	NOGRP	Number of assembly groups for the network model (used only if NETWK > 0).
41-45	NBLOCK	Number of axial locations where axial flow blockages occur.
46-50	NVISCW	Flag for hot wall correction to viscosity in friction factor correlation:  = 0; no hot wall correction.  = 1; apply hot wall correction to channels with heated surfaces.
51-55	NFTRAN	Number of lateral flow fraction loss correlation sets to be read in on DRAG.13.

## Friction Factor Correlations

Wall friction losses are described using the Darcy friction factor correlation form. This is defined in terms of the pressure drop such that the axial frictional pressure gradient is given by

$$\frac{dP}{dX} = \frac{f|m|m}{2 g_c D_h \rho A} \quad (6.12)$$

The friction factor,  $f$ , is determined from the Blasius relation,

$$f = aRe^b + c \quad (6.13)$$

Each correlation consists of a formulation with a set of turbulent and a set of laminar coefficients, of the form

$$f_T = aRe^b + cRe^d + e \quad (6.14)$$

$$f_L = a_L Re^{b_L} + c_L \quad (6.15)$$

The code evaluates the friction factor as the maximum of the turbulent and laminar values,

$$f = \max(f_T, f_L)$$

A total of NFRICT[DRAG.1] correlations are specified by input on DRAG.2. These correlations are numbered sequentially in the order they are read in on DRAG.2. These identification numbers are used when specifying the values for N on CHAN.7, in the geometry description for each channel of an assembly type. The default for N is type 1, and in most cases, this will be sufficient for all channels. However, the user can specify as many friction factor correlations as needed to model the various wall resistances in a given problem.

Alternatively, instead of specifying N for each channel on CHAN.7, a friction factor correlation can be specified for the assembly type, in the value for MDFLT on CHAN.6. The value for N can then be entered as zero, and all channels in assemblies of that type will be assigned the friction factor correlation identified in MDFLT on CHAN.6.

**DRAG.2** (AA(I),BB(I),CC(I),DD(I),EE(I),AAL(I),BBL(I),CCL(I),I=1,NFRICT[DRAG.1])  
 Read only if NFRICT > 0 on DRAG.1.  
 FORMAT(8F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AA(I)	First multiplicative coefficient on the Reynolds number for turbulent flow friction factor correlation (a in Eq. [6.14]).
6-10	BB(I)	First Reynolds number exponent for turbulent flow friction factor correlation (b in Eq. [6.14]).
11-15	CC(I)	Second multiplicative coefficient on the Reynolds number for turbulent flow friction factor correlation (c in Eq. [6.14]).
16-20	DD(I)	Second Reynolds number exponent for turbulent flow friction factor correlation (d in Eq. [6.14]).
21-25	EE(I)	Additive coefficient for turbulent flow friction factor correlation (e in Eq. [6.14]).
26-30	AAL(I)	Multiplicative coefficient on the Reynolds number for laminar flow friction factor correlation ( $a_L$ in Eq. [6.15]).
31-35	BBL(I)	Reynolds number exponent for laminar flow friction factor correlation ( $b_L$ in Eq. [6.15]).
31-35	CCL(I)	Additive coefficient for laminar flow friction factor correlation ( $c_L$ in Eq. [6.15]).

\*\*\* DRAG.2 is read NFRICT[DRAG.1] times. \*\*\*

### Axial Form Loss Coefficients

The pressure loss due to form drag on local obstructions in the flow field such as grid spacers and orifice plates is given by

$$\Delta P = C_D \frac{m|m|}{2g_c \rho A^2}$$

where  $C_D$  = loss coefficient  
 $m$  = upstream mass flow rate (lbm/s).

Axial form loss coefficients can be specified for any channel or group of channels, at any axial location in the problem. Local loss coefficients are specified by assembly type. Assemblies of the same type see the same form losses. In each assembly type having local losses, the user must define

the axial levels and channels where the losses occur, and specify the loss coefficient to be applied at each location. This input is specified on DRAG.3 and DRAG.4. These two input records are read in sequence NOLC[DRAG.1] times.

**DRAG.3** LCASST(I),NB1(I),LCFF(I) Read only if NOLC > 0 on DRAG.1.  
 FORMAT(3I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	LCASST(I)	Identification number of an assembly type with loss coefficients.
6-10	NB1(I)	Number of sets of channels in assembly type LCASST(I) that have loss coefficients. (NOTE: A channel set is defined as a group of sequentially numbered channels that have the same loss coefficients at the same axial locations.)
11-15	LCFF(I)	Identification number of loss coefficient forcing function to be applied to loss coefficients in assembly type LCASST(I) (optional input; must correspond to a forcing function entered on DRAG.6).

DRAG.4 (with DRAG.5, if needed) is read for L=1,NB1(I) sets of channels with local losses in assemblies of type LCASST(I).

**DRAG.4** (NLEVEL(I,L),ILCS(I,L),ILCE(I,L),(FACTOR(I,L,J),XCD(I,L,J),J=1,6)  
 Read only if NOLC > 0 on DRAG.1.  
 FORMAT(3I5,12F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NLEVEL(I,L)	Number of axial locations in the Lth location set for assembly type LCASST(I). (NOTE: L=1,NB1(I))
6-10	ILCS(I,L)	Identification number of starting channel in the Lth location set of assembly type LCASST(I).
11-15	ILCE(I,L)	Identification number of ending channel in the Lth location set of assembly type LCASST(I). (NOTE: all channels from ILCS(I,L) to ILCE(I,L) will be assigned the same losses at the same locations, as specified by the values for the FACTOR and ZCD arrays.)
16-20, 26-30, etc.	FACTOR(I,L,J)	Relative height (x/L) of the Jth axial location for the Lth channel set in assembly type LCASST(I).
21-25, 31-35, etc.	XCD(I,L,J)	Loss coefficient to be applied in channels ILCS(I,L) through ILCE(I,L) at axial location FACTOR(I,L,J).

Up to six pairs of (FACTOR, XCD) values can be entered on DRAG.4. If NLEVEL(I,L) is greater than six, the remaining pairs of (FACTOR,XCD) are read on DRAG.5 for the current location set before going on to the next location set.

**DRAG.5** (FACTOR(I,L,J),XCD(I,L,J),J=7,NLEVEL(I,L))  
 Read only if NLEVEL(I,L) > 6 on DRAG.4.  
 FORMAT(15X,12F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15		blank
16-20, 26-30, etc.	FACTOR(I,L,J)	Relative height (x/L) of the Jth axial location for the Lth channel set in assembly type LCASST(I).
21-25, 31-35, etc.	XCD(I,L,J)	Loss coefficient to be applied in channels ILCS(I,L) through ILCE(I,L) at axial location FACTOR(I,L,J).

DRAG.3 and DRAG.4 (with DRAG.5, if needed) are read in sequence NOLC[DRAG.1] times, with DRAG.4 (and DRAG.5) read NB1(I) times for each time DRAG.3 is read.

### Losses for Network Model

This input is required only if the network model has been specified by NETWK = 1 on DRAG.1. The assembly grouping for the network is defined by reading DRAG.6 for each assembly. Each assembly is assigned a group number, inlet loss parameters, and optional loss coefficient forcing functions. A loss coefficient for the total flow in all assemblies is specified on DRAG.7. Loss parameters for individual groups of assemblies are specified on DRAG.8 for NOGRP[DRAG.1] groups of assemblies.

**DRAG.6** (NETGRP(N),NINFF(N),NOUTFF(N),RAIN(N),PWIN(N),HAIN(N),RAOUT(N),  
 PWOUT(N),HAOUT(N),N=1,NASSEM[CHAN.1])  
 Read only if NETWK = 1 on CHAN.1  
 FORMAT(3I5,6F10.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NETGRP(N)	Assembly grouping number to be used with assembly N, corresponding to the assembly groups described in DRAG.9 (default is 1).
6-10	NINFF(N)	Identification number of loss coefficient forcing function versus Reynolds number, (defined on DRAG.5 and DRAG.6), to be used with assembly inlet loss, RAIN(N) (default is 0 for no forcing function).

11-15	NOUTFF(N)	Identification number of loss coefficient forcing function versus Reynolds number, to be used with assembly outlet loss RAOUT(N) (default is 0 for no forcing function).
16-25	RAIN(N)	Inlet loss parameter (ft-lbm) <sup>-1</sup> to be applied at inlet of assembly N.
26-35	PWIN(N)	Wetted perimeter (in.) associated with inlet loss RAIN (used only in Reynolds number calculation when NINFF(N) > 0; default is 1.0).
36-45	HAIN(N)	Gravitational head length (in.) associated with the assembly inlet loss.
46-55	RAOUT(N)	Outlet loss parameter (ft-lbm) <sup>-1</sup> to be applied at outlet of assembly N.
56-55	PWOUT(N)	Wetted perimeter (in.) associated with outlet loss RAOUT(N). (Used only in Reynolds number calculation when NOUTFF(N) > 0; default is 1.0).
66-75	HAOUT(N)	Gravitational head length (in.) associated with the assembly outlet loss.

\*\*\* DRAG.6 is read NASSEM[CHAN.1] times. \*\*\*

**DRAG.7** RTIN Read only if NETWK > 0 on DRAG.1  
 FORMAT(F10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	RTIN	Loss parameter (ft-lbm) <sup>-1</sup> to be applied to the total flow rate (sum of all assembly flow rates; Reynolds-number independent).

**DRAG.8** (RGIN(L),HGIN(L),RGOUT(L),HGOUT(L),L=1,NOGRP)  
 Read only if NOGRP > 0 on DRAG.1  
 FORMAT(4F10.5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	RGIN(L)	Loss parameter (ft-lbm) <sup>-1</sup> to be applied at assembly grouping inlet (Reynolds number independent).
11-20	HGIN(L)	Gravitational head length (in.) associated with the inlet pressure loss for group L.
21-30	RGOUT(L)	Loss parameter (ft-lbm) <sup>-1</sup> to be applied at assembly grouping outlet (Reynolds number independent).
31-40	HGOUT(L)	Gravitational head length (in.) associated with the outlet pressure loss for group L.

\*\*\* DRAG.8 is read NOGRP[DRAG.1] times. \*\*\*

## Loss Coefficient Forcing Functions

This input is read only if  $NLCFP > 0$  on DRAG.1. Loss coefficients are specified as constants (in  $XCD(I,L,J)$  on DRAG.4 and DRAG.5), but the user has the option of varying loss coefficients as a function of the Reynolds number. The forcing functions are numbered sequentially in the order they are defined on DRAG.7.

The variations in loss coefficients are specified as a function of Reynolds number. The same table of Reynolds number values applies to all forcing functions. The variations in loss coefficient with Reynolds number are expressed as multiplicative factors normalized to the value of  $XCD(I,L,J)$  entered on DRAG.4.

**DRAG.9** (RECL(I),I=1,NLCFP[DRAG.1]) Read only if  $NCLFP > 0$  on DRAG.1  
FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	RECL(I)	Reynolds number for the Ith point in the loss coefficient forcing function profiles.

NOTE: the table of Reynolds number values must be monotonically increasing.

**DRAG.10** ((FFLC(J,L),J=1,NLCFP[DRAG.1]),L=1,NLCFF[DRAG.1])  
Read only if  $NCLFF > 0$  and  $NCLFP > 0$  on DRAG.1.  
FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	FFLC(J,L)	The loss coefficient factor for Reynolds number RECL(I) in the Lth forcing function profile.

\*\*\* DRAG.10 is read NLCFF[DRAG.1] times. \*\*\*

## Local Blockages

This input allows the user to specify flow blockages at some axial locations in some channels. The axial flow at the blocked level is set to zero in the affected channels. Note that this input is specified by axial level number; if the axial noding is changed in a problem, this input must be adjusted to reflect the new locations of the blocked axial levels, or the blockages may turn up in the wrong places. This is optional and is read only if NBLOCK is greater than zero on DRAG.1.

**DRAG.11** NBLOKA,NBLOKC Read only if NBLOCK > 0 on DRAG.1.  
 FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NBLOKA	Axial level where the blockage is located.
6-10	NBLOKC	Number of channels blocked at axial level NBLOKA.

**DRAG.12** (IBLOKA(I),IBLOKC(I),I=1,NBLOKC[DRAG.10])  
 Read only if NBLOCK > 0 on DRAG.1  
 FORMAT(16I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	IBLOKA(I)	Identification number of assembly containing blocked channel IBLOKC(I).
6-10, 16-20, etc.	IBLOKC(I)	Identification number of the Ith blocked channel.

DRAG.11 and DRAG.12 are read in sequence NBLOCK times to define all blocked channels.

### Lateral Friction Loss Correlations

The default approach in COBRA-SFS is to treat lateral friction and form losses with a constant value for all gaps. This is specified by input as the gap resistance KIJ(1), defined in group CHAN on CHAN.8. This input provides an alternative, in which the lateral friction loss can be expressed as a Blasius-type friction factor correlation.

The lateral pressure loss due to flow through a gap can be expressed in the form

$$\Delta P = \frac{f' |w| w}{2g_c \rho S^2} \quad (6.16)$$

where  $f'$  = friction loss coefficient for flow through a single gap.

The friction loss coefficient is expressed in terms of the transverse Reynolds number, which is defined with the rod diameter as the characteristic length, such that

$$Re_D = \frac{V D_{rod} \rho}{\mu}$$



The friction loss coefficient is then represented by a set of turbulent and laminar formulations as follows:

$$f'_T = a_T Re_D^{b_T} + c_T Re_D^{d_T} + e_T$$

$$f'_L = a_L Re_D^{b_L} + c_L$$
(6.17)

The code evaluates the lateral friction factor as the maximum of the turbulent and laminar formulations as follows:

$$f' = \max(f'_T, f'_L)$$

The correlations are numbered sequentially from 1 to NFTRAN, in the order they are read in on DRAG.13. The friction factor correlation to be used in a particular lateral flow connection is determined by the value specified for N on CHAN.7 for the lower-numbered channel of the pair of channels that make up the lateral flow connection. The input for N on CHAN.7 is thus made to do double duty, since it also specifies the axial friction factor correlation (defined by input on DRAG.2) used to determine the channel friction pressure loss.

For example, suppose some channel II uses axial friction factor correlation number 3, specified on DRAG.2. For channel II that is connected to channel JJ through gap K, if NFTRAN is specified as greater than zero, then the lateral pressure loss in gap K will be determined using lateral friction factor correlation number 3, as specified on DRAG.13. The user must make certain that the value entered on CHAN.7 in variable N for a given channel corresponds to the correct axial friction factor for the channel *and* to the correct lateral friction factor correlation for the channel's lateral connections to other channels.

If the value of N on CHAN.7 for some channel II specifies an axial friction factor correlation number that has no corresponding lateral friction factor correlation number from DRAG.13, then the friction loss in any lateral flow connection from channel II to adjacent channels is calculated with the default loss coefficient, KIJ(1), entered on CHAN.8.

**DRAG.13** AAT(I),BBT(I),CCT(I),DDT(I),EET(I),AATL(I),BBTL(I),CCTL(I),  
I=1,NFTRAN) Read only if NFTRAN > 0 on DRAG.1.  
FORMAT(8F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	AAT(I)	First multiplicative coefficient on the Reynolds number for turbulent flow lateral wall friction factor correlation ( $a_T$ in Eq. [6.17]).
6-10	BBT(I)	First Reynolds number exponent for turbulent flow lateral wall friction factor correlation ( $b_T$ in Eq. [6.17]).
11-15	CCT(I)	Second multiplicative coefficient on the Reynolds number for turbulent flow lateral wall friction factor correlation ( $c_T$ in Eq. [6.17]).

16-20	DDT(I)	Second Reynolds number exponent for turbulent flow lateral wall friction factor correlation (i.e., $d_T$ in Eq. [6.17]).
21-25	EET(I)	Additive coefficient for turbulent flow lateral wall friction factor correlation ( $e_T$ in Eq. [6.17]).
26-30	AATL(I)	Multiplicative coefficient on the Reynolds number for laminar flow lateral wall friction factor correlation ( $a_L$ in Eq. [6.17]).
31-35	BBTL(I)	Reynolds number exponent for laminar flow lateral wall friction factor correlation ( $b_L$ in Eq. [6.17]).
36-40	CCTL(I)	Additive coefficient for laminar flow lateral wall friction factor correlation ( $c_L$ in Eq. [6.17]).

\*\*\* DRAG.13 is read in NFTRAN[DRAG.1] times. \*\*\*

### 6.2.10 Group BDRY--Thermal Boundary Conditions

The input for this group defines the heat transfer boundary conditions for the calculation. In most instances, this will be the interface between the system being modeled and the environment, but the user has a great deal of flexibility in specifying the heat transfer boundary conditions for a problem.

The default in COBRA-SFS is to assume that the side boundaries of a system are adiabatic. If there is heat transfer to the environment in the radial direction, boundary conditions must be defined explicitly for the sides of the system. In COBRA-SFS, heat is transferred out the boundaries through a series of user-defined boundary regions, each of which can use a different boundary connection type. A boundary connection type is defined as a semi-empirical correlation for the boundary surface heat flux  $q_b$ , and has the form

$$q_b = C_1 [C_2 (T_i - T_{i+1})]^{C_3} (T_i - T_{i+1}) + \sigma \left( \frac{1}{\epsilon_i} + \frac{1}{\epsilon_{i+1}} - 1 \right)^{-1} (T_i^4 - T_{i+1}^4) \quad (6.18)$$

where

- $C_n$  = user-defined constants
- $\sigma$  = Stefan-Boltzmann constant,  $0.1714(10^8)$  Btu/h-ft<sup>2</sup>-°R<sup>4</sup>
- $\epsilon_i, \epsilon_{i+1}$  = surface emissivity of boundary nodes  $i$  and  $i+1$
- $T_i, T_{i+1}$  = temperature of nodes  $i$  and  $i+1$ .

The total number of boundary types, NBTYP, is defined for a given problem by the user on BDRY.1. These boundary connection types are defined by input on BDRY.2, with coefficients for correlations of the form shown in Eq. (6.18). The total number of slab nodes that use these boundary connection types is specified in NWSID on BDRY.1.

The temperature of the environment seen by the boundary node is defined using axial profiles of boundary temperatures, read on BDRY.3. The number of axial boundary temperature profiles to be read in is defined by the user, as NBTEMP on BDRY.1. The user can read in as many axial profiles as needed to characterize a given problem. The boundary condition types seen by specific slab nodes are defined by input on BDRY.5 through BDRY.7. This input also includes the option to define axially varying boundary connection types.

If the plenum model is not used, the boundaries at the top and bottom of the system are assumed adiabatic for heat transfer, and the flow boundary conditions are defined by the input in group OPER. In this case, the inlet flow boundary condition treats the entering fluid as coming from an essentially infinite reservoir at the fluid boundary temperature and pressure defined in OPER. At the exit, the fluid vanishes into another essentially infinite reservoir at the specified boundary pressure. If flow should reverse at the exit, the properties of the incoming fluid are defined at the boundary pressure and some user-specified temperature.

The group BDRY input includes the alternative option of defining inlet and outlet plena to model a closed flow system, with heat transfer connections to the external environment. When the optional plenum model is used, the boundary conditions specified in group OPER are used as the initial values for inlet flow rate, temperature, and pressure. Conditions in the inlet plenum are calculated for each iteration by summing the flows from all channels having negative inlet flow. This temperature and pressure are used to define the conditions for flow into all channels with positive inlet flow. In the upper plenum, the conditions are calculated by summing the flow from channels with positive flow at the exit. These conditions define the temperature and pressure of flow into channels with negative flow at the exit.

The fluid conditions in the inlet plenum and the exit plenum are treated as one-dimensional. It is assumed that the flow mixes instantaneously in these regions, and no radial gradients are maintained in the fluid. Thermal boundary conditions on the upper and lower plena can vary radially, however. In addition, different plenum boundary regions can connect axially to different slab nodes. It is also possible to specify separate top and side boundary conditions on the plenum regions. The plenum input is specified on lines BDRY.8 through BDRY.14.

**BDRY.1** AGROUP,NBTYP,NBTEMP,NWSID,NPR  
 FORMAT(A4,1X,4I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter BDRY.
6-10	NBTYP	Number of thermal boundary connection types.
11-15	NBTEMP	Number of axial boundary temperature profiles.
16-20	NWSID	Number of solid structure nodes connected to the side thermal boundary.

21-25          NPR                  Plenum model flag:

= 0; no plenum heat transfer model.

= 1; one plenum only is modeled, either the upper or the lower plenum.

= 2; both lower and upper plenum are modeled.

### Thermal Boundary Connection Types

The thermal boundary connection types are structured to be as general as possible. They are formulated assuming that the heat flux in the boundary region can be expressed as a function of the temperature difference between adjacent boundary nodes, in the form shown in Eq. (6.18).

**BDRY.2**    NN,C1,C2,C3,EM1,EM2  
 FORMAT(I5,3F10.0,2F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NN	Thermal boundary connection type identification number. (NOTE: must be entered in sequence, from 1 to NBTYP[BDRY.1].)
6-15	C1	Leading coefficient for user-defined heat flux correlation at the boundary ( $C_1$ in Eq. [6.18]). (NOTE: $C_1$ is in units of Btu/s-in. <sup>2</sup> -°F.)
16-25	C2	Second coefficient for user-defined heat flux correlation at the boundary ( $C_2$ in Eq. [6.18]). (NOTE: $C_2$ has units of °F <sup>-1</sup> .)
26-35	C3	Exponent on temperature difference term of user-defined heat flux correlation at the boundary ( $C_3$ in Eq. [6.18]). (NOTE: $C_3$ is dimensionless.)
36-40	EM1	Surface emissivity for boundary node $i$ of a pair connected by thermal boundary connection type NN. (NOTE: enter zero if there is no radiative heat transfer for this boundary type.)
41-45	EM2	Surface emissivity for boundary node $i+1$ of a pair connected by thermal boundary connection type NN. (NOTE: enter zero if there is no radiative heat transfer for this boundary type. If EM1 > 0.0, the default for EM2 is 1.0.)

\*\*\* BDRY.2 is read NBTYP[BDRY.1] times. \*\*\*

## Boundary Axial Temperature Profiles

This input defines the temperature of the environment seen by the system. A boundary temperature can be specified as uniform by entering a two-point table, with the same temperature at relative axial locations 0.0 and 1.0. A total of NBTEMP[BDRY.1] boundary temperature profiles must be entered.

**BDRY.3** (N,NZONET(N),(ZENDT(I,N),BT(I,N),I=1,6)  
 Read only if NBTEMP > 0 on BDRY.1.  
 FORMAT(2I5,6(2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	N	Side boundary temperature profile identification number. (NOTE: must be entered in sequence, from 1 to NBTEMP.)
6-10	NZONET(N)	Number of pairs of entries in the Nth table of boundary temperature versus axial position. (NOTE: each table must include at least two points; for relative axial locations 0.0 and 1.0.)
11-15, 21-25, 31-35, etc.	ZENDT(I,N)	Relative axial location (x/L) of the Ith element of boundary temperature profile N. (ZENDT must begin with 0.0 and increase monotonically to 1.0)
16-20, 26-30, 36-40, etc.	BT(I,N)	Boundary temperature at location ZENDT (I,N).

Up to six pairs of values for (ZENDT[I,N],BT[I,N]) can be entered on BDRY.3. If NZONET(N) > six, the remaining pairs of points are read on BDRY.4, with 6 pairs of points per line. All points for the Nth profile are entered before going on to the input for the next profile.

**BDRY.4** (ZENDT(I,N),BT(I,N),I=7,NZONET(I)) Read only if NZONET(N) > 6.  
 FORMAT(10X,6(2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10		blank
11-15, 21-25, 31-35, etc.	ZENDT(I,N)	Relative axial location (x/L) of the Ith element of boundary temperature profile N. (ZENDT must begin with 0.0 and increase monotonically to 1.0.)
16-20, 26-30, etc.	BT(I,N)	Boundary temperature at location ZENDT(I,N).

\*\*\* BDRY.3 (with BDRY.4, if needed) is read NBTEMP[BDRY.1] times \*\*\*

## Side Thermal Boundary Connections

Heat transfer connections to the boundary must be defined for NWSID[BDRY.1] slab nodes. The heat transfer connections linking a given slab node to the boundary are defined as occurring through a series of sequentially connected regions, each of which can use a different boundary connection type to define the heat transfer through that region. In most cases, one region is sufficient to define the heat transfer connection from the slab node constituting the edge of the system to the environment, (e.g., from the outer surface of the cask to the ambient). In some cases, however, there may be additional material between the boundary slab node and the environment that is not modeled explicitly with slab nodes using the input in group SLAB. The effect of this material can be included by using appropriately defined boundary regions, connected by user-defined boundary connection types. The connection types specified between the boundary regions must define the heat flux through the regions using the relation in Eq. (6.18).

The total number of regions connecting a given slab node to the environment is defined by the user, in variable NHSID(I) on BDRY.5 for each of the NWSID nodes with connections to the boundary. The first region connects to the slab node on one side, and to the next region out on the other side. The remaining regions, 2 through NHSID(I), are connected in series, each to the adjacent region by an appropriate boundary type. The NHSID(I)th region connects to the specified boundary temperature. (If NHSID(I) is 1, the slab node is connected directly to the boundary temperature, using the specified boundary connection type, with no intermediate regions.)

The user can also specify a heat source term in the outermost radial region of a thermal boundary connection. This allows the user to model heat loading on the cask due to external conditions, such as fire or solar radiation.

For the IWSID(N) solid structure node identified on BDRY.5, the user must define the radial boundary region associated with that node. BDRY.6 is read NHSID(N) times to define all radial boundary regions connecting node IWSID(N) to the boundary temperature.

**BDRY.5** (IWSID(N),NHSID(N),AWSID(N),NBTYP(N),QBSRC(N),N=1,NWSID[BDRY.1])  
Read only if NWSID > 0 on BDRY.1.  
FORMAT(2I5,E5.3,I5,E5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IWSID(N)	Index number of Nth solid structure node connected to a side thermal boundary temperature.
6-10	NHSID(N)	Number of radial boundary regions connecting node IWSID(N) to the boundary temperature.
11-15	AWSID(N)	Perimeter (in.) of node IWSID(N) facing the boundary.
16-20	NBTYP(N)	Identification number of boundary temperature profile seen by node IWSID(N) on the far side of region NHSID(N). (NOTE: must correspond to the index of a profile entered on BDRY.3.)
21-25	QBSRC(N)	Optional boundary heat flux (Btu/h-ft <sup>2</sup> ); to be applied to the NHSID(N)th radial region (default is 0.0).

**BDRY.6** ((I,SPER(I,N),NZONEB(I,N),(ZENDB(I,N,K),NBCTYP(I,N,K),I=1,NHSID(N)),  
 K=1,6)  
 Read only if NHSID(N) > 0 on BDRY.5.  
 FORMAT(I5,F5.0,I5,6(F5.0,I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	I	Index of a radial boundary region associated with node IWSID(N). (NOTE: Boundary regions must be numbered sequentially from 1 to NHSID(N).)
6-10	SPER(I,N)	Perimeter of Ith boundary region of node IWSID(N), normalized to AWSID(N) (dimensionless).
11-15	NZONEB(I,N)	Number of axial zones to be read for a table of boundary connection type versus axial position for region I. (Default is 1, for an axially uniform boundary connection.)
16-20, 26-30, etc.	ZENDB(I,N,K)	Relative axial location (x/L) of the end of the Kth boundary connection type zone in region I. (NOTE: if NZONEB(I,N) is 1, the boundary is axially uniform, and ZENDB(I,N,1) must be entered as 1.0.)
21-25, 31-35, etc.	NBCTYP(I,N,K)	Boundary thermal connection type for the Kth axial segment of region I. (This index must correspond to a boundary connection type defined on BDRY.2).

NOTE: the first six pairs of (ZENDB,NBCTYP) for any radial boundary region I of the NWSID(N)th boundary node are read on BDRY.6. If NZONEB(I,N) > 6, the remainder are read on BDRY.7, with six entries on a line, before going on to the input for the next region.

**BDRY.7** (ZENDB(I,N,K),NBCTYP(I,N,K),K=7,NZONEB(I,J))  
 Read only if NZONEB(I,N) > 6 on BDRY.6.  
 FORMAT(15X,6(F5.0,I5))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-15		Blank
16-20, 26-30, etc.	ZENDB(I,N,K)	Relative axial location (x/L) of the end of the Kth boundary connection type zone in region I. (NOTE: if NZONEB(I,N) is 1, the boundary is axially uniform, and ZENDB(I,N,1) must be entered as 1.0.)

21-25, NBCTYP(I,N,K) Boundary thermal connection type for the Kth axial  
31-35, segment of region I (this index must correspond  
etc. to a boundary connection type defined on BDRY.2).

BDRY.6 (with BDRY.7, if needed) is read NHSID(N)[BDRY.5] times for node IWSID(N), which is connected to a thermal boundary.

\*\*\* BDRY.5 and BDRY.6 are read as a set NWSID[BDRY.1] times. \*\*\*

### Plenum Boundary Connections

This input group defines the flow and heat transfer connections to the upper and lower plena. This input is optional; if NPR[BDRY.1] is zero, the upper and lower heat transfer boundaries are assumed adiabatic, and the flow boundary is determined by the input in group OPER. When this option is used, the model may include the lower plenum only, the upper plenum only, or both the lower and upper plenum.

The upper and lower plena are basically specialized heat transfer boundary models, but also include a lumped parameter fluid node. Flow from all channels connected to a plenum is assumed to be instantaneously and completely mixed at a single temperature and pressure. The flow channels of all assemblies are assumed to connect to the plena, unless an assembly is explicitly specified as not connected, using the input on BDRY.13 and BDRY.14. The lower plenum provides an inlet fluid temperature for positive flow entering the channels at the inlet, and the upper plenum provides the fluid temperature for negative flow entering the channels at the exit. Natural recirculation is calculated in the momentum solution.

Heat transfer paths can be specified through the plena, connecting the slab nodes with the appropriate boundary temperatures by means of detailed connections between the plenum regions and the top or bottom of the slab nodes. The plena are defined with nominal heat transfer areas in the axial and radial directions. These areas model surfaces through which heat can move through the sides and top (or bottom) of the plenum. The heat transfer paths through the plenum to the boundary are defined using a series of plenum boundary regions, in the same manner as the boundary regions for the side boundary nodes. These regions connect radially or axially from the plenum to the boundary, and the connections between regions are specified by thermal boundary connection type, as defined on BDRY.2. The ambient temperature of the environment beyond the upper or lower plenum is specified in TAMB(1,N) for the radial regions, and in TAMB(2,N) for the axial regions, both on BDRY.8. The user may also specify heat sources in the outermost region of the plenum.

The input for each plenum is read on BDRY.8 through BDRY.12. If NPR[BDRY.1] is 1, only one plenum is specified. It may be either the upper plenum or the lower plenum, as specified by the flag N on BDRY.8. If NPR[BDRY.1] is specified as 2, both the lower plenum and upper plenum must be defined. In this case, the lower plenum is defined in the first set of inputs for BDRY.8 through BDRY.12, and the upper plenum is defined in the second set of inputs.



**BDRY.8** N,AWPS(N),AWPAX(N),NPSID(N),NPAX(N),TAMBP(1,N),TAMBP(2,N),  
 QPSRC(1,N),QPSRC(2,N)  
 Read only if NPR > 0 on BDRY.1  
 FORMAT(I5,2F5.3,2I5,4F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	N	Flag for plenum region under consideration:  = 1; lower plenum.  = 2; upper plenum.
6-10	AWPS(N)	Nominal plenum radial heat transfer area (in. <sup>2</sup> ).
11-15	AWPAX(N)	Nominal plenum axial heat transfer area (in. <sup>2</sup> ).
16-20	NPSID(N)	Number of radial plenum boundary regions.
21-25	NPAX(N)	Number of axial plenum boundary regions.
26-30	TAMBP(1,N)	Plenum radial boundary temperature (F).
31-35	TAMBP(2,N)	Plenum axial boundary temperature (F).
36-40	QPSRC(1,N)	Optional plenum radial boundary heat flux (Btu/h-ft <sup>2</sup> ). (NOTE: this is applied to the outermost plenum radial region only.)
41-45	QPSRC(2,N)	Optional plenum axial boundary heat flux (Btu/h-ft <sup>2</sup> ). (NOTE: this is applied to the outermost plenum axial region only.)

**BDRY.9** (I,SPERP(K,1,N),NPTYP(K,1,N),NPWN, (IPINTP(1,NPW),MATYPP(NPW),  
 AREACP(NPW),DXPLEN(NPW),J=1,3)  
 Read only if NPSID(N) > 0 on BDRY.8  
 FORMAT(I5,F5.0,2I5,3(2I5,2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	I	Plenum radial region number. (NOTE: Regions must be numbered in sequence from 1 to NPSID(N).)
6-10	SPERP(K,1,N)	Plenum radial region heat transfer area multiplier, normalized to AWPS(N) on BDRY.8 (dimensionless).

11-15	NPTYP(K,1,N)	Thermal connection type number for connections between slab nodes and plenum radial region I. (NOTE: This must correspond to a boundary connection type defined on BDRY.2.)
16-20	NPWN	Number of slab nodes connected to plenum radial region I.
21-25, 41-45, 61-65	IPINTP(1,NPW)	Identification number of the Jth slab node connected to plenum radial region I. (NOTE: must correspond to a node entered on SLAB.3.)
26-30, 46-50, 66-70	MATYPP(NPW)	Identification number of the material type for the connection between node IPINTP(1,NPW) and plenum axial region I.
31-35, 51-55, 71-75	AREACP(NPW)	Heat transfer area (in. <sup>2</sup> ) for connection between node IPINTP(1,NPW) and plenum region I. (Default is node IPINTP(1,NPW) cross-sectional area for axial conduction, WALLXC, specified on SLAB.3.)
36-40, 56-60 76-80	DXPLEN(NPW)	Length (in.) for heat transfer from node IPINTP(1,NPW) to the center of plenum radial region I.

NOTE: Up to three sets of values for (IPINTP, MATYPP, AREACP, DXPLEN) may be read on BDRY.9. If NPWN[BDRY.9] is greater than three, then the remaining sets of values are read on BDRY.10, with 3 sets of values per line, before going on to the input for the next plenum region.

**BDRY.10** (IPINTP(1,NPW),MATYPP(NPW),AREACP(NPW),DXPLEN(NPW),J=4,NPWN)  
Read only if NPWN > 3 on BDRY.9  
FORMAT(20X,3(2I5,2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-20		BLANK
21-25, 41-45, 61-65	IPINTP(1,NPW)	Identification number of the Jth slab node connected to plenum radial region I. (NOTE: must correspond to a node entered on SLAB.3.)
26-30, 46-50, 66-70	MATYPP(NPW)	Identification number of the material type for the connection between node IPINTP(1,NPW) and plenum axial region I.
31-35, 51-55, 71-75	AREACP(NPW)	Heat transfer area (in. <sup>2</sup> ) for connection between node IPINTP(1,NPW) and plenum region I. (Default is node IPINTP[1,NPW] cross-sectional area for axial conduction; WALLXC, specified on SLAB.3.)

36-40, 56-60 DXPLEN(NPW) Length (in.) for heat transfer from node IPINTP(1,NPW) to the center of plenum radial region I.

76-80

\*\*\* BDRY.9 (with BDRY.10, if needed), is read  $K=1, NPSID(N)[BDRY.8]$  times. \*\*\*

**BDRY.11** (I,SPERP(K,2,N),NPTYP(I,2,N),NPWN, (IPINTP(1,NPW),MATYPP(NPW), AREACP(NPW),DXPLEN(NPW),J=1,3)  
 Read only if NPAX(N) > 0 on BDRY.8  
 FORMAT(I5,F5.0,2I5,3(2I5,2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	I	Plenum axial region number. (NOTE: Regions must be numbered sequentially from 1 to NPAX(N).)
6-10	SPERP(K,2,N)	Plenum axial region heat transfer area multiplier, normalized to AWPAX(N) on BDRY.8.
11-15	NPTYP(K,2,N)	Thermal connection type for connections between slab nodes and plenum axial region I. (NOTE: This must correspond to a boundary connection type defined on BDRY.2.)
16-20	NPWN	Number of solid nodes connected to plenum axial region I.
21-25, 41-45, 61-65	IPINTP(1,NPW)	Identification number of the Jth solid node connected to plenum axial region I. (NOTE: must correspond to a node entered on SLAB.3.)
26-30, 46-50, 66-70	MATYPP(NPW)	Identification number of the solid material type for the connection between IPINTP(1,NPW) and plenum axial region I.
31-35, 51-55, 71-75	AREACP(NPW)	Heat transfer area (in. <sup>2</sup> ) for connection from slab node IPINTP(1,NPW) and plenum axial region I (default is IPINTP(1,NPW) area for axial conduction, WALLXC, specified on SLAB.4).
36-40, 56-60,	DXPLEN(I,J)	Length (in.) for heat transfer from node IPINTP(1,NPW) to center of plenum axial region I. 76-80

NOTE: Up to three sets of values for (IPINTP, MATYPP, AREACP, DXPLEN) may be read on BDRY.11. If NPWN[BDRY.11] is greater than three, then the remaining sets are read on BDRY.12, with three sets of values per line, before going on to the next plenum region.

**BDRY.12**

(IPINTP(I,J),MATYPP(I,J),AREACP(I,J),DXPLEN(J),J=4,NPWN)

Read only if NPWN &gt; 3 on BDRY.11

FORMAT(20X,3(2I5,2F5.0))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-20		Blank
21-25, 41-45, 61-65	IPINTP(1,NPW)	Identification number of the Jth slab node connected to plenum axial region I. (NOTE: must correspond to a node entered on SLAB.3.)
26-30, 46-50, 66-70	MATYPP(NPW)	Identification number of the solid material type for the connection between IPINTP(1,NPW) and plenum axial region I.
31-35, 51-55, 71-75	AREACP(NPW)	Heat transfer area (in. <sup>2</sup> ) for connection from slab node IPINTP(1,NPW) and plenum axial region I. (Default is IPINTP(1,NPW) area for axial conduction, WALLXC, specified on SLAB.4).
36-40, 56-60,	DXPLEN(I,J)	Length (in.) for heat transfer from node IPINTP(1,NPW) to center of plenum axial region I. 76-80

\*\*\* BDRY.11 (with BDRY.12, if needed), is read K=1,NPAX(N)[BDRY.8] times. \*\*\*

\*\*\* The input lines BDRY.8 through BDRY.12 are read as a set NPR[BDRY.1] \*\*\*

\*\*\* times to define the plenum models for a given problem. \*\*\*

**Assemblies Not Connected to Plena**

This input is read only when the plenum model is used. It allows the user to specify assemblies that are not connected to the plena. The total number of unconnected assemblies is specified in the variable NNPA in BDRY.13, and the index numbers of the unconnected assemblies are read on BDRY.14. If all assemblies are connected to the plena, then NNPA is specified as zero, and BDRY.14 is not read.

**BDRY.13** NNPA Read only if NPR > 0 on BDRY.1  
FORMAT(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NNPA	Number of assemblies that are <b>not</b> connected to the plenum model

**BDRY.14** IUNCA(I), I=1,NNPA Read only if NNPA > 0 on BDRY.13  
 FORMAT(12I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, 16-20,	IUNCA(I)	Index number of the Ith assembly that is <b>not</b> connected to the plenum model

56-60

\*\*\* BDRY.14 is read as many times as necessary to supply NNPA entries, \*\*\*  
 \*\*\* with up to 12 entries on a record. \*\*\*

### 6.2.11 Group OPER--Operating Conditions

The input for this group defines the flow boundary conditions for the system, and the heat generation rate for the rods. When the plenum model in group BDRY is not used, group OPER is used to specify the flow boundary conditions at the inlet and outlet of the system. The user specifies the temperature and flow rate for fluid entering the system at the inlet to the channels and the channel exit pressure. This constitutes a complete set of flow boundary conditions for normal upflow conditions. If the flow could be negative at a channel exit, the user must also specify an enthalpy for fluid carried into the system by this reverse flow.

If the lower plenum model is specified in group BDRY, the inlet conditions are calculated as part of the overall solution, based on fluid conditions in the plenum. In such a case, the boundary conditions specified in group OPER are used as the first guess in the iterative solution. If the upper plenum model is used, the properties of the fluid transported into the channels if flow reverses at the exit are determined from the solution of the fluid conditions in the upper plenum.

This group can also be used to define forcing functions for transient calculations. The transient capability has not been validated, however, and should be used with due caution. Axial and radial distributions of the heating rates are also specified in this group.

**OPER.1** AGROUP,IHH,IG,ITDP,NRPF,NP,NH,NG,NGPRFL,NHX,NQ,NQPRFL,  
 NDPA,NRAMPH,NAXP FORMAT(A4,1X,14I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter OPER.
6-10	IHH	flag for units of enthalpy or temperature on inlet (or initial) fluid conditions:  = 0; uniform inlet (or initial) enthalpy (Btu/lbm) in all channels.

= 1; uniform inlet (or initial) temperature ( $^{\circ}\text{F}$ ) in all channels.

= 2; define inlet enthalpy (Btu/lbm) for each channel on OPER.3.

= 3; define inlet temperature ( $^{\circ}\text{F}$ ) for each channel on OPER.3.

(NOTE: options 2 or 3 should not be used with the plenum model if all assemblies are connected to the lower plenum. However, IHH = 2 or 3 can be used to specify non-uniform inlet enthalpies or temperatures in the channels of any assemblies that are not connected to the lower plenum.)

11-15

IG

Flag for option to specify inlet (or initial) flow distribution:

= 0; average inlet mass flux ( $\text{Mlbm/h-ft}^2$ ).

= 2; average inlet mass flux ( $\text{Mlbm/h-ft}^2$ ), but assembly inlet flows will be determined by user-supplied flow fractions on OPER.4. Channel mass flux will be uniform within each assembly.

= 3; average inlet mass flux ( $\text{Mlbm/h-ft}^2$ ), but channel flows in each assembly will be determined by user-supplied flow fractions on OPER.7.

= 4; average inlet mass flux ( $\text{Mlbm/h-ft}^2$ ), but flows will be determined by both assembly flow fractions on OPER.4 and channel flow fractions on OPER.6.

(NOTE: when the lower plenum is modeled, IG=0 is the logical option for this input, unless it is necessary to specify inlet flow conditions for assemblies that are not connected to the plenum.)

16-20

ITDP

Flag for flow or pressure drop boundary condition to be applied to all assemblies:

= -1; boundary condition flag set for each assembly by using ITDPA(NASS) on CHAN.6.

= 0; specified flow boundary condition.

- = 1; uniform pressure drop (specified in DPS on OPER.2).
- = 2; uniform inlet mass flux (required pressure drop calculated as part of the solution).
- = 3; uniform pressure drop calculated for zero net inlet mass flow rate (FTOTAL on OPER.2. This is valid only for FTOTAL = 0.0).
- = 4; zero flow at both top and bottom boundaries.  
(NOTE: ITDP=4 can be used only when the plenum model is not used.)

(NOTE: If the plenum model is used, the logical choice for ITDP is 3. If any assemblies are not connected to the plenum, the individual boundary condition flag, ITDPA(NASS) [CHAN.6], should be specified as 3 for each assembly connected to the plenum. In such a case, ITDP must be -1.

21-25 NRPF

Flag for assembly heating rate specification:

- = 0; all assemblies have the same average heat generation rate (PDN on OPER.2).
- = 1; average heating rate is specified in PDN on OPER.2, and relative heat generation factors normalized to PDN are specified for each assembly on OPER.8.
- = 2; heat generation rate is read for each assembly, on OPER.8 (PDN on OPER.2 is not used).

26-30 NP

Number of pairs of values in table for transient forcing function on system pressure, to be read on OPER.9.

31-35 NH

Number of pairs of values in transient forcing function table on inlet enthalpy or temperature, to be read on OPER.10.

(NOTE: If the lower plenum model is specified in group BDRY, this option can be used to define forcing functions only for channels in assemblies not connected to the lower plenum. Forcing functions cannot be applied to the inlet flow of channels in assemblies connected to the lower plenum.)

- 36-40      NG      Number of pairs of values in transient forcing function table on the flow boundary, to be read on OPER.11 and OPER.12:
- if ITDP = 0 or 2, the forcing function is on the inlet flow rate.
  - if ITDP = 1 or 3, the forcing function is on the pressure drop boundary condition.
  - if ITDP = 4, this option cannot be used.
- (NOTE: If the lower plenum model is specified in group BDRY, this option can be used to define forcing functions only for channels in assemblies not connected to the lower plenum. Forcing functions cannot be applied to the flow boundary condition on channels in assemblies connected to the lower plenum.)
- 41-45      NGPRFL      Number of transient forcing functions for inlet mass flux or pressure drop profiles to be read on OPER.12. (Used only if NG > 0.)
- 46-50      NHX      Number of pairs of values in transient forcing function table on exit enthalpy or temperature, to be read on OPER.15.
- (NOTE: If the upper plenum model is specified in group BDRY, this option can be used to define a forcing function only for channels in assemblies not connected to the upper plenum. A forcing function cannot be applied to the exit enthalpy boundary condition on channels in assemblies connected to the upper plenum.)
- 51-55      NQ      Number of entries in table(s) of transient forcing function(s) on average heat generation rate. The table of time-values to be used by all forcing functions on heat generation rate is read on OPER.13.
- (NOTE: all tables of heat generation rate forcing functions must use the same set of time values.)
- 56-60      NQPRFL      Number of heat generation rate transient forcing function tables to be read on OPER.14 (used only if NQ > 0).
- 61-65      NDPA      Flag for option to specify different pressure drop boundary conditions for each assembly when using the network model.
- = 1; assembly pressure drops to be specified on OPER.18.



= 0; uniform pressure drop or flow boundary condition.

(NOTE: The network model is specified by NETWK > 0 on DRAG.1. If NETWK is specified as zero, the input for NDPA is not used. See the input instructions for group DRAG.)

(WARNING: The plenum model specified in group BDRY cannot be used with the network option. If the network option is used, ITDP must be specified as 1 or 2, and all assemblies must use the same boundary condition option.)

66-70	NRAMPH	Number of iterations over which the heat generation is to be ramped into the solution (default is 1).
71-75	NAXP	Number of axial heat generation profiles to be entered on OPER.16 and OPER.17 (default is zero, for uniform axial heat generation distribution for all rods).

### Operating Conditions

This input defines the nominal operating conditions for the system. When the plenum model is not used, the input on OPER.2 defines the steady-state system pressure, inlet flow enthalpy or temperature, average heat generation rate, and flow or pressure drop boundary condition on the system. When the plenum model is used, this input defines the initial state of the system at the beginning of the steady-state iterative solution.

**OPER.2** PEXIT,HIN(1),GIN,PDN,HOUT,DPS,FTOTAL  
 FORMAT(7F10.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-10	PEXIT	System pressure (psia). Used to calculate the boiling temperature limit, based on the PLIQ[PROP.2] array (not used when coolant is gas).
11-20	HIN	If IHH = 0 or 2 on OPER.1, average inlet (or initial) enthalpy (Btu/lbm).

If IHH = 1 or 3, average inlet temperature (°F).

(NOTE: When there are 2 fluid types defined in input group PROP and IHH = 0 [or 1], it is assumed that both fluids have the same inlet enthalpy [or temperature]. If IHH > 1, the inlet enthalpy or temperature is defined for each channel individually, on input line OPER.3.)

21-30	GIN	Average inlet mass flux (Mlbm/h-ft <sup>2</sup> ).
31-40	PDN	Average heat generation rate (MBtu/h-ft <sup>3</sup> ): <ul style="list-style-type: none"> <li>- When the fuel model is used (NC &gt; 4 on RODS.1), PDN is based on the volume of the fuel (defined using fuel diameter, DROD[I], specified on RODS.4).</li> <li>- When the fuel model is not used (NC = 0 or NC = 1 on RODS.1), PDN is based on volume of the rod, (defined using rod diameter, DIA[I], specified on RODS.3).</li> </ul>
41-50	HOUT	If IHH = 0 or 2 on OPER.1, exit enthalpy (Btu/lbm).  If IHH = 1 or 3, exit temperature (°F)  (NOTE: HOUT is used to define fluid conditions for reverse flow at the exit of any channel. If the upper plenum model is used, however, HOUT is applied only to reverse exit flow in channels of assemblies that are not connected to the plenum. Default is HOUT = HIN.)
51-60	DPS	Total system pressure drop (psia) (used only when ITDP = 1 or 2 on OPER.1, or if ITDPA [NASS] = 1 or 2 on CHAN.6 for one or more assemblies.)
61-70	FTOTAL	Total inlet flow rate (lbm/s) (used only when ITDP = 3 on OPER.1).

### Nonuniform Operating Parameter Distributions

This input is read only if option flags IHH or IG on OPER.1 have been defined with values that require detailed channel-by-channel specification of inlet flow conditions. The input on lines OPER.3 through OPER.8 is used to specify nonuniform inlet conditions for channel flow and enthalpy (or temperature), nonuniform assembly pressure drops, and to define a nonuniform radial distribution of heat generation rate.

### Inlet Enthalpy/Temperature Distribution

This input is optional; it is read only if IHH is specified as 2 or 3 on OPER.1.

**OPER.3** (HINLET(I),I=1,NCHANL) Read only if IHH > 1 on OPER.1  
 FORMAT(12E5.0)

Columns	Variable	Description
1-5, 6-10, 11-15, etc.	HINLET(I)	If IHH = 2 on OPER.1, inlet enthalpy (Btu/lbm) for channel I. If IHH = 3 on OPER.1, inlet temperature (°F) for channel I.

### Inlet Flow Distribution by Assembly

This input is optional; it is read only if IG is specified as 2 or 4 on OPER.1.

**OPER.4** (ZDUM(I),I=1,NASSEM[CHAN.1]) Read only if IG = 2 or 4 on  
OPER.1  
FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	ZDUM(I)	Inlet flow fraction for assembly I, normalized to the average mass flux, GIN on OPER.2.  Within assembly I, the inlet flow for each channel II is calculated as

$$F(II,1) = ZDUM(I)*[GIN*(10)^6/3600.]*A(I,1)$$

### Inlet Flow Distribution by Channel

This input is optional; it is read only if IG is specified as 3 or 4 on OPER.1. The total number of assemblies that have specified inlet flow distributions is specified as NASSIN on OPER.5. Then OPER.6 and OPER.7 are read in sequence NASSIN[OPER.5] times to define the specified inlet flow for the channels in the individual assemblies.

**OPER.5** NASSIN Read only if IG = 3 or 4 on OPER.1  
FORMAT(15)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NASSIN	Number of assemblies for which channel inlet flow distribution is to be specified using the input on OPER.6 and OPER.7.

**OPER.6** NASS Read only if NASSIN > 0 on OPER.5  
FORMAT(15)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NASS	Identification number of an assembly with specified channel inlet flow distribution.

**OPER.7** TERM1(II), II=NCHAN1,NCHAN2) Read only if NASS > 0 on OPER.6  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	TERM1(II)	Inlet mass flow fraction for channel II of assembly NASS[OPER.6]; this value must be normalized to GIN on OPER.2. The mass flow fraction is used to define the channel inlet flow as

$$F(II,1) = \text{TERM1}(II) * [\text{GIN} * 10^6 / 3600.] * A(II,1)$$

\*\*\* OPER.6 and OPER.7 are read in sequence NASSIN[OPER.5] times. \*\*\*

### Assembly Heat Generation Distribution

This input is optional; it is read only if NRPF is specified as greater than zero on OPER.1.

**OPER.8** (PDNA(N),N=1,NASSEM[CHAN.1]) Read only if NRPF > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	PDNA(N)	Heating rate for assembly N: if NRPF = 1, PDNA(N) is normalized to PDN on OPER.2.  if NRPF = 2, PDNA(N) is assembly heating rate in units of MBtu/h-ft <sup>3</sup> (PDN on OPER.2 is not used).

### Transient Forcing Function Tables

The input on lines OPER.9 through OPER.15 can be used to specify forcing functions on the system pressure, inlet enthalpy or temperature, inlet flow rate or pressure drop, and the heat generation rate. However, the transient capability has not been validated in the Cycle 2 release of the code, and should be used with caution. Forcing functions on the inlet flow conditions are ignored for assemblies connected to a lower plenum model. They are also ineffective when using the zero-flow boundary condition option, specified with ITDP = 4 on OPER.1.

### System Pressure Transient Forcing Function

This input is optional; it is read only if NP is specified as greater than zero on OPER.1.

**OPER.9** (YP(I),FP(I),I=1,NP) Read only if NP > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	YP(I)	Transient time (sec) for the Ith element of system pressure forcing function table. (NOTE: This array must be entered as monotonically increasing from time zero.)
6-10, 16-20, etc.	FP(I)	Fraction of steady-state system pressure at time YP(I), normalized to PEXIT on OPER.2.

### **Inlet Enthalpy or Temperature Transient Forcing Function**

This input is optional; it is read only if NH is specified as greater than zero on OPER.1.

**OPER.10** (YH(I),FH(I),I=1,NH) Read only if NH > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	YH(I)	Transient time (sec) for the Ith element of enthalpy (or temperature) forcing function table. (NOTE: This array must be entered as monotonically increasing from time zero.)
6-10, 16-20, etc.	FH(I)	If IHH = 0 or 2, the fraction of inlet enthalpy at time YH(I), normalized to HIN entered on OPER.2. If IHH = 1 or 3, the fraction of inlet temperature at time YH(I), normalized to HIN entered on OPER.2.

### **Inlet Flow or Pressure Drop Transient Forcing Functions**

This input is optional. OPER.11 and OPER.12 are read only if NG is specified as greater than zero on OPER.1. OPER.11 is read only once to define the time array to be used with all NGPRFL[OPER.1] forcing functions. OPER.12 is read NGPRFL[OPER.1] times to define each of the inlet flow or pressure drop forcing functions. The tables are numbered sequentially in the code, in the order in which they are read in on OPER.12. These sequence numbers are assigned to specific assemblies by the values entered for NPFVT(NASS) on CHAN.6.

**OPER.11** (YG(J),J=1,NG) Read only if NG > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	YG(J)	Transient time (sec) for the Jth element of inlet flow or pressure drop forcing function tables. (NOTE: must be entered as monotonically increasing from time zero.)

**OPER.12** ((FG(J,L),J=1,NG),L=1,NGPRFL)  
 Read only if NG > 0 and NGPRFL > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	FG(J,L)	If ITDP = 0 or 2 on OPER.1, the inlet flow fraction at time YG(J) for the Lth table, normalized to GIN entered on OPER.2. (NOTE: If ITDP = -1, the inlet flow forcing function[s] can be applied only to assemblies with ITDPA[IASS] defined as 0 or 2 on CHAN.6.)  If ITDP = 1 on OPER.1, the pressure drop multiplier at time YG(J) for the Lth table, normalized to DPS entered on OPER.2. (NOTE: If ITDP = -1, the pressure drop forcing function[s] can be applied only to channels in assemblies with ITDPA[IASS] defined as 1 on CHAN.6.)

\*\*\* OPER.12 is read NGPRFL[OPER.1] times. \*\*\*

### Heat Generation Rate Transient Forcing Functions

This input is optional. OPER.13 and OPER.14 are read only if NQ is specified as greater than zero on OPER.1. OPER.13 is read only once to define the time array to be used with all NQPRFL[OPER.1] forcing functions. OPER.14 is read NQPRFL[OPER.1] times to define the heat generation rate forcing functions. The tables are numbered sequentially in the code, in the order in which they are read in on OPER.14. These sequence numbers are assigned to specific assemblies by the values entered in NHFVT(NASS) on CHAN.6.

**OPER.13** (YQ(J),J=1,NQ) Read only if NQ > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	YQ(J)	Transient time (sec) for the Jth element of heat generation rate forcing function tables. (NOTE: must be entered as monotonically increasing from time zero.)

**OPER.14** ((FQ(J,L),J=1,NQ),L=1,NQPRFL)  
 Read only if NQ > 0 and NQPRFL > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	FQ(J,L)	The heat generation rate multiplier, normalized to PDN on OPER.2, at transient time YQ(J), for the Lth table.

\*\*\* OPER.14 is read NQPRFL[OPER.1] times. \*\*\*

### Exit Enthalpy Transient Forcing Function

This input is optional; it is read only if NHX is specified as greater than zero on OPER.1.

**OPER.15** (YHX(I),FHX(I),I=1,NHX) Read only if NHX > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	YHX(I)	Transient time (sec) for the Ith element of exit flow conditions forcing function table. (NOTE: must be entered as monotonically increasing from time zero.)
6-10, 16-20, etc.	FHX(I)	If IHH = 0 or 2, the fraction of exit enthalpy entered in HOUT on OPER.2. If IHH = 1 or 3, the fraction of exit temperature entered in HOUT on OPER.2.

### Axial Heat Generation Profiles

Unless specified otherwise, the axial heat generation profile is assumed to be a uniform distribution along the total axial length Z, specified in group CHAN on CHAN.3. This default option is obtained simply by entering NAXP as zero on OPER.1. In most applications, however, the axial heat generation rate will not be axially uniform. Nonuniform axial heat generation profiles can be specified as tables of relative axial power factors versus axial distance. This option is specified by setting NAXP on OPER.1 to the number of axial heat generation rate profiles needed for the problem. Tables describing the axial heat generation rate distributions are then entered using the input on OPER.16 and OPER.17.

The axial heat generation rate profiles are numbered sequentially from 1 to NAXP, in the order they are entered on OPER.16 and OPER.17. If the option for the default uniform profile is used, this profile is automatically numbered heat generation profile 1. The particular profile to be used in an assembly is specified by input on CHAN.5, by the value specified for NAFLX(NASS).

**OPER.16** NAX(I) Read only if NAXP > 0 on OPER.1  
 FORMAT(15)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NAX(I)	Number of pairs of entries in the Ith axial heat generation rate profile table.

**OPER.17** (Y(L,I),AXIAL(L,I),L=1,NAX(I)) Read only if NAXP > 0 on OPER.1  
 FORMAT(12F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, 21-25, etc.	Y(L,I)	Relative location (X/L) of the Lth element of axial heat generation rate profile I. (NOTE: Must be entered monotonically increasing from 0.0 to 1.0.)
6-10, 16-20, 26-30, etc.	AXIAL(L,I)	Relative heat generation rate at Y(I,L), normalized to average heat generation rate, in PDN on OPER.2.

\*\*\* OPER.16 and OPER.17 are read in sequence NAXP[OPER.1] times. \*\*\*

### Non-Uniform Pressure Drop Boundary Condition

This input is optional. It is read only if NDPA is greater than zero on OPER.1. It is used to specify a separate pressure drop boundary condition for each assembly. This option can be used only with ITDP = 1 on OPER.1, or if some assemblies have ITDPA(NASS) = 1 specified on CHAN.5. When this option is used, the pressure drop entered in DPA on OPER.2 is ignored.

**OPER.18** (DPA(I),I=1,NASSEM[CHAN.1]) Read only if NDPA > 0 on OPER.1  
 FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, etc.	DPA(I)	Pressure drop (psi) for assembly I. (NOTE: Supersedes DPS on OPER.2. This option cannot be used with the network model, flagged by NETWK > 0 on DRAG.1.)

### 6.2.12 Group CALC--Calculational Parameters

This group defines the calculational parameters needed to specify the manner in which the problem will be solved. The input specifies the various convergence limits, damping factors, acceleration factors, and iteration limits required for the steady state solution. The simulation time and time step size for transients is also defined by this input.



**CALC.1** AGROUP,ISCHEME,ITSTEP,NODUMP  
 FORMAT(A4,1X,I5,5X,2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter CALC.
6-10	ISCHEME	Lateral momentum solution flag:  = 0; lateral momentum equation solved for crossflows (default).  = 1; lateral flows set to zero (no solution of lateral momentum equation).
11-15		Blank
16-20	ITSTEP	Number of entries in variable time step table. (NOTE: this input is used only for transient calculations. Default is zero, for uniform time step size throughout the transient.)
21-25	NODUMP	Flag for option to write a binary restart file on logical unit 8:  = 0; create a restart file (default).  = 1; do not create a restart file.

**Solution Damping and Acceleration Factors**

Damping and acceleration factors used in the solution are entered on CALC.2. All factors are applied as

$$X^n = \alpha X^n + (1-\alpha)X^{n-1}$$

where X = calculational variable  
 $\alpha$  = damping or acceleration factor  
 n = current iteration number.

Convergence limits are defined in nondimensional form so that the relative magnitude of the affected variable does not affect the rate of convergence.

CALC.2 TTIME,WERRY,FERROR,HERROR,QERROR,DAMPNG,ACCELY,ACCELF,  
 ACCRHO,ACCELW,ACCROD,ACCAX,SLDAMP  
 FORMAT(13E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	TTIME	Total transient time (sec) (default is a steady state calculation, with TTIME = 0.0).
6-10	WERRY	Inner loop convergence criterion for the recirculation solution scheme (default is 0.001).
11-15	FERROR	External axial flow convergence criterion, defined for the implicit axial momentum equation as the maximum allowable error for iterative axial flows. If the error is greater than FERROR, another iterative sweep of the entire bundle is made (default is 0.01).
16-20	HERROR	Convergence criterion for fluid enthalpy (default is 0.001).
21-25	QERROR	Convergence criterion for total energy balance (default is 0.001).
26-30	DAMPNG	No longer used; leave blank.
31-35	ACCELY	Acceleration factor for the iterative Gauss-Seidel solution of the momentum equations (default is 1.0).
36-40	ACCELF	Damping factor for iterative axial flow (default is 0.7).
41-45	ACCRHO	Damping factor for change in fluid density (default is 1.0).
46-50	ACCELW	Acceleration factor for the solid structure energy equation solution (default is 1.2).
51-55	ACCROD	Damping factor for the rod energy solution (default is 0.5).
56-60	ACCAX	Damping factor for axial energy rebalancing (default is 0.0, no axial energy rebalancing).
61-65	SLDAMP	Damping factor in the energy solution for slab conduction (default is 0.0).

## Time Step and Iteration Control

This input specifies the number of time steps for a transient, and the time step size, if the variable time step option is used. For steady state calculations, the number of time steps must be specified as zero. The number of iterations for the steady state solution is specified with NTRIES on CALC.3. This value is used for all time steps in a transient, as well.

**CALC.3** NDT,NTRIES  
FORMAT(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NDT	Total number of time steps. (NOTE: enter zero for steady-state calculations.)
6-10	NTRIES	Maximum number of external iterations allowed for any time step, including the steady state at time zero(default is 20).

The default in the code is a uniform time step size for the transient, calculated from NDT on CALC.3 and TTIME on CALC.2 as  $DT = TTIME/NDT$ . Nonuniform time step sizes are specified by setting ITSTEP > 0 on CALC.1, and entering a table of time step size versus time on CALC.4.

**CALC.4** (YT(I),FT(I),I=1,ITSTEP)  
Read only if ITSTEP > 0 on CALC.1.  
FORMAT(12E5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, 21-25, etc.	YT(I)	Transient time (s) when the first time step that is FT(I) seconds long begins.
6-10, 16-20, 26-30, etc.	FT(I)	Time step size (sec) from time YT(I).

When this option is used, the time step size for any give time step is calculated by linear interpolation in the table entered on CALC.4.

## Damping For Heat Transfer Solution of Slab Nodes

This input is optional, and is read only when SLDAMP is greater than zero on CALC.2. It allows the user to specify a damping factor on the heat transfer solution of specific slab nodes in a problem. This is a particularly useful capability for cases where certain nodes are not well-coupled with the fluid energy solution.

**CALC.5** NSLGRP Read only if SLDAMP > 0.0 on CALC.2.  
 FORMAT(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5,	NSLGRP	Number of groups of slabs for which the conduction solution will be damped. (NOTE: for this input, a group is defined as a set of slabs with sequential identification numbers.)

**CALC.6** (BSLGRP(I),ESLGRP(I),I=1,NSPGRP) Read only if NSLGRP > 0 on CALC.5.  
 FORMAT(12I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, 21-25, etc.	BSLGRP(I)	The index number of the first slab in the Ith group of slabs with the conduction solution damped.
6-10, 16-20, 26-30, etc.	ESLGRP(I)	The index number of the last slab in the Ith group of slabs with the conduction solution damped.

### 6.2.13 Group OUTP--Output Options

This group allows the user to select the information to be written to the output file. The default is to print everything (i.e., all channels, rods, lateral flow connections, and solid structure nodes at every axial level for the steady state solution and for every time step in a transient). For large problems, or long-running transients, this can result in enormous output files. The user might want to limit the output, either by restricting the number of channels, rods, lateral flow connections, or solid structure nodes for which output will be produced, or by limiting the frequency of output generated during a transient. This input allows considerable flexibility in specifying the output, so that the user can print only what is actually needed in the way of calculational results.

**OUTP.1** AGROUP, NCOUT, NROUT, NGOUT, NWOUT, NALL, NSKIPX, NSKIPT,  
 NPCHAN, NPROD, NPNODE, NPGAP, NPAVG, NPWALL, IPREB, IOTEMP  
 FORMAT(A4, 1X, 5I1, 10I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter OUTP.
6	NCOUT	Flag for channel output: = 0; no channel printout. = 1; channel printout controlled by NPCHAN.
7	NROUT	Flag for rod output: = 0; no rod printout. = 1; rod printout controlled by NPROD.
8	NGOUT	Flag for lateral flow connection output: = 0; no lateral flow connection printout. = 1; lateral flow connection printout controlled by NPGAP.
9	NWOUT	Flag for solid node output: = 0; no solid node printout. = 1; solid node printout controlled by NPWALL.
10	NALL	Flag for output of all information: = 0; output is governed by NCOUT, etc. flags. = 1; everything is printed, regardless of values assigned to other flags.
11-15	NSKIPX	Number of axial nodes to skip when printing results (default is 1, which results in every axial node being printed).
16-20	NSKIPT	Number of time steps to skip between printed results (default is 1, which results in every time step being printed).

21-25	NPCHAN	<p>Flag for number of channels to be printed if NCOUT = 1:</p> <p>= 0; print all channels (default).</p> <p>&gt; 0; print NPCHAN channels (see OUTP.2).</p>
26-30	NPROD	<p>Flag for number of rods to be printed if NROUT = 1:</p> <p>= 0; print all rods (default).</p> <p>&gt; 0; print NPROD rods (see OUTP.3).</p>
31-35	NPNODE	<p>Flag for fuel temperature node printout, if NROUT = 1:</p> <p>= 0; print only fuel centerline temperature, fuel surface temperature, and cladding surface temperature (default).</p> <p>&gt; 0; print cladding surface temperature and all internal node temperatures (used only with the conduction model; NC &gt; 4 on RODS.1).</p>
36-40	NPGAP	<p>Flag for number of lateral flow connections to be printed if NGOUT = 1:</p> <p>= 0; print all lateral flow connections (default).</p> <p>&gt; 0; print NPGAP lateral flow connections (see OUTP.4.)</p>
41-45	NPAVG	<p>Flag for assembly average and channel exit values to be printed:</p> <p>= 0; no additional output (default).</p> <p>= 1; channel exit values only.</p> <p>= 2; assembly-average values only.</p> <p>= 3; channel exit and assembly-average values.</p>

46-50	NPWALL	Flag for solid node printout if NWOUT = 1: = 0; print all solid nodes (default). > 0; print NPWALL solid nodes (see OUTF.5).
51-55	IPREB	Flag for solution description output; = 0; normal iteration summary only (default). > 0; axial rebalancing summary and detailed iteration summary.
56-60	IOTEMP	Flag for output units on temperature: = 0; output in Fahrenheit = 1; output in Celsius.

### User-Defined Output Options

The following input lines, OUTF.2 through OUTF.5, are optional. They are needed only if the corresponding options to limit the output to user-defined items have been specified by setting NPCHAN, NPROD, NPGAP, NPWALL or NSKIPT to non-zero values on OUTF.1. The default in the code is to print all information about all modeled structures.

### Channels to be Printed

This input is optional; it is read only if NCOUT is 1 and NPCHAN is greater than zero on OUTF.1. A total of NPCHAN pairs of assembly index number and channel index number must be specified on OUTF.2.

**OUTF.2** (PRINTA(I),PRINTC(I),I=1,NPCHAN)  
Read only if NCOUT = 1 and NPCHAN > 0 on OUTF.1  
FORMAT(16I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc,	PRINTA(I)	Identification number of assembly containing channel PRINTC(I).
6-10, 16-20, etc.	PRINTC(I)	Index number of the Ith channel in assembly PRINTA(I) for which output will be printed.

### Rods to be Printed

This input is optional; it is read only if NROUT is 1 and NPROD is greater than zero on OUTP.1. A total of NPROD pairs of assembly index number and rod index number must be specified on OUTP.3.

**OUTP.3** (PRINTA(I),PRINTR(I),I=1,NPROD)  
Read only if NROUT = 1 and NPROD > 0 on OUTP.1  
FORMAT(16I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	PRINTA(I)	Index number of assembly containing rod PRINTR(I).
6-10, 16-20, etc.	PRINTR(I)	Index number of the Ith rod in assembly PRINTA(I) for which output will be printed.

### Lateral Flow Connections to be Printed

This input is optional; it is read only if NGOUT is 1 and NPGAP is greater than zero on OUTP.1. A total of NPGAP pairs of assembly index number and gap index number must be specified on OUTP.4.

**OUTP.4** (PRINTA(I),PRINTG(I),I=1,NPGAP)  
Read only if NGOUT = 1 and NPGAP > 0 on OUTP.1  
FORMAT(16I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 11-15, etc.	PRINTA(I)	Index number of assembly containing lateral flow connection PRINTG(I).
6-10, 16-20, etc.	PRINTG(I)	Index number of the Ith lateral flow connection in assembly PRINTA(I) for which output will be printed.

### Solid Nodes to be Printed

This input is optional; it is read only if NWOUT is 1 and NPWALL is greater than zero on OUTP.1. A total of NPWALL slab node index numbers must be specified on OUTP.5.



**OUTP.5** (PRINTW(I),I=1,NPWALL)  
 Read only if NWOUT = 1 and NPWALL > 0 on OUTP.1  
 FORMAT(12I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5, 6-10, 11-15, etc.	PRINTW(I)	Index number of Ith slab node for which temperatures are to be printed.

### Transient Output Interval

This input is optional; it is read only if NSKIPT is greater than zero on OUTP.1. To define the output interval using only NSKIPT, enter zero for TRANT. Otherwise, output will be produced every NSKIPT time steps, or every TRANT seconds, whichever comes first.

**OUTP.6** TRANT Read only if NSKIPT > 1 on OUTP.1  
 FORMAT(F5.3)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	TRANT	Time interval (sec) between successive output time steps in a transient calculation (default is 0.0, which means that the output interval will be controlled by the value of NSKIPT alone).

### 6.2.14 Termination of Input File

Once all groups have been entered, the end of the input stream is indicated by entering an end-flag for the group input name.

**SETIN.1:** AGROUP  
 FORMAT(A4)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-4	AGROUP	Enter ENDD, to signal end of input stream for this case.

### 6.3 View Factor Input Generation: RADGEN Code

The RADGEN code is an auxiliary code which can be used to generate the grey body view factors for radiative heat transfer within an enclosure modeled as an assembly in COBRA-SFS. RADGEN is documented in detail elsewhere (Rector 1987). It is not the purpose of this section to give a complete description of this auxiliary code. However, it is virtually impossible to run any significant fuel storage analysis problem with COBRA-SFS without using view factors of the sort generated with the RADGEN code. The user must therefore be able to use the RADGEN code to obtain view factors that are appropriate for the problem being modeled in COBRA-SFS. This section

presents a general description of the RADGEN code in Section 6.3.1, including a description of the rod arrangements and enclosure geometries for which the code can determine the grey body view factors. Section 6.3.2 gives the input instructions for the RADGEN code for both interactive and batch mode execution. Examples are also presented for three common rod geometries.

### 6.3.1 Description of RADGEN Code

The RADGEN code calculates grey body view factors to be used in the calculation of radiative heat transfer between solid surfaces, as defined by Eq. (3.5) in Section 3.2.4. The method used in RADGEN is an extension of Cox's cross-string correlation approach for square and triangular pitch rod arrays (Cox 1977). It can also be used to define the view factors for an arbitrary enclosure made up of user-defined surfaces, and containing no internal structures to block or reflect radiation exchange among the surfaces.

The black body view factor expressions for a given surface  $A_n$  to another surface  $A_m$  are determined using Hottel's cross-string method for two-dimensional geometries. In this approach, a "view" from surface  $n$  to surface  $m$  is approximated by the concept of anchoring two strings at each end of surface  $n$  and tying one to the nearer end of surface  $m$  and the other to the farther end of surface  $m$ . This is illustrated in Figure 6.6. The endpoints of surface  $n$  are labeled  $a$  and  $b$ , and the endpoints of surface  $m$  are labeled  $c$  and  $d$ , so the strings from  $a$  to  $c$  and from  $b$  to  $d$  are the "crossed" strings, and the strings from  $a$  to  $d$  and from  $b$  to  $c$  are the 'uncrossed' strings. Note that the strings wrap around any other surfaces lying along the 'line of sight' between the endpoints of surface  $n$  and  $m$ . This wrapping is taken into account in computing the length of a given string.

In Hottel's method, the black body view factor between surface  $n$  and surface  $m$  is calculated as a function of the difference between the sums of the lengths of the crossed and uncrossed strings. The view factor from surface  $n$  to  $m$  is given by the following relation:

$$F_{nm} = \frac{(S_{ac} + S_{bd}) - (S_{ad} + S_{bc})}{2 L_n}$$

The  $S$  terms refer to the lengths of the strings between the endpoints designated by the subscript pair, and  $L_n$  is the length of surface  $n$ . Note that the strings wrap around obstructions in the path from one surface endpoint to another.

In the RADGEN code, an enclosure for radiative heat transfer is treated as a set of discrete surfaces, each of which acts independently in emitting, absorbing, and reflecting radiant energy. The basic assumptions for this radiative heat transfer model are as follows:

1. Each rod and wall surface is isothermal.
2. Surfaces are grey, with uniform emissivity for all rods.
3. Reflected radiation is diffusely distributed.
4. Emitted radiation is diffusely distributed.
5. Radiosity is uniform over each surface segment.
6. Energy not transferred to rod surfaces must go to a wall surface.
7. Wall surfaces correspond to slab nodes in the COBRA-SFS model.

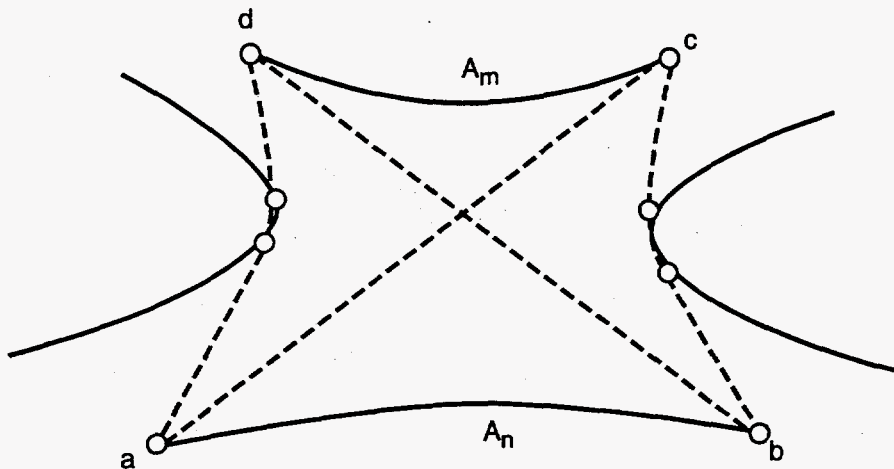


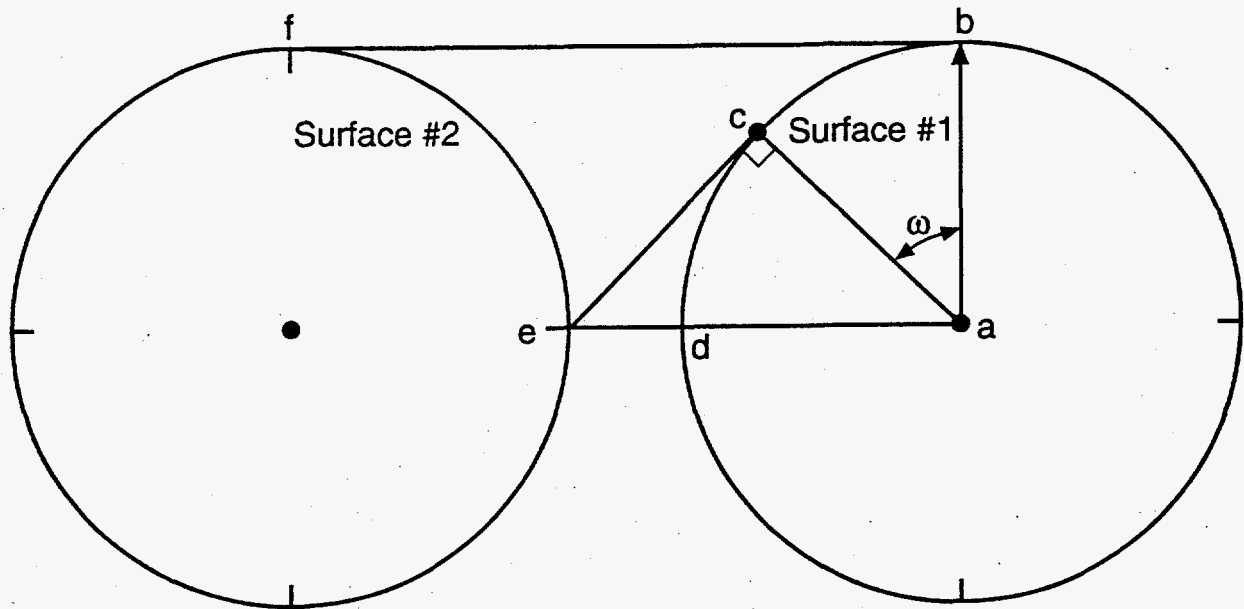
Figure 6.6. Hottel's Crossed String Correlation for Black Body View Factors

In an enclosure containing rods on a square pitch, the rods are divided into quarter segments, so that each rod consists of four surfaces. The enclosure is assumed to be rectangular, and the walls must be modeled with two surfaces each, for a total of eight wall surfaces. In an enclosure containing rods on a triangular pitch, the rods are divided into one-sixth segments. The walls of the enclosure are modeled with a single surface each, for a total of four wall surfaces for a rectangular enclosure, or a total of six for a hexagonal enclosure. The RADGEN code calculates the black body view factors for each rod surface that sees other rods, for each rod surface that sees other rods and the walls, and the walls that see rod surfaces or other wall surfaces.

An example of adjacent quarter-segment surfaces for radiative heat transfer between two rods in a square array is shown in Figure 6-7. For these two surfaces, the view factor expression using Hottel's method is

$$F_{12} = \frac{2(bc + ce) - (bf + de)}{2bd}$$

The string lengths can be expressed in terms of the geometry of the rod array, using the rod radius  $R$ , and the ratio of the array pitch to the rod diameter,  $D_{rod}$  (where the array pitch is the distance between the centers of adjacent rods,  $p$ ). For this example, the uncrossed strings are the lengths  $ed$  and  $fb$ ; the crossed strings are the lengths  $fd$  and  $eb$ . Note that the crossed string lengths



S9509026.1

Figure 6.7. Example of Cross-String Method for Calculating View Factors Between Rod Quarter-Segment Surfaces in the RADGEN Code

wrap some portion of one surface. The length of the wrapped portion of the string is determined by the pitch of the array and the rod diameter. The relevant lengths can be expressed in terms of the array geometry, as follows:

$$ab = ac = ad = R$$

$$bf = 2R \frac{P}{D_{rod}}$$

$$de = 2\left(R \frac{P}{D_{rod}} - R\right)$$

$$\begin{aligned}
ce &= \sqrt{(ae)^2 - (ac)^2} \\
&= \sqrt{\left(2R \frac{p}{D_{rod}} - R\right)^2 - R^2} \\
&= 2R \sqrt{\left(\frac{p}{D_{rod}}\right)^2 - \frac{p}{D_{rod}}}
\end{aligned}$$

The length of the portion of string eb that is wrapped around surface 2 (i.e., the segment cb) can be calculated from plane geometry. First, the angle  $\omega$  must be determined, as

$$\omega = \frac{\pi}{2} - \arctan \frac{ce}{ac}$$

Substituting the length definitions given above yields,

$$\omega = \frac{\pi}{2} - \arctan 2 \sqrt{\left(\frac{p}{D_{rod}}\right)^2 - \frac{p}{D_{rod}}}$$

The length of the arc bc, therefore, is

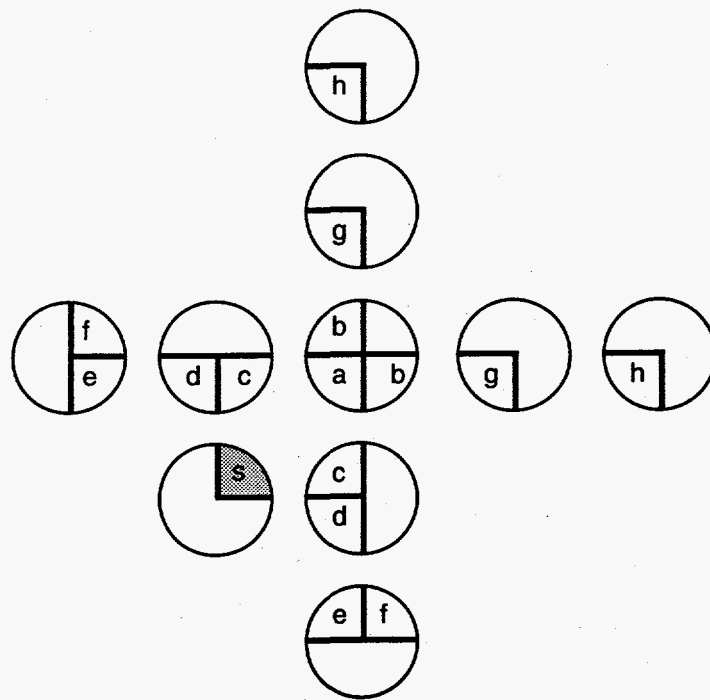
$$bc = R \omega = R \left[ \frac{\pi}{2} - \arctan \left( 2 \sqrt{\left(\frac{p}{D_{rod}}\right)^2 - \frac{p}{D_{rod}}} \right) \right]$$

The expression for the view factor from surface 1 to surface 2, therefore, can be expressed as

$$F_{12} = \frac{2}{\pi} \left[ 2 \sqrt{\left(\frac{p}{D_{rod}}\right)^2 - \frac{p}{D_{rod}}} - \arctan \left( 2 \sqrt{\left(\frac{p}{D_{rod}}\right)^2 - \frac{p}{D_{rod}}} \right) - 2 \frac{p}{D_{rod}} + 1 + \frac{\pi}{2} \right]$$

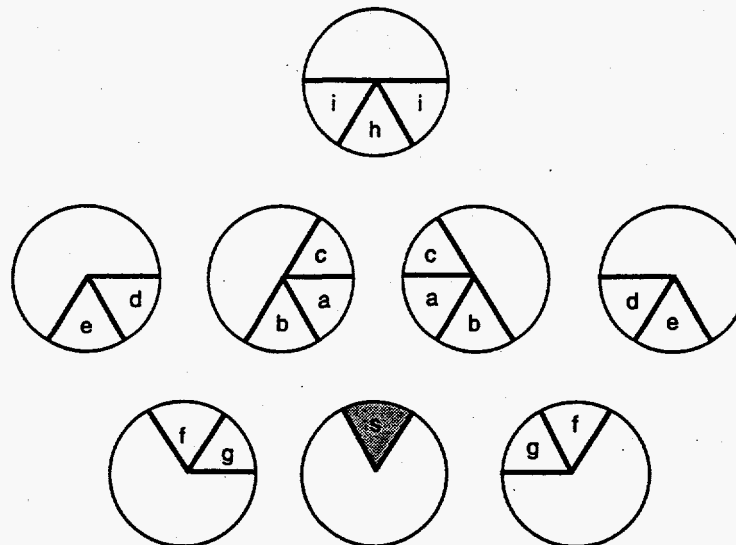
Similar calculations are made in RADGEN for all surfaces seen by a given surface. For rods on a square pitch, a given rod quarter-segment can see up to 15 other rod quarter-segment surfaces, as illustrated in Figure 6.8. For rods on a triangular pitch, a given rod sixth-segment can see up to 17 other rod sixth-segment surfaces, as illustrated in Figure 6.9. The black body view factors are calculated for each of these connections, taking into account rod shadowing due to pitch and diameter.

View factors between the wall surfaces are calculated by arbitrarily dividing the walls into segments of uniform length based on the rod pitch. For rods on a square pitch, a wall segment can potentially see eleven quarter-segment rod surfaces, as shown in Figure 6.10. For rods on a



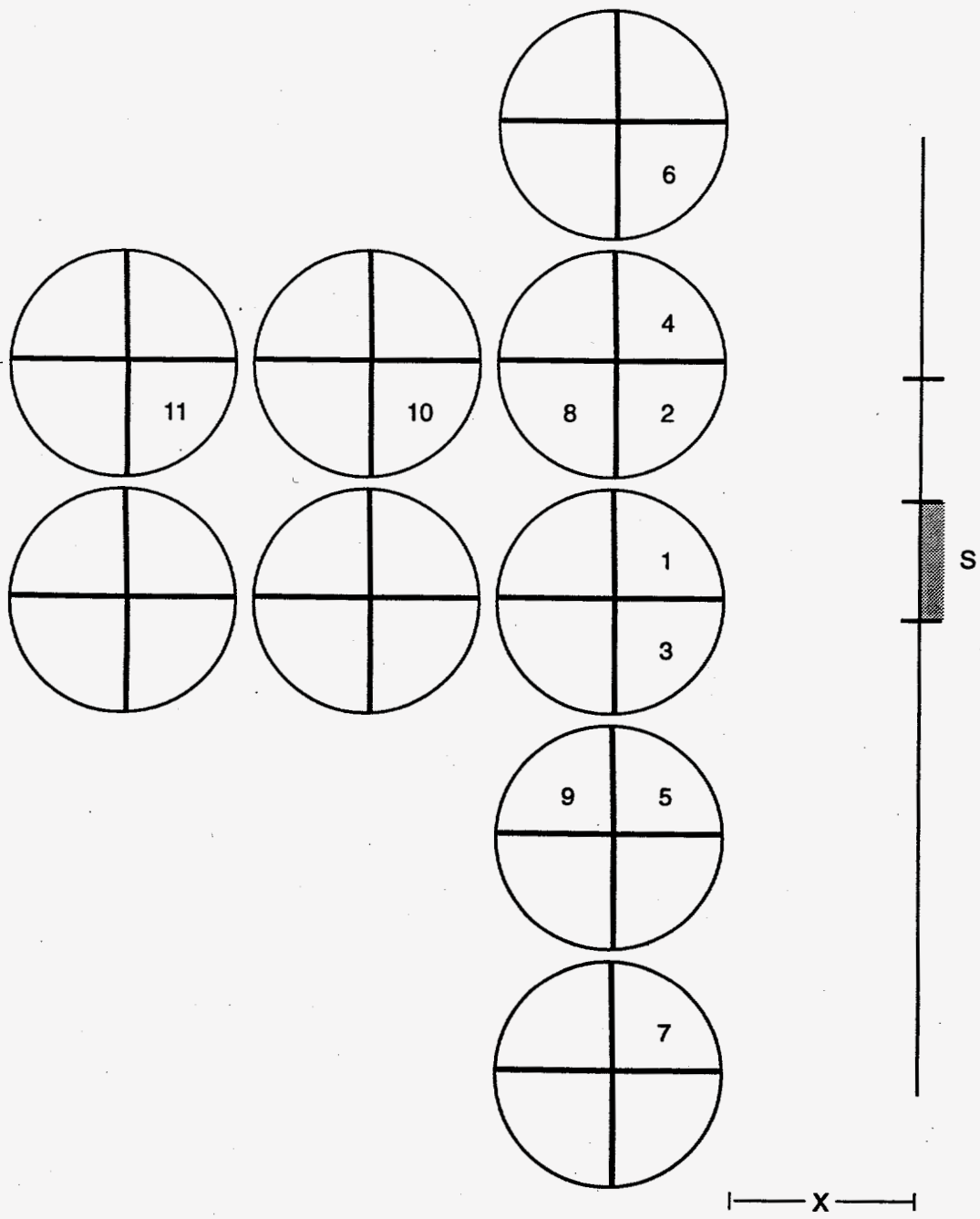
S9503052.17

**Figure 6.8.** Quarter-Section Surfaces on Adjacent Rods With Black Body View Factors to Surface S in a Square Array Rod Bundle



S9503052.18

**Figure 6.9.** One-Sixth Section Surfaces on Adjacent Rods With Black Body View Factors to Surface S in a Triangular Array Rod Bundle



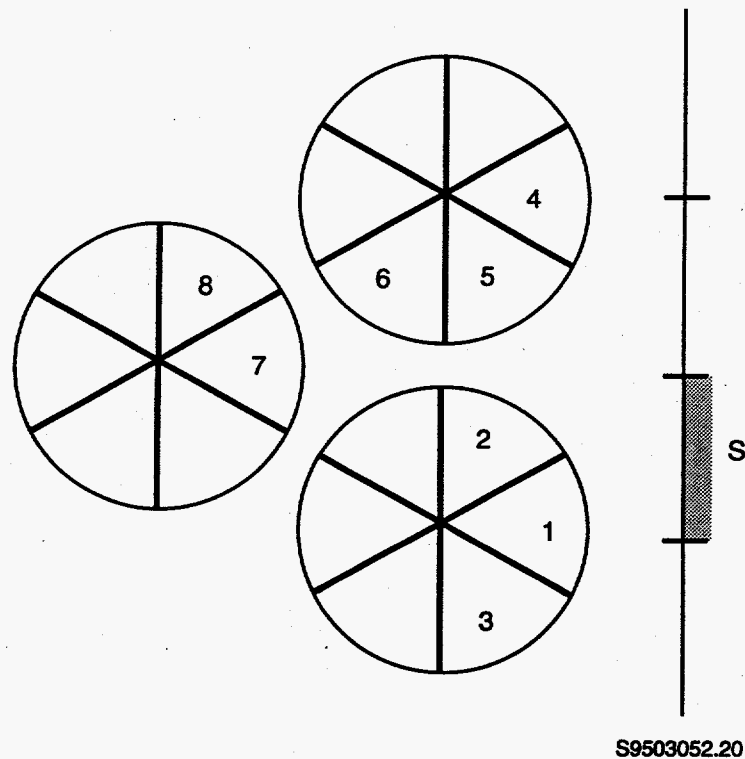
S9503052.19

**Figure 6.10.** Black Body View Factors From Wall Surface S to Quarter-Section Surfaces on Rods in a Square Array.

triangular pitch, a wall segment can see up to eight sixth-segment rod surfaces, as shown in Figure 6.11. This applies to both rectangular canisters and hexagonal canisters for rods on a triangular pitch.

Some adjustments must be made for the corner effects, and view factors between wall segments have to be calculated, as well. The grey body exchange factors for these surfaces are determined by solving Eq. (3.6), as shown in Section 2.4.2. This forms a set of  $N^2$  equations, where  $N$  is the total number of surfaces in the enclosure, which is solved numerically in RADGEN using successive over-relaxation. The grey body view factors for each rod and wall (four for each rod and two for each wall in the square array case, and six for each rod and one for each wall in the triangular array case) are summed to obtain the view factor input expected by COBRA-SFS for a rod assembly.

The arrays of view factors are written to an output file named TAPE10 by the RADGEN code, and this file can be used for input to the COBRA-SFS model of the assembly. If the COBRA-SFS model consists of a lumped rod and channel model of the assembly, rather than a detailed model in which every rod and subchannel is represented explicitly, RADGEN can be used to combine the rod view factors to match the lumping used in the COBRA-SFS input. In that case, the TAPE10 output



**Figure 6.11.** Black Body View Factors From Wall Surface S to One-Sixth Section Surfaces on Rods in a Triangular Array



containing the view factors for the complete assembly is not used as input to COBRA-SFS. It is instead renamed TAPE11 and used as input to RADGEN for a new calculation to generate the lumped TAPE10.

When this option is used in RADGEN, the greybody view factors are not calculated again, but are read from the previously created TAPE10, renamed TAPE11. The user supplies a description of the rod lumping appropriate to the problem, and RADGEN combines the view factors to make a new TAPE10 output file. This second TAPE10 is used as input to COBRA-SFS for the radiation modeling. The lumping calculation can be done in the same step as the TAPE10 generation for the full assembly. In that case, the view factors for the lumped geometry are added to the TAPE10 file, as a second group of view factors after the view factors for the full assembly. This means that the full assembly view factors will be identified in TAPE10 as radiation group 1, and the lumped geometry view factors will be group 2.

### 6.3.2 RADGEN Input Instructions

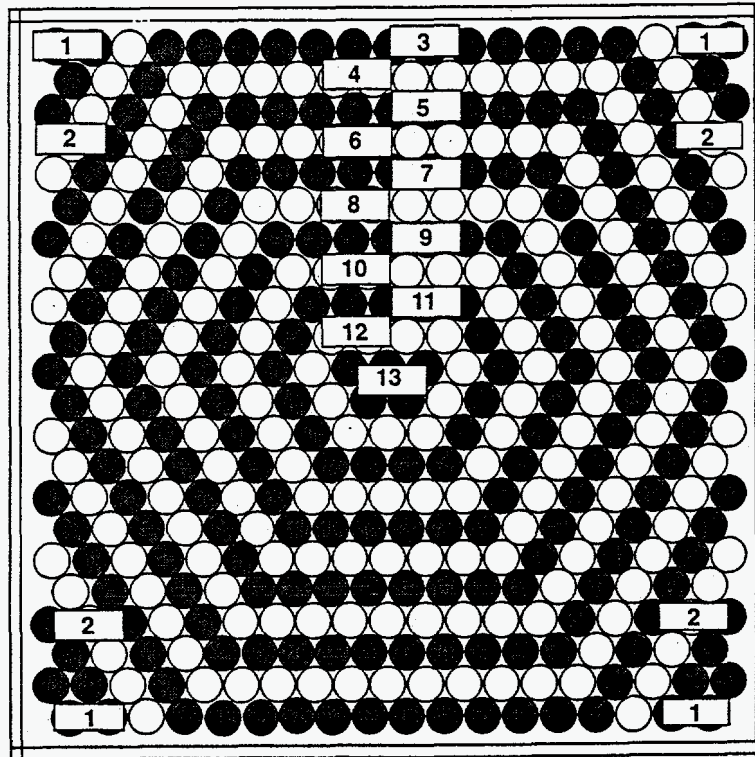
The RADGEN code is designed to accept input in interactive mode. However, it also has the option to read input from an external file. This option can be used in the interactive mode, but the code can also be run in batch mode. The decision as to which mode to use must be made when the code is compiled, by setting the variable flag "ibatch" at the top of program RADGEN. If  $ibatch=0$ , input will be interpreted in interactive mode. If  $ibatch=1$ , all input will be read from logical unit 9, a file named *input*. The input required by the RADGEN code is essentially the same in both cases, but is organized slightly differently. Therefore, the input instructions for the two options are presented separately below. In the interactive mode, the input is entered format-free, with entries separated by commas if more than one variable is called for on a given line. In the batch mode, however, the input file must be in the format given in the input instructions.

The major portion of the input to RADGEN is concerned with describing the geometry of the surfaces for which the view factors are to be calculated. For rod arrays, the code needs to know the number of rod rows and columns. In a square array rod bundle, the numbers of rows and columns are usually the same (i.e., a 15 x 15 bundle has 15 rows and 15 columns). In a triangular array within a square canister, the matter is slightly less straightforward. Depending on the angle one chooses to view the array, one might define different numbers of rows and columns, and in any case, not all rows (or columns) are of the same length. The convention used in RADGEN for the orientation of the triangular array within a square canister is as illustrated in Figure 6.12.

In this example, there are 22 rod rows in the array. The number of columns in a triangular array within a square canister is defined as the number of rods in the longer rows; in this case, 19 rods.

For a triangular array of rods in a hexagonal canister, there is no meaningful distinction between rows and columns. Because of the regularity of the triangular array within a hexagonal bundle, defining the number of rod rows automatically defines the number of columns, so the input requires only that the user define the number of rows.

In the RADGEN code it is assumed that the rods in the array have been numbered according to the pattern illustrated in Figure 6.12. Starting in the upper left-hand corner of the top row, the rods



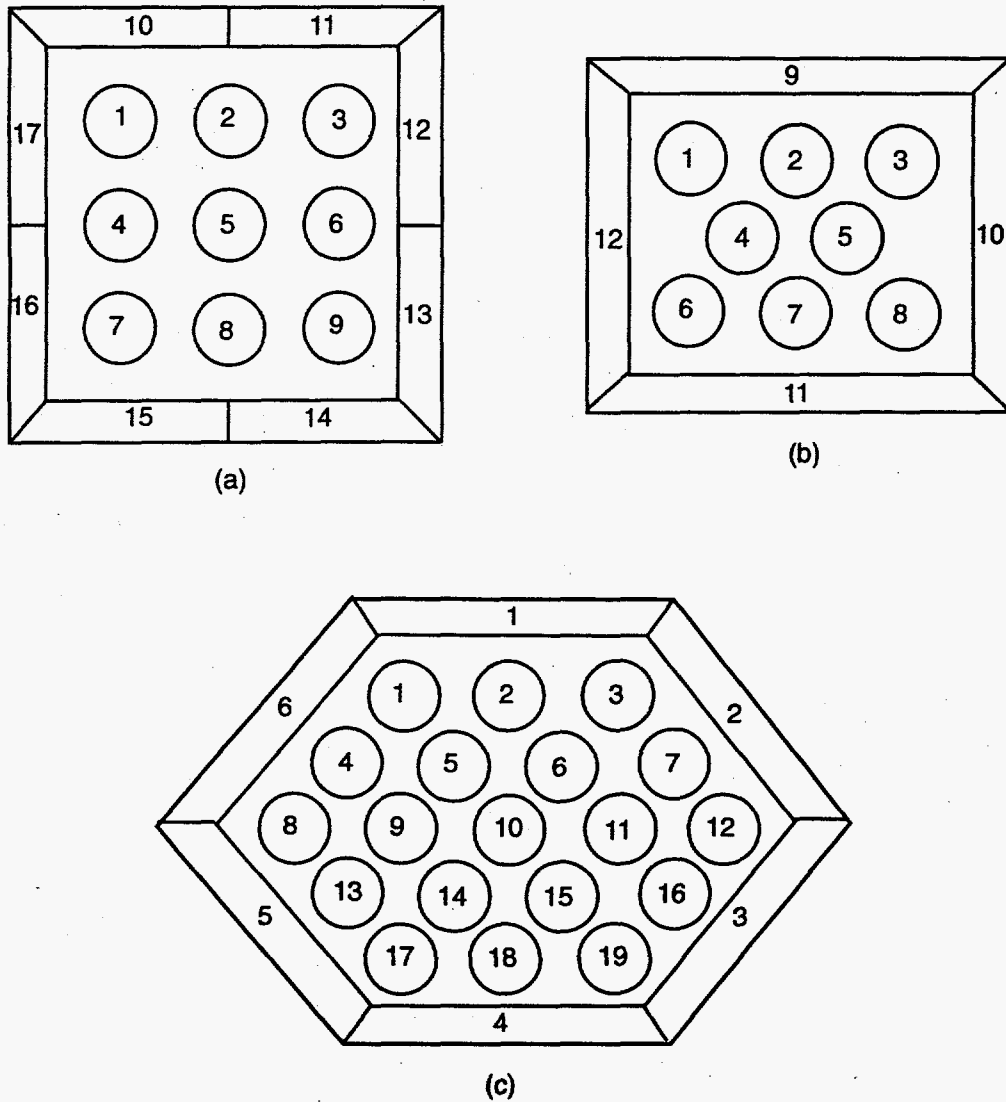
S9506018.2

Figure 6.12. Triangular Array of Rods within a Square Canister

are numbered from left to right, row by row. The wall nodes are assigned numbers starting with  $nrod + 1$ , where  $nrod$  is the number of rods. For a square canister containing a square array of rods, the walls are assumed to have two nodes per face, as shown in Figure 6.13(a). For a square canister containing a triangular array of rods, the walls are assumed to have only one node per face, as shown in Figure 6.13(b). For a triangular array of rods in a hexagonal canister, the walls are assumed to consist of one node per face, as illustrated in Figure 6.13(c).

It is important for the user to observe this numbering convention in defining the rod geometry with the COBRA-SFS input in group RODS, so that the view factors generated with RADGEN will apply to the correct rods within the array. The COBRA-SFS code checks the view factors for each rod array to determine that the factors sum to unity within an assembly. Error messages from COBRA-SFS relating to improper view factors will in most cases be due to inconsistencies in the numbering convention between the input for RADGEN and that for COBRA-SFS.

In many applications, the COBRA-SFS model includes rod arrays that are not complete bundles, either because a bundle is a section of symmetry, or because the rod geometry has been lumped into a number of average rods in order to reduce the number of nodes required for a given model. For such cases, RADGEN can be used to generate view factors for the section of symmetry or the lumped assembly model, but the first step in all cases is the calculation of view factors for the full array. The user must take care to adhere to the rod numbering convention illustrated above in the process of generating view factors for lumped assemblies or sections of symmetry, or the resultant



S9506018.9

**Figure 6.13.** Examples of Rod Numbering Convention Assumed in RADGEN

view factors will not conform to the model geometry, and probably will not conserve energy in radiative heat transfer. Illustrative examples are presented below for the two cases of modeling a section of symmetry and modeling an array with lumped rods. The procedure is not complicated, but it can be confusing if not followed through step by step.



corresponds to rod No. 17 of the full bundle; rod No. 4 corresponds to rod No. 31 of the full bundle, and so on. The numbering of the rods in the COBRA-SFS model must bear this relationship to the full bundle in order for the automatic view factor lumping feature to work properly in RADGEN. A similar pattern must be adhered to for the eighth section and sixth section of symmetry.

The lumped rod model for the full bundle illustrated in Figure 6.5 is used here as an example of how view factors are combined for lumped rod geometries. As with the above illustration for a section of symmetry, the first step is to obtain a set of view factors for the full bundle. The tape10 generated using RADGEN is then renamed tape11, and is used as an input file to a second RADGEN run. In this case, however, the code cannot automatically determine the proper view factor summations, since the lumping is purely at the user's discretion. The user must specify by input to RADGEN exactly which rods of the original array correspond to the lumped rods in the COBRA-SFS model.

This is done by defining each lumped rod as a "group," and specifying by rod number the rods in the original array that make up that "group." For the lumped rod model in Figure 6.5, there are a total of 105 groups, which corresponds to the number of lumped rods in the model. Note that even the rods that are not lumped with other rods in this model are also each counted as a separate "group" of lumped rods. Each of these "lumped" rods consists of only one rod, but it is still necessary to identify which rod it corresponds to in the numbering scheme assumed for the original full bundle array. In general, the rod number in the original array will not correspond to the rod number in the COBRA-SFS model.

In this example, rod No. 1 in the COBRA-SFS lumped model corresponds to group 1, and consists of rods 2, 3, 4, 5, 6, and 7 in the full bundle. RADGEN will combine the view factors for these six rods to make the view factors for rod No. 1 in the lumped model. Similarly, rod No. 49 in the COBRA-SFS lumped model corresponds to group 49, and consists of rod 1 in the full bundle. The following is a partial listing of the grouping input required by RADGEN for this case.

Total number of groups:

Q22: 105

input for group 1 (i.e., rod 1 in the COBRA-SFS lumped model):

Q23: 6

Q24: 2,3,4,5,6,7

Q25: 0

input for group 2 (i.e., rod 2 in the COBRA-SFS lumped model):

Q23: 6

Q24: 9,10,11,12,13,14

Q25: 0

input for group 3 (i.e., rod 3 in the COBRA-SFS lumped model):

Q23: 6

Q24: 30,45,60,75,90,105

Q25: 0

input for group 4 (i.e., rod 4 in the COBRA-SFS lumped model):

Q23: 6

Q24: 135,150,165,180,195,210

Q25: 0

input for group 5 (i.e., rod 5 in the COBRA-SFS lumped model):

Q23: 6

Q24: 219,220,221,222,223,224

Q25: 0

input for group 6 (i.e., rod 6 in the COBRA-SFS lumped model):

Q23: 6

Q24: 212,213,214,215,216,217

Q25: 0

input for group 7 (i.e., rod 7 in the COBRA-SFS lumped model):

Q23: 6

Q24: 121,136,151,166,181,196

Q25: 0

input for group 8 (i.e., rod 8 in the COBRA-SFS lumped model):

Q23: 6

Q24: 16,31,46,61,76,91

Q25: 0

input for group 9 (i.e., rod in the COBRA-SFS lumped model):

Q23: 5

Q24: 18,19,20,21,22

Q25: 0

.

.

.

input for group 34 (i.e., rod 34 in the COBRA-SFS lumped model):

Q23: 2

Q24: 69,70

Q25: 0

input for group 35 (i.e., rod 35 in the COBRA-SFS lumped model):

Q23: 2

Q24: 86,101

Q25: 0

.

.

.

input for group 94 (i.e., rod 94 in the COBRA-SFS lumped model):

Q23: 1  
Q24: 143  
Q25: 0

input for group 105 (i.e., rod 105 in the COBRA-SFS lumped model):

Q23: 1  
Q24: 113  
Q25: 0

The RADGEN code uses the input describing the correspondence between the full array and the lumped rod model to combine the view factors. The example given here is for lumping that includes only whole rods in a given group. If the rod lumping pattern selected for the COBRA-SFS model includes parts of rods in some lumped rods, that too can be accommodated in the view factor summation process, by using the input for Q25 and Q26.

When RADGEN is used to generate the view factors for an enclosure that does not contain rods, the input is greatly simplified, compared to that required for enclosures containing rod arrays. In such a case, the user only has to describe the geometry of the enclosure, by specifying the number of wall surfaces, and defining their emissivities, and their geometric relationship to each other. The enclosure can be of virtually any shape; the only restriction is that all wall surfaces (which must correspond to slab nodes in the COBRA-SFS input) must be flat or concave. That is, a wall surface cannot shadow itself or other surfaces from the line-of-sight view of any other surface.

The specification of the number of wall surfaces and their emissivities is relatively straightforward, but defining the geometry may require a moment's thought. The user must supply the x- and y-coordinates of the endpoints of each wall surface on a Cartesian grid. The grid is essentially arbitrary, since the x- and y-coordinates are meaningful only in relation to each other in this application. The simplest approach is to place the left endpoint of wall surface 1 at the origin, and specify all other locations relative to that endpoint.

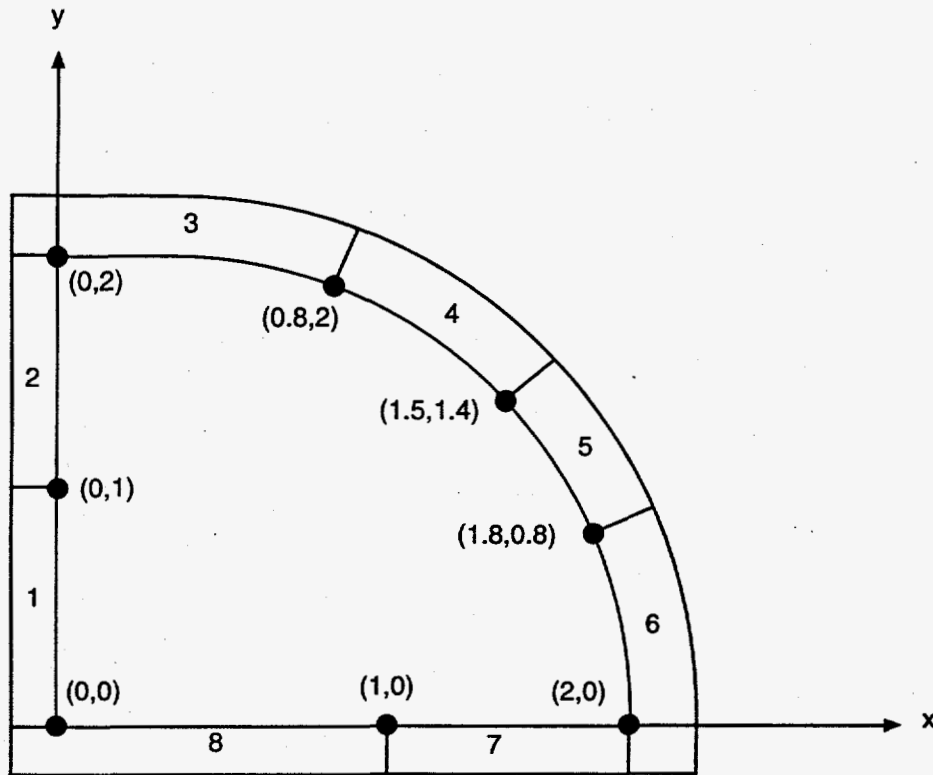
The orientation of the enclosure geometry on the Cartesian grid is up to the user, but in most cases it will be most convenient to orient it with as many surfaces as possible perpendicular or parallel to the x- or y-axis. Figure 6.15 shows an example of a typical case, with an enclosure consisting of both straight and curved surfaces. For convenience, the straight surfaces are laid on the x-y axes. The endpoints of the wall surface nodes making up the curved surface can be determined by inspection, or by application of simple plane geometry relations.

number of wall surfaces (NWALL):

Q12: 8

x- and y-coordinates for first endpoint of wall surface 1:

Q13: 0.0,0.0,1.0



S9506018.10

Figure 6-15. Example of Orientation of Enclosure on a Cartesian Grid

x- and y-coordinates for first endpoint of wall surface 2:

Q13: 0.0,1.0,1.0

x- and y-coordinates for first endpoint of wall surface 3:

Q13: 0.0,2.0,1.15

x- and y-coordinates for first endpoint of wall surface 4:

Q13: 0.8,2.0,0.95

x- and y-coordinates for first endpoint of wall surface 5:

Q13: 1.5,1.4,0.8

x- and y-coordinates for first endpoint of wall surface 6:

Q13: 1.8,0.8,0.8

x- and y-coordinates for first endpoint of wall surface 7:

Q13: 2.0,0.0,1.0

x- and y-coordinates for first endpoint of wall surface 8:

Q13: 1.0,0.0,1.0



The code assumes that the second endpoint of node N coincides with the first endpoint of node N+1, with the single exception that the second endpoint of the last node (NWALL) coincides with the first endpoint of node 1. Therefore, it is only necessary to specify the first endpoint for each wall node.

For a surface N that lies on a straight line on the Cartesian grid, the length PERIM(N) can be entered as zero, and the correct length will be calculated in the code from the specified endpoints of surfaces N and N+1. For curved surfaces, such as surfaces 3, 4, 5, and 6 in the example in Figure 6.14, the user must supply the correct length. This allows the user to model a surface of any curvature with a relatively few number of wall nodes.

The RADGEN code reads from and writes to a number of files, providing informative output for the user, as well as the view factors needed for COBRA-SFS input. A summary of these files is given in Table 6.3.

### 6.3.2.1 Input Instructions for Interactive Mode

The RADGEN code queries the user for the appropriate input, giving the name of the variables required, and their definition. These instructions list each RADGEN query, note the interdependence of selected input options, and define the variables requested. Interactive input is entered format free, with individual variable entries on a line separated by commas. Note that it is possible to use the "batch mode" input file (lun 9, named "input"), in the interactive mode, simply by answering Q1 with an "n." The input supplied on lun 9 must be formatted, however, following the input instructions in Section 6.3.2.3.

Table 6.3. Input and Output Files in RADGEN

Logical unit number	File name	Description
7	output	summary of results
8	tape8	detailed results
9	input	input file for batch mode execution
10	tape10	view factors for COBRA-SFS input
11	tape11	former tape10, to be combined in new view factors for lumped model

Q1: INTERACTIVE INPUT? IF NOT, READ FROM TAPE9. (code variable READIN)  
(Y/YES,RET/NO)

<u>Variable</u>	<u>Description</u>
READIN	Flag for input mode; y -- yes, input is interactive. n -- no, read input from file 'input', (lun 9).

If READIN  $\neq$  y, all further input is read from logical unit 9, as in batch mode. See input instructions in Section 6.3.2.3.

Q2: SPECIFY MODE OF OPERATION (code variable IMODE)

<u>Variable</u>	<u>Description</u>
IMODE	Flag for mode of operation; = 0 -- generate view factors for new geometry. = 1 -- read existing view factors from unit 11 and combine according to specified input.

\*\*\*\*\*  
\*\*\*\*\* NOTE: If IMODE = 1, processing skips immediately \*\*\*\*\*  
\*\*\*\*\* to Q20; input lines Q3 through Q19 are not read. \*\*\*\*\*  
\*\*\*\*\*

Q3: ENTER TITLE FOR PROBLEM (UP TO 64 CHARACTERS) (code variable TEXT)

<u>Variable</u>	<u>Description</u>
TEXT	Title for case identification.

Q4: TYPE OF GEOMETRY DESIRED (code variable IGEOM)

<u>Variable</u>	<u>Description</u>
IGEOM	Geometry type flag; = 0 -- rods on square pitch in a rectangular canister = 1 -- rods on triangular pitch in a rectangular canister = 2 -- rods on triangular pitch in a hexagonal canister = 3 -- enclosure without rods, (i.e., open channel)

\*\*\*\*\*  
NOTE: If IGEOM = 3, processing skips to Q12; the input for Q5 through Q11 is required only for enclosures with square or triangular pitch rod arrays.  
\*\*\*\*\*

Q5: ROD PITCH-TO-DIAMETER RATIO (code variable PDR)

<u>Variable</u>	<u>Description</u>
PDR	Rod pitch-to-diameter ratio (dimensionless).

Q6: NUMBER OF ROWS OF RODS (code variable NROWS)

<u>Variable</u>	<u>Description</u>
NROWS	Number of rod rows in the assembly matrix.

Q7: NUMBER OF RODS IN EACH ROW (code variable NCOL)

<u>Variable</u>	<u>Description</u>
NCOL	For square pitch bundle (IGEOM=0), number of rod columns.  For triangular pitch bundle in a square canister (IGEOM=1), number of rods in the longer rows.  For triangular pitch bundle in a hexagonal canister (IGEOM=2), this input is not needed; enter zero  For nonuniform arrays (IGEOM < 3), enter zero; number of rods in each row will be specified in response to Q8.

NOTE: Q8 is processed only if IGEOM < 2 in response to Q4 *and* NCOL = 0 in response to Q7.

Q8: USER SPECIFIED GEOMETRY:  
FOR EACH ROW, ENTER THE BEGINNING LOCATION AND NUMBER OF RODS

<u>Variable</u>	<u>Description</u>
JBEG(I)	Index number of first rod in the Ith row.
NLINE(I)	Number of rods in the Ith row.

Input for Q8 is specified for NROWS pairs of entries.

NOTE: Q9 is processed only if IGEOM = 1 (i.e., for triangular pitch rod array in a rectangular canister).

Q9: RODS IN A SQUARE CANISTER ARE ASSUMED TO BE ARRANGED IN ALTERNATING LONG AND SHORT ROWS.  
INDICATE WHETHER THE TOP ROW IS LONG OR SHORT (code variable IFIRST)

<u>Variable</u>	<u>Description</u>
IFIRST	Flag for triangular array geometry in a square canister; = 1 -- top row is a long row. = -1 -- top row is a short row.

Q10: ROD OUTSIDE DIAMETER (IN.) (code variable DIA)

<u>Variable</u>	<u>Description</u>
DIA	Rod outside diameter (in.).

NOTE: The value for DIA is the nominal diameter of the fuel rods; it is assumed that all rods in the bundle are the same diameter. The effect of different rod diameters (such as the control rod thimbles) is ignored.

Q11: DISTANCE BETWEEN THE EDGE OF THE RODS AND THE CANISTER WALL (IN.)

If IGEOM = 2, it is assumed that all wall gaps are uniform, and only one value is read, in XR(1).

<u>Variable</u>	<u>Description</u>
XR(1)	Clearance distance (in.) between the rods and the walls on all six sides of the hexagonal canister.

If IGEOM = 0 or 1, the wall gap must be specified at the top, right side, bottom, and left side, entered for XR(1), XR(2), XR(3), and XR(4), as shown below.

AT THE TOP (code variable XR[1])

<u>Variable</u>	<u>Description</u>
XR(1)	Clearance distance (in.) between the top row of rods and the canister wall.

ON THE RIGHT SIDE (code variable XR[2])

<u>Variable</u>	<u>Description</u>
XR(2)	Clearance distance (in.) between the right column of rods and the canister wall.

(NOTE: for triangular arrays, this is the distance between the wall and the end rod of a long row.)

AT THE BOTTOM (code variable XR[3])

<u>Variable</u>	<u>Description</u>
XR(3)	Clearance distance (in.) between the bottom row of rods and the canister wall.

ON THE LEFT SIDE (code variable XR[4])

<u>Variable</u>	<u>Description</u>
XR(4)	Clearance distance (in.) between the left column of rods and the canister wall.

(NOTE: for triangular arrays, this is the distance between the wall and the end rod of a long row.)

NOTE: Input for Q12 and Q13 is required only for an enclosure without rods. If IGEOM < 3, processing skips Q12 and Q13, and continues with Q14. Q12 and Q13 are processed only if IGEOM = 3.

Q12: NUMBER OF RADIATION SURFACES SURROUNDING THE CHANNEL (code variable NWALL)

<u>Variable</u>	<u>Description</u>
NWALL	Number of wall surfaces in the enclosure.

Q13: FOR EACH SURFACE, ENTER THE X AND Y COORDINATES FOR THE FIRST ENDPOINT, AND THE SURFACE LENGTH, (IN.) (ENTER ZERO FOR A STRAIGHT LINE.)

<u>Variable</u>	<u>Description</u>
X(I)	X-coordinate for first endpoint coordinate for the Ith surface.
Y(I)	Y-coordinate for first endpoint coordinate for the Ith surface.
PERIM(I)	Perimeter of Ith surface (in.). (NOTE: if the surface is not curved, enter zero, and the code will automatically calculate the length as the distance to the first endpoint of the next node .)

Input for Q13 is required for NWALL surfaces.

Q14: ROD EMISSIVITY (code variable EMROD)

<u>Variable</u>	<u>Description</u>
EMROD	emissivity of rod surfaces. (NOTE: all rods are assumed to have the same emissivity.)

Q15: WALL EMISSIVITY (ENTER ZERO IF VALUES ARE TO BE READ FOR EACH WALL) (code variable EMWALL)

<u>Variable</u>	<u>Description</u>
EMWALL	Emissivity of wall surfaces, if all walls have the same emissivity.

NOTE: Q16 is processed only if NWALL has not already been defined. That is, if IGEOM < 3, and the assembly contains rods on a square or triangular pitch.

Q16: NUMBER OF WALL SURFACES (code variable NWALL)

<u>Variable</u>	<u>Description</u>
NWALL	Number of wall surfaces in an enclosure containing rods; for IGEOM=0, NWALL is set to 8. for IGEOM=1, NWALL is set to 4. for IGEOM=2, NWALL is set to 6.

NOTE: Q17 is processed only if EMWALL = 0.0 is entered in response to Q15. If EMWALL=0.0, the user must enter a value for the emissivity of each wall surface, in response to Q17.

Q17: (EMISSIVITY FOR WALL NO. i, (code variable EMW[i])

<u>Variable</u>	<u>Description</u>
EMW(I)	Emissivity of wall surface I.

The user must respond with the emissivity value for the Ith wall surface, until all NWALL surfaces have been specified.

NOTE: Q18 and Q19 are processed only if this run is to combine existing exchange factors, which will be read from file TAPE11. In that case, IMODE=1 must be specified in response to Q2.

Q18: ENTER TITLE FOR LUMPED GEOMETRY TAPE10 (UP TO 64 CHARACTERS)  
(code variable TITLE)

<u>Variable</u>	<u>Description</u>
TITLE	Title for lumped geometry tape10 output file.

Q19: GEOMETRY TYPE OF UNLUMPED ROD ARRAY (variable IGEOM)

<u>Variable</u>	<u>Description</u>
IGEOM	Geometry type flag; = 0 -- rods on square pitch. = 1 -- rods on triangular pitch in a rectangular canister. = 2 -- rods on triangular pitch in a hexagonal canister. = 3 -- enclosure without rods.

(NOTE: the value of IGEOM in Q19 must be the same as the IGEOM value used to generate the TAPE10 that is renamed TAPE11 for the view factor lumping.)

Q20: INDICATE AUTOMATIC EXCHANGE FACTOR LUMPING OPTIONS  
(code variable ILUMP)

<u>Variable</u>	<u>Description</u>
ILUMP	Flag for view factor lumping; = -1 -- no view factor lumping. = 0 -- user-specified lumping. = 1 -- automatic lumping for half symmetry on diagonal (square pitch). = 2 -- automatic lumping for eighth symmetry (square pitch). = 3 -- automatic lumping for sixth symmetry (triangular pitch).

Q21: NUMBER OF WALL SURFACES IN LUMPED MODEL (code variable LNWALL)

<u>Variable</u>	<u>Description</u>
LNWALL	Number of wall surfaces in the radiation group.

Q22: NUMBER OF GROUPS FOR USER SPECIFIED LUMPING (code variable NGRP)

<u>Variable</u>	<u>Description</u>
NGRP	Number of groups for user specified lumping.  Enter zero if no lumping groups are to be specified.

Note that all three queries, Q20, Q21, and Q22, must be answered, even if no lumping options are desired. If the lumping options are not used, the input value for each of the three queries is zero.

If Q20 and Q21 are processed with nonzero values for ILUMP or NGRP when IMODE = 0, the view factors will be calculated for the detailed rod array, but the output to TAPE10 will be the lumped view factors. The view factors for the detailed rod array will be lost.

NOTE: Q23 through Q26 are processed only if NGRP > 0 in response to Q21. If NGRP is greater than zero, the input lines Q23 through Q26 are processed as a group NGRP times.

Q23: NUMBER OF WHOLE SURFACES IN GROUP i (code variable NWHOLE)

<u>Variable</u>	<u>Description</u>
NWHOLE	Number of whole surfaces in the Ith group.

Q24: LIST THE WHOLE SURFACE NUMBERS (code variable LLOC(j))

<u>Variable</u>	<u>Description</u>
LLOC(J)	Index number of Jth whole surface in the Ith group. (NOTE: NWHOLE entries are expected.)

Q25: NUMBER OF PARTIAL SURFACES IN GROUP  $i$  (code variable NPART)

<u>Variable</u>	<u>Description</u>
NPART	Number of partial surfaces in the $i$ th group

Q26: LIST THE SURFACE NUMBERS AND FRACTION OF SURFACE FOR EACH

<u>Variable</u>	<u>Description</u>
LLOC( $J+N_w$ )	Index number of $J$ th partial surface in the $i$ th group.
AX( $J+N_w$ )	Fraction of the $J$ th surface that is in the $i$ th group.

(NOTE: NPART pairs of entries are expected.  $N_w = N_{\text{WHOLE}}$ .)

\*\*\*\*\* THIS IS THE END OF THE INPUT REQUIRED IN THE INTERACTIVE MODE \*\*\*\*\*

### 6.3.2.2 Sample Problems for RADGEN

The following sample problems illustrate the interactive input for the RADGEN code. The problems are for the rod geometries shown in Figure 6.13. Test case 1 is a 3 x 3 square array of rods in a square canister (Figure 6.13[a]). Test case 2 is a triangular array of rods in a square canister (Figure 6.13[b]). Test case 3 is for a triangular array of rods in a hexagonal canister (Figure 6.13[c]).

In all three cases, the rod diameter is 0.422 in., and the pitch-to-diameter ratio is 1.2891 for cases 1 and 3. Case 2 models consolidated rods, and the pitch-to-diameter ratio is 1.0. The rod emissivity is assumed to be 0.8, and the wall emissivity is 0.2 in all three cases. For case 1, the gap between the walls and the rods is 0.1 in.; for case 2, the gap is 0.1565 in. at the top and bottom, and 0.1 in. at the left and right sides.

#### INTERACTIVE PROCESS FOR RADGEN TEST CASE 1:

INTERACTIVE INPUT? IF NOT, READ FROM INPUT (lun 9) (code variable READIN)  
(Y/YES,RET/NO)

y

SPECIFY MODE OF OPERATION (code variable IMODE)

0 - GENERATE NEW EXCHANGE FACTORS

1 - LUMP EXISTING EXCHANGE FACTORS (TO BE READ FROM TAPE11)

0

ENTER TITLE FOR PROBLEM (UP TO 64 CHARACTERS) (code variable TEXT)

RADGEN test case 1 (3x3 square array in a square canister)

TYPE OF GEOMETRY DESIRED (code variable IGEOM)

0 - RODS ON SQUARE PITCH IN A RECTANGULAR CAN

1 - RODS ON TRIANGULAR PITCH IN RECTANGULAR CAN  
OR ANY OTHER NON-HEXAGONAL ENCLOSURE

2 - RODS ON TRIANGULAR PITCH IN A HEXAGONAL CAN

3 - ARBITRARY UNOBSTRUCTED ENCLOSURE; NO RODS

0



ROD PITCH-TO-DIAMETER RATIO (code variable PDR)  
 1.2891  
 NUMBER OF ROWS OF RODS (code variable NROWS)  
 3  
 NUMBER OF RODS IN EACH ROW (code variable NCOL)  
 3  
 ROD OUTSIDE DIAMETER (IN.) (code variable DIA)  
 .422  
 DISTANCE BETWEEN THE EDGE OF THE RODS AND THE CANISTER WALL (IN.)  
   AT THE TOP (code variable XR(1))  
 .2  
   ON THE RIGHT SIDE (code variable XR(2))  
 .2  
   AT THE BOTTOM (code variable XR(3))  
 .2  
   ON THE LEFT SIDE (code variable XR(4))  
 .2  
 ROD EMISSIVITY (code variable EMROD)  
 0.8  
 WALL EMISSIVITY (ENTER ZERO IF VALUES ARE TO BE READ FOR EACH WALL)  
   (code variable EMWALL)  
 0.2  
 NUMBER OF WALL SURFACES (code variable NWALL)  
 8  
 INDICATE AUTOMATIC EXCHANGE FACTOR LUMPING OPTIONS (code variable ILUMP)  
   -1 - NO VIEW FACTOR LUMPING  
   0 - USER SPECIFIED LUMPING  
   1 - AUTOMATIC LUMPING FOR HALF SYMMETRY ALONG DIAGONAL (SQUARE PITCH)  
   2 - AUTOMATIC LUMPING FOR EIGHTH SYMMETRY (SQUARE PITCH)  
   3 - AUTOMATIC LUMPING FOR SIXTH SYMMETRY (TRIANGULAR PITCH)  
 -1

\*\*\*\*\*

**INTERACTIVE PROCESS FOR RADGEN TEST CASE 2:**

INTERACTIVE INPUT? IF NOT, READ FROM INPUT (lun 9) (code variable READIN)  
 (Y/YES,RET/NO)

y

SPECIFY MODE OF OPERATION (code variable IMODE)  
   0 - GENERATE NEW EXCHANGE FACTORS  
   1 - LUMP EXISTING EXCHANGE FACTORS (TO BE READ FROM TAPE11)

0

ENTER TITLE FOR PROBLEM (UP TO 64 CHARACTERS) (code variable TEXT)

RADGEN test case 2 (triangular array in a square canister)

TYPE OF GEOMETRY DESIRED (code variable IGEOM)

- 0 - RODS ON SQUARE PITCH IN A RECTANGULAR CAN
- 1 - RODS ON TRIANGULAR PITCH IN RECTANGULAR CAN  
OR ANY OTHER NON-HEXAGONAL ENCLOSURE
- 2 - RODS ON TRIANGULAR PITCH IN A HEXAGONAL CAN
- 3 - ARBITRARY UNOBSTRUCTED ENCLOSURE; NO RODS

1

ROD PITCH-TO-DIAMETER RATIO (code variable PDR)

1.0

NUMBER OF ROWS OF RODS (code variable NROWS)

3

NUMBER OF RODS IN EACH ROW (code variable NCOL)  
(FOR TRIANGULAR ARRAY IN A RECTANGULAR CANISTER, ENTER THE  
NUMBER OF RODS IN THE LONGER ROW)

(FOR IRREGULAR TRIANGULAR ARRAY IN A  
NON-HEXAGONAL CANISTER, THE NUMBER  
OF RODS MUST BE SPECIFIED FOR EACH  
ROW; ENTER ZERO FOR ncol)

3

RODS IN A RECTANGULAR CANISTER ARE ASSUMED TO BE ARRANGED IN ALTERNATING  
LONG AND SHORT ROWS

INDICATE WHETHER THE TOP ROW IS LONG OR SHORT (code variable IFIRST)

- 1 - LONG ROW
- 1 - SHORT ROW

1

ROD OUTSIDE DIAMETER (IN.) (code variable DIA)

0.422

DISTANCE BETWEEN THE EDGE OF THE RODS AND THE CANISTER WALL (IN.)

AT THE TOP (code variable XR(1))

0.1565

ON THE RIGHT SIDE (code variable XR(2))

0.1

AT THE BOTTOM (code variable XR(3))

0.1565

ON THE LEFT SIDE (code variable XR(4))

0.1

ROD EMISSIVITY (code variable EMROD)

0.8

WALL EMISSIVITY (ENTER ZERO IF VALUES ARE TO BE READ FOR EACH WALL)

(code variable EMWALL)

0.2

NUMBER OF WALL SURFACES (code variable NWall)

4

INDICATE AUTOMATIC EXCHANGE FACTOR LUMPING OPTIONS (code variable ILUMP)

- 1 - NO VIEW FACTOR LUMPING
- 0 - USER SPECIFIED LUMPING
- 1 - AUTOMATIC LUMPING FOR HALF SYMMETRY ALONG DIAGONAL (SQUARE PITCH)
- 2 - AUTOMATIC LUMPING FOR EIGHTH SYMMETRY (SQUARE PITCH)
- 3 - AUTOMATIC LUMPING FOR SIXTH SYMMETRY (TRIANGULAR PITCH)

-1

\*\*\*\*\*

#### INTERACTIVE PROCESS FOR RADGEN TEST CASE 3:

INTERACTIVE INPUT? IF NOT, READ FROM INPUT (lun 9) (code variable READIN)  
(Y/YES,RET/NO)

y

SPECIFY MODE OF OPERATION (code variable IMODE)

- 0 - GENERATE NEW EXCHANGE FACTORS
- 1 - LUMP EXISTING EXCHANGE FACTORS (TO BE READ FROM TAPE11)

0

ENTER TITLE FOR PROBLEM (UP TO 64 CHARACTERS) (code variable TEXT)

RADGEN test case 3 (hexagonal array in hexagonal canister)

TYPE OF GEOMETRY DESIRED (code variable IGEOM)

- 0 - RODS ON SQUARE PITCH IN A RECTANGULAR CAN.
- 1 - RODS ON TRIANGULAR PITCH IN RECTANGULAR CAN OR ANY OTHER NON-HEXAGONAL ENCLOSURE
- 2 - RODS ON TRIANGULAR PITCH IN A HEXAGONAL CAN
- 3 - ARBITRARY UNOBSTRUCTED ENCLOSURE; NO RODS

2

ROD PITCH-TO-DIAMETER RATIO (code variable PDR)

1.2891

NUMBER OF ROWS OF RODS (code variable NROWS)

5

ROD OUTSIDE DIAMETER (IN.) (code variable DIA)

0.422

DISTANCE BETWEEN THE EDGE OF THE RODS AND THE CANISTER WALL (IN.)

0.1

ROD EMISSIVITY (code variable EMROD)

0.8

WALL EMISSIVITY (ENTER ZERO IF VALUES ARE TO BE READ FOR EACH WALL)  
(code variable EMWALL)

0.2

NUMBER OF WALL SURFACES (code variable NWall)

6

12 wall nodes, (2 per face), are assumed for triangular array hexagonal bundles.

nwall will be set to 12

INDICATE AUTOMATIC EXCHANGE FACTOR LUMPING OPTIONS (code variable ILUMP)

-1 - NO VIEW FACTOR LUMPING

0 - USER SPECIFIED LUMPING

1 - AUTOMATIC LUMPING FOR HALF SYMMETRY ALONG DIAGONAL (SQUARE PITCH)

2 - AUTOMATIC LUMPING FOR EIGHTH SYMMETRY (SQUARE PITCH)

3 - AUTOMATIC LUMPING FOR SIXTH SYMMETRY (TRIANGULAR PITCH)

-1

\*\*\*\*\*

The view factors produced in these three test cases are as follows:

31.2890.2000.800		RADGEN test case 1 (3x3 square array in a square canister)						
17	1	1	8					
0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000
7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01	7.20346E-01
1.91532E-01	1.55108E-01	2.18470E-02	1.55108E-01	7.51515E-02	1.29645E-02	2.18470E-02	1.29645E-02	9.93075E-04
9.93075E-04	6.67102E-02	8.45465E-03	1.04329E-03	3.36853E-05	3.36853E-05	1.04329E-03	8.45465E-03	6.67102E-02
1.55108E-01	1.07015E-01	1.55108E-01	7.55036E-02	1.06820E-01	7.55036E-02	1.29645E-02	5.56963E-03	1.29645E-02
1.29645E-02	3.80426E-02	3.80426E-02	7.85580E-03	2.78989E-04	5.44135E-04	5.44135E-04	2.78989E-04	7.85580E-03
2.18470E-02	1.55108E-01	1.91532E-01	1.29645E-02	7.51515E-02	1.55108E-01	9.93075E-04	1.29645E-02	2.18470E-02
2.18470E-02	8.45465E-03	6.67102E-02	6.67102E-02	8.45465E-03	1.04329E-03	3.36853E-05	3.36853E-05	1.04329E-03
1.55108E-01	7.55036E-02	1.29645E-02	1.07015E-01	1.06820E-01	5.56963E-03	1.55108E-01	7.55036E-02	1.29645E-02
1.29645E-02	7.85580E-03	2.78989E-04	5.44135E-04	5.44135E-04	2.78989E-04	7.85580E-03	3.80426E-02	3.80426E-02
7.51515E-02	1.06820E-01	7.51515E-02	1.06820E-01	4.02624E-02	1.06820E-01	7.51515E-02	1.06820E-01	7.51515E-02
7.51515E-02	3.98131E-03	3.98131E-03	3.98131E-03	3.98131E-03	3.98131E-03	3.98131E-03	3.98131E-03	3.98131E-03
1.29645E-02	7.55036E-02	1.55108E-01	5.56963E-03	1.06820E-01	1.07015E-01	1.29645E-02	7.55036E-02	1.55108E-01
1.55108E-01	2.78989E-04	7.85580E-03	3.80426E-02	3.80426E-02	7.85580E-03	2.78989E-04	5.44135E-04	5.44135E-04
2.18470E-02	1.29645E-02	9.93075E-04	1.55108E-01	7.51515E-02	1.29645E-02	1.91532E-01	1.55108E-01	2.18470E-02
2.18470E-02	1.04329E-03	3.36853E-05	3.36853E-05	1.04329E-03	8.45465E-03	6.67102E-02	6.67103E-02	8.45465E-03
1.29645E-02	5.56963E-03	1.29645E-02	7.55036E-02	1.06820E-01	7.55036E-02	1.55108E-01	1.07015E-01	1.29645E-02
1.55108E-01	5.44135E-04	5.44135E-04	2.78989E-04	7.85580E-03	3.80426E-02	3.80426E-02	7.85580E-03	2.78989E-04
9.93075E-04	1.29645E-02	2.18470E-02	1.29645E-02	7.51515E-02	1.55108E-01	2.18470E-02	1.55108E-01	9.93075E-04
1.91532E-01	3.36853E-05	1.04329E-03	8.45465E-03	6.67102E-02	6.67102E-02	8.45465E-03	1.04329E-03	3.36853E-05
9.26086E-02	5.28116E-02	1.17369E-02	1.09056E-02	5.52694E-03	3.87299E-04	1.44831E-03	7.55380E-04	9.26086E-02
4.67626E-05	1.28946E-02	1.15187E-03	9.17294E-05	2.30217E-06	1.62914E-06	1.05301E-05	9.17294E-05	4.67626E-05
1.17369E-02	5.28116E-02	9.26086E-02	3.87299E-04	5.52694E-03	1.09056E-02	4.67626E-05	7.55380E-04	1.17369E-02
1.44831E-03	1.15187E-03	1.28946E-02	9.52825E-03	9.17294E-05	1.05301E-05	1.62914E-06	2.30217E-06	1.44831E-03
1.17369E-02	9.17294E-05	9.52825E-03	1.28946E-02	1.15187E-03	9.17294E-05	2.30217E-06	1.62914E-06	1.17369E-02
4.67626E-05	3.87299E-04	1.17369E-02	7.55380E-04	5.52694E-03	5.28116E-02	1.44831E-03	1.09056E-02	4.67626E-05
9.26086E-02	2.30217E-06	9.17294E-05	1.15187E-03	1.28946E-02	9.52826E-03	9.17294E-05	1.05301E-05	9.26086E-02
1.62914E-06	7.55380E-04	1.44831E-03	3.87299E-04	5.52694E-03	1.09056E-02	1.17369E-02	5.28116E-02	1.62914E-06
4.67626E-05	1.62914E-06	1.05301E-05	9.17294E-05	9.52826E-03	1.28946E-02	1.15187E-03	9.17294E-05	4.67626E-05
2.30217E-06	7.55380E-04	4.67626E-05	1.09056E-02	5.52694E-03	3.87299E-04	9.26086E-02	5.28116E-02	2.30217E-06
1.44831E-03	1.05301E-05	1.62914E-06	2.30217E-06	9.17294E-05	1.15187E-03	1.28946E-02	9.52825E-03	1.44831E-03
1.17369E-02	3.87299E-04	4.67626E-05	5.28116E-02	5.52694E-03	7.55380E-04	9.26086E-02	1.09056E-02	1.17369E-02
1.44831E-03	9.17294E-05	2.30217E-06	1.62914E-06	1.05301E-05	9.17294E-05	9.52825E-03	1.28946E-02	1.44831E-03
1.15187E-03	1.09056E-02	1.44831E-03	5.28116E-02	5.52694E-03	7.55380E-04	1.17369E-02	3.87299E-04	1.15187E-03
9.26086E-02	9.52825E-03	9.17294E-05	1.05301E-05	1.62914E-06	2.30217E-06	9.17294E-05	1.15187E-03	9.26086E-02
4.67626E-05	1.28946E-02							4.67626E-05

\*\*\*\*\*

31.0000.2000.800		RADGEN test case 2 (triangular array in a square canister)						
12	1	1	4					
0.2000000	0.2000000	0.2000000	0.2000000					
1.19102E+00	1.02050E+00	1.19102E+00	1.02050E+00					
2.50649E-01	1.72135E-01	1.36512E-03	1.58459E-01	2.32095E-07	7.08450E-02	1.28059E-05	4.88526E-07	
7.10516E-02	8.14953E-06	1.83325E-04	7.52908E-02					

1.72135E-01	1.50555E-01	1.72135E-01	1.21224E-01	1.21224E-01	1.28059E-05	3.39451E-09	1.28059E-05
6.18761E-02	4.12186E-04	4.92025E-08	4.12186E-04				
1.36512E-03	1.72135E-01	2.50649E-01	2.32095E-07	1.58459E-01	4.88526E-07	1.28059E-05	7.08450E-02
7.10516E-02	7.52908E-02	1.83325E-04	8.14953E-06				
1.58459E-01	1.21224E-01	2.32095E-07	8.91584E-02	1.21212E-01	1.58459E-01	1.21224E-01	2.32095E-07
1.73825E-04	2.81068E-09	1.73825E-04	2.99145E-02				
2.32095E-07	1.21224E-01	1.58459E-01	1.21212E-01	8.91584E-02	2.32095E-07	1.21224E-01	1.58459E-01
1.73825E-04	2.99145E-02	1.73825E-04	2.81068E-09				
7.08450E-02	1.28059E-05	4.88526E-07	1.58459E-01	2.32095E-07	2.50649E-01	1.72135E-01	1.36512E-03
1.83325E-04	8.14953E-06	7.10516E-02	7.52908E-02				
1.28059E-05	3.39451E-09	1.28059E-05	1.21224E-01	1.21224E-01	1.72135E-01	1.50555E-01	1.72135E-01
4.92025E-08	4.12186E-04	6.18760E-02	4.12186E-04				
4.88526E-07	1.28059E-05	7.08450E-02	2.32095E-07	1.58459E-01	1.36512E-03	1.72135E-01	2.50649E-01
1.83325E-04	7.52908E-02	7.10516E-02	8.14953E-06				
5.96560E-02	5.19521E-02	5.96560E-02	1.45946E-04	1.45946E-04	1.53922E-04	4.13112E-08	1.53922E-04
1.77814E-02	5.17711E-03	5.98530E-07	5.17711E-03				
7.98585E-06	4.03907E-04	7.37786E-02	2.75422E-09	2.93137E-02	7.98585E-06	4.03907E-04	7.37786E-02
6.04221E-03	1.02209E-02	6.04221E-03	9.65426E-08				
1.53922E-04	4.13112E-08	1.53922E-04	1.45946E-04	1.45946E-04	5.96560E-02	5.19521E-02	5.96560E-02
5.98530E-07	5.17711E-03	1.77814E-02	5.17711E-03				
7.37786E-02	4.03908E-04	7.98585E-06	2.93137E-02	2.75422E-09	7.37786E-02	4.03908E-04	7.98585E-06
6.04221E-03	9.65427E-08	6.04221E-03	1.02209E-02				

\*\*\*\*\*

51.2890.2000.800 RADGEN test case 3 (hexagonal array in hexagonal canister)

31 1 1 12

0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000
0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000	0.2000000
5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01
5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01	5.45770E-01
2.21805E-01	1.45195E-01	1.60206E-03	1.45195E-01	1.17256E-01	2.47041E-02	8.06696E-05	1.60206E-03
2.47041E-02	1.36967E-03	7.79126E-05	1.42236E-06	8.06696E-05	7.79126E-05	3.33967E-05	1.42011E-06
1.42236E-06	1.42011E-06	3.00219E-07	5.74811E-02	6.15541E-04	7.91442E-06	1.71136E-07	7.83008E-09
9.03226E-09	9.03226E-09	7.83008E-09	1.71136E-07	7.91442E-06	6.15541E-04	5.74811E-02	1.42236E-06
1.45195E-01	1.42112E-01	1.45195E-01	2.48748E-02	1.08786E-01	1.08786E-01	2.48748E-02	8.06696E-05
1.34532E-03	2.44175E-02	1.34532E-03	8.06696E-05	3.35802E-05	7.09899E-05	7.09899E-05	3.35802E-05
1.42011E-06	1.39557E-06	1.42011E-06	3.58026E-02	3.58026E-02	5.35734E-04	7.78644E-06	4.84310E-07
5.53067E-08	7.07593E-09	7.07592E-09	5.53067E-08	4.84310E-07	7.78644E-06	5.35734E-04	1.42236E-06
1.60206E-03	1.45195E-01	2.21805E-01	8.06696E-05	2.47041E-02	1.17256E-01	1.45195E-01	1.42236E-06
7.79126E-05	1.36967E-03	2.47041E-02	1.60206E-03	1.42011E-06	3.33967E-05	7.79126E-05	8.06696E-05
3.00219E-07	1.42011E-06	1.42236E-06	6.15541E-04	5.74811E-02	5.74812E-02	6.15541E-04	7.91442E-06
1.71136E-07	7.83008E-09	9.03226E-09	9.03226E-09	7.83008E-09	1.71136E-07	7.91442E-06	1.45195E-01
1.45195E-01	2.48748E-02	8.06696E-05	1.42112E-01	1.08786E-01	1.34532E-03	3.35802E-05	1.45195E-01
1.08786E-01	2.44175E-02	7.09899E-05	1.42011E-06	2.48748E-02	1.34532E-03	7.09899E-05	1.39557E-06
8.06696E-05	3.35802E-05	1.42011E-06	5.35734E-04	7.78644E-06	4.84310E-07	5.53067E-08	7.07592E-09
7.07592E-09	5.53067E-08	4.84310E-07	7.78644E-06	5.35734E-04	3.58026E-02	3.58026E-02	1.17256E-01
1.17256E-01	1.08786E-01	2.47041E-02	1.08786E-01	4.83164E-02	1.00011E-01	1.34532E-03	2.47041E-02
1.00011E-01	9.98788E-02	2.43659E-02	7.79126E-05	1.34532E-03	2.43659E-02	1.30504E-03	7.09899E-05
7.79126E-05	7.09899E-05	3.33967E-05	6.85591E-03	3.55958E-04	2.56835E-05	5.36124E-06	3.82840E-07
4.25203E-07	4.25203E-07	3.82840E-07	5.36124E-06	2.56835E-05	3.55958E-04	6.85591E-03	1.17256E-01
2.47041E-02	1.08786E-01	1.17256E-01	1.34532E-03	1.00011E-01	4.83164E-02	1.08786E-01	7.79126E-05
2.43659E-02	9.98788E-02	1.00011E-01	2.47041E-02	7.09899E-05	1.30504E-03	2.43659E-02	1.34532E-03
3.33967E-05	7.09899E-05	7.79126E-05	3.55958E-04	6.85591E-03	6.85591E-03	3.55958E-04	2.56835E-05
5.36124E-06	3.82840E-07	4.25203E-07	4.25203E-07	3.82840E-07	5.36124E-06	2.56835E-05	8.06696E-05
8.06696E-05	2.48748E-02	1.45195E-01	3.35802E-05	1.34532E-03	1.08786E-01	1.42112E-01	1.42011E-06
7.09899E-05	2.44175E-02	1.08786E-01	1.45195E-01	1.39557E-06	7.09899E-05	1.34532E-03	2.48748E-02
1.42011E-06	3.35802E-05	8.06696E-05	7.78644E-06	5.35734E-04	3.58026E-02	3.58026E-02	5.35734E-04
7.78644E-06	4.84310E-07	5.53067E-08	7.07592E-09	7.07592E-09	5.53067E-08	4.84310E-07	1.60206E-03
1.60206E-03	8.06696E-05	1.42236E-06	1.45195E-01	2.47041E-02	7.79126E-05	1.42011E-06	2.21805E-01
1.17256E-01	1.36967E-03	3.33967E-05	3.00219E-07	1.45195E-01	2.47041E-02	7.79126E-05	1.42011E-06
1.60206E-03	8.06696E-05	1.42236E-06	7.91442E-06	1.71136E-07	7.83008E-09	9.03226E-09	9.03226E-09
7.83008E-09	1.71136E-07	7.91442E-06	6.15541E-04	5.74811E-02	5.74812E-02	6.15541E-04	2.47041E-02
2.47041E-02	1.34532E-03	7.79126E-05	1.08786E-01	1.00011E-01	2.43659E-02	7.09899E-05	1.17256E-01
4.83164E-02	9.98788E-02	1.30504E-03	3.33967E-05	1.08786E-01	1.00011E-01	2.43659E-02	7.09899E-05
2.47041E-02	1.34532E-03	7.79126E-05	2.56835E-05	5.36124E-06	3.82840E-07	4.25203E-07	4.25203E-07

3.82840E-07	5.36124E-06	2.56835E-05	3.55958E-04	6.85591E-03	6.85591E-03	3.55958E-04	
1.36967E-03	2.44175E-02	1.36967E-03	2.44175E-02	9.98788E-02	9.98788E-02	2.44175E-02	1.36967E-03
9.98788E-02	4.57227E-02	9.98788E-02	1.36967E-03	2.44175E-02	9.98788E-02	9.98788E-02	2.44175E-02
1.36967E-03	2.44175E-02	1.36967E-03	2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05
2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05	2.34210E-05	
7.79126E-05	1.34532E-03	2.47041E-02	7.09899E-05	2.43659E-02	1.00011E-01	1.08786E-01	3.33967E-05
1.30504E-03	9.98789E-02	4.83164E-02	1.17256E-01	7.09899E-05	2.43659E-02	1.00011E-01	1.08786E-01
7.79126E-05	1.34532E-03	2.47041E-02	5.36124E-06	2.56835E-05	3.55958E-04	6.85591E-03	6.85591E-03
3.55958E-04	2.56835E-05	5.36124E-06	3.82840E-07	4.25203E-07	4.25203E-07	3.82840E-07	
1.42236E-06	8.06696E-05	1.60206E-03	1.42011E-06	7.79126E-05	2.47041E-02	1.45195E-01	3.00219E-07
3.33967E-05	1.36967E-03	1.17256E-01	2.21805E-01	1.42011E-06	7.79126E-05	2.47041E-02	1.45195E-01
1.42236E-06	8.06696E-05	1.60206E-03	1.71136E-07	7.91442E-06	6.15541E-04	5.74811E-02	5.74811E-02
6.15541E-04	7.91442E-06	1.71136E-07	7.83008E-09	9.03226E-09	9.03226E-09	7.83008E-09	
8.06696E-05	3.35802E-05	1.42011E-06	2.48748E-02	1.34532E-03	7.09899E-05	1.39557E-06	1.45195E-01
1.08786E-01	2.44175E-02	7.09899E-05	1.42011E-06	1.42112E-01	1.08786E-01	1.34532E-03	3.35802E-05
1.45195E-01	2.48748E-02	8.06696E-05	4.84310E-07	5.53067E-08	7.07592E-09	7.07592E-09	5.53067E-08
4.84310E-07	7.78644E-06	5.35734E-04	3.58026E-02	3.58026E-02	5.35734E-04	7.78644E-06	
7.79126E-05	7.09899E-05	3.33967E-05	1.34532E-03	2.43659E-02	1.30504E-03	7.09899E-05	2.47041E-02
1.00011E-01	9.98788E-02	2.43659E-02	7.79126E-05	1.08786E-01	4.83164E-02	1.00011E-01	1.34532E-03
1.17256E-01	1.08786E-01	2.47041E-02	3.82840E-07	4.25203E-07	4.25203E-07	3.82840E-07	5.36124E-06
2.56835E-05	3.55958E-04	6.85591E-03	6.85591E-03	3.55958E-04	2.56835E-05	5.36124E-06	
3.33967E-05	7.09899E-05	7.79126E-05	7.09899E-05	1.30504E-03	2.43659E-02	1.34532E-03	7.79126E-05
2.43659E-02	9.98788E-02	1.00011E-01	2.47041E-02	1.34532E-03	1.00011E-01	4.83164E-02	1.08786E-01
2.47041E-02	1.08786E-01	1.17256E-01	4.25203E-07	3.82840E-07	5.36124E-06	2.56835E-05	3.55958E-04
6.85591E-03	6.85591E-03	3.55958E-04	2.56835E-05	5.36124E-06	3.82840E-07	4.25203E-07	
1.42011E-06	3.35802E-05	8.06696E-05	1.39557E-06	7.09899E-05	1.34532E-03	2.48748E-02	1.42011E-06
7.09899E-05	2.44175E-02	1.08786E-01	1.45195E-01	3.35802E-05	1.34532E-03	1.08786E-01	1.42112E-01
8.06696E-05	2.48748E-02	1.45195E-01	5.53067E-08	4.84310E-07	7.78644E-06	5.35734E-04	3.58026E-02
3.58026E-02	5.35734E-04	7.78644E-06	4.84310E-07	5.53067E-08	7.07592E-09	7.07592E-09	
1.42236E-06	1.42011E-06	3.00219E-07	8.06696E-05	7.79126E-05	3.33967E-05	1.42011E-06	1.60206E-03
2.47041E-02	1.36967E-03	7.79126E-05	1.42236E-06	1.45195E-01	1.17256E-01	2.47041E-02	8.06696E-05
2.21805E-01	1.45195E-01	1.60206E-03	7.83008E-09	9.03226E-09	9.03226E-09	7.83008E-09	1.71136E-07
7.91442E-06	6.15541E-04	5.74811E-02	5.74812E-02	6.15541E-04	7.91442E-06	1.71136E-07	
1.42011E-06	1.39557E-06	1.42011E-06	3.35802E-05	7.09899E-05	7.09899E-05	3.35802E-05	8.06696E-05
1.34532E-03	2.44175E-02	1.34532E-03	8.06696E-05	2.48748E-02	1.08786E-01	1.08786E-01	2.48748E-02
1.45195E-01	1.42112E-01	1.45195E-01	7.07592E-09	7.07592E-09	5.53067E-08	4.84310E-07	7.78644E-06
5.35734E-04	3.58026E-02	3.58026E-02	5.35734E-04	7.78644E-06	4.84310E-07	5.53067E-08	
3.00219E-07	1.42011E-06	1.42236E-06	1.42011E-06	3.33967E-05	7.79126E-05	8.06696E-05	1.42236E-06
7.79126E-05	1.36967E-03	2.47041E-02	1.60206E-03	8.06696E-05	2.47041E-02	1.17256E-01	1.45195E-01
1.60206E-03	1.45195E-01	2.21805E-01	9.03226E-09	7.83008E-09	1.71136E-07	7.91442E-06	6.15541E-04
5.74811E-02	5.74812E-02	6.15541E-04	7.91442E-06	1.71136E-07	7.83008E-09	9.03226E-09	
1.05321E-01	6.56001E-02	1.12784E-03	9.81611E-04	1.25619E-02	6.52213E-04	1.42669E-05	1.45014E-05
4.70592E-05	4.29136E-05	9.82325E-06	3.13569E-07	8.87387E-07	7.01467E-07	7.79089E-07	1.01337E-07
1.43468E-08	1.29650E-08	1.65496E-08	9.44086E-03	7.31206E-04	4.04328E-06	2.55064E-08	1.25056E-09
5.02807E-10	6.66516E-11	8.86007E-11	1.25056E-09	4.86355E-08	4.04328E-06	3.44362E-03	
1.12784E-03	6.56001E-02	1.05321E-01	1.42669E-05	6.52213E-04	1.25619E-02	9.81611E-04	3.13569E-07
9.82325E-06	4.29136E-05	4.70592E-05	1.45014E-05	1.01337E-07	7.79089E-07	7.01467E-07	8.87387E-07
1.65496E-08	1.29650E-08	1.43468E-08	7.31206E-04	9.44086E-03	3.44362E-03	4.04328E-06	4.86355E-08
1.25056E-09	8.86007E-11	6.66516E-11	5.02807E-10	1.25056E-09	2.55064E-08	4.04328E-06	
1.45014E-05	9.81611E-04	1.05321E-01	8.87387E-07	4.70592E-05	1.25619E-02	6.56001E-02	1.43468E-08
7.01467E-07	4.29136E-05	6.52213E-04	1.12784E-03	1.29650E-08	7.79089E-07	9.82325E-06	1.42669E-05
1.65496E-08	1.01337E-07	3.13569E-07	4.04328E-06	3.44362E-03	9.44086E-03	7.31206E-04	4.04328E-06
2.55064E-08	1.25056E-09	5.02807E-10	6.66516E-11	8.86007E-11	1.25056E-09	4.86355E-08	
3.13569E-07	1.42669E-05	1.12784E-03	1.01337E-07	9.82325E-06	6.52213E-04	6.56001E-02	1.65496E-08
7.79089E-07	4.29136E-05	1.25619E-02	1.05321E-01	1.29650E-08	7.01467E-07	4.70592E-05	9.81611E-04
1.43468E-08	8.87387E-07	1.45014E-05	2.55064E-08	4.04328E-06	7.31206E-04	9.44086E-03	3.44362E-03
4.04328E-06	4.86355E-08	1.25056E-09	8.86007E-11	6.66516E-11	5.02807E-10	1.25056E-09	
1.43468E-08	8.87387E-07	1.45014E-05	1.29650E-08	7.01467E-07	4.70592E-05	9.81611E-04	1.65496E-08
7.79089E-07	4.29136E-05	1.25619E-02	1.05321E-01	1.01337E-07	9.82325E-06	6.52213E-04	6.56001E-02
3.13569E-07	1.42669E-05	1.12784E-03	1.25056E-09	4.86355E-08	4.04328E-06	3.44362E-03	9.44086E-03
7.31206E-04	4.04328E-06	2.55064E-08	1.25056E-09	5.02807E-10	6.66516E-11	8.86007E-11	
1.65496E-08	1.01337E-07	3.13569E-07	1.29650E-08	7.79089E-07	9.82325E-06	1.42669E-05	1.43468E-08
7.01467E-07	4.29136E-05	6.52213E-04	1.12784E-03	8.87387E-07	4.70592E-05	1.25619E-02	6.56001E-02
1.45014E-05	9.81611E-04	1.05321E-01	5.02807E-10	1.25056E-09	2.55064E-08	4.04328E-06	7.31206E-04
9.44086E-03	3.44362E-03	4.04328E-06	4.86355E-08	1.25056E-09	8.86007E-11	6.66516E-11	
1.65496E-08	1.29650E-08	1.43468E-08	1.01337E-07	7.79089E-07	7.01467E-07	8.87388E-07	3.13569E-07
9.82325E-06	4.29136E-05	4.70592E-05	1.45014E-05	1.42669E-05	6.52213E-04	1.25619E-02	9.81611E-04
1.12784E-03	6.56001E-02	1.05321E-01	6.66516E-11	8.86007E-11	1.25056E-09	4.86355E-08	4.04328E-06

3.44362E-03	9.44086E-03	7.31206E-04	4.04328E-06	2.55064E-08	1.25056E-09	5.02807E-10	
1.43468E-08	1.29650E-08	1.65496E-08	8.87387E-07	7.01467E-07	7.79089E-07	1.01337E-07	1.45014E-05
4.70592E-05	4.29136E-05	9.82325E-06	3.13569E-07	9.81611E-04	1.25619E-02	6.52213E-04	1.42669E-05
1.05321E-01	6.56001E-02	1.12784E-03	8.86007E-11	6.66516E-11	5.02807E-10	1.25056E-09	2.55064E-08
4.04328E-06	7.31206E-04	9.44086E-03	3.44362E-03	4.04328E-06	4.86355E-08	1.25056E-09	
3.13569E-07	1.01337E-07	1.65496E-08	1.42669E-05	9.82325E-06	7.79089E-07	1.29650E-08	1.12784E-03
6.52213E-04	4.29136E-05	7.01467E-07	1.43468E-08	6.56001E-02	1.25619E-02	4.70592E-05	8.87387E-07
1.05321E-01	9.81611E-04	1.45014E-05	1.25056E-09	5.02807E-10	6.66516E-11	8.86007E-11	1.25056E-09
4.86355E-08	4.04328E-06	3.44362E-03	9.44086E-03	7.31206E-04	4.04328E-06	2.55064E-08	
1.45014E-05	8.87387E-07	1.43468E-08	9.81611E-04	4.70592E-05	7.01467E-07	1.29650E-08	1.05321E-01
1.25619E-02	4.29136E-05	7.79089E-07	1.65496E-08	6.56001E-02	6.52213E-04	9.82325E-06	1.01337E-07
1.12784E-03	1.42669E-05	3.13569E-07	4.86355E-08	1.25056E-09	8.86007E-11	6.66516E-11	5.02807E-10
1.25056E-09	2.55064E-08	4.04328E-06	7.31206E-04	9.44086E-03	3.44362E-03	4.04328E-06	
1.12784E-03	1.42669E-05	3.13569E-07	6.56001E-02	6.52213E-04	9.82325E-06	1.01337E-07	1.05321E-01
1.25619E-02	4.29136E-05	7.79089E-07	1.65496E-08	9.81611E-04	4.70592E-05	7.01467E-07	1.29650E-08
1.45014E-05	8.87387E-07	1.43468E-08	4.04328E-06	2.55064E-08	1.25056E-09	5.02807E-10	6.66516E-11
8.86007E-11	1.25056E-09	4.86355E-08	4.04328E-06	3.44362E-03	9.44086E-03	7.31206E-04	
1.05321E-01	9.81611E-04	1.45014E-05	6.56001E-02	1.25619E-02	4.70592E-05	8.87387E-07	1.12784E-03
6.52213E-04	4.29136E-05	7.01467E-07	1.43468E-08	1.42669E-05	9.82325E-06	7.79089E-07	1.29650E-08
3.13569E-07	1.01337E-07	1.65496E-08	3.44362E-03	4.04328E-06	4.86355E-08	1.25056E-09	8.86007E-11
6.66516E-11	5.02807E-10	1.25056E-09	2.55064E-08	4.04328E-06	7.31206E-04	9.44086E-03	

### 6.3.2.3 Input Instructions for Batch Mode

The RADGEN input is read from logical unit 9, file name "input," when the code is executed in batch mode, or when the option READIN=n is selected in the interactive mode. The input requirements are the same as for the interactive mode, but must be supplied in the formatted input lines described here.

**RADGEN.1** IMODE  
Format(15)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IMODE	Flag for mode of operation;
		= 0 -- generate view factors for new geometry.
		= 1 -- read existing view factors from unit 11 and process according to user input.

\*\*\*\*\*  
 \*\*\*\* IF IMODE = 1, input lines RADGEN.2 through RADGEN.9 are not read \*\*\*\*  
 \*\*\*\* Input processing skips to line RADGEN.10 if IMODE = 1 \*\*\*\*  
 \*\*\*\*\*

**RADGEN.2** TEXT  
Format(16A4)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-64	TEXT	Title for case identification.

**RADGEN.3** IGEOM, PDR, NROWS, NCOL, IFIRST  
 Format(I5,f5.3,3i5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IGEOM	Geometry type flag; = 0 -- rods on square pitch. = 1 -- rods on triangular pitch in a in a rectangular canister. = 2 -- rods on a triangular pitch in a hexagonal canister. = 3 -- enclosure without rods.
6-10	PDR	Rod pitch-to-diameter ratio
11-15	NROWS	number of rod rows.
16-20	NCOL	For square pitch array, number of rod columns.  For triangular pitch array in rectangular canister, number of rods in the longer rows.  For triangular pitch array in hexagonal canister, enter zero.  For nonuniform square pitch or triangular pitch arrays, enter zero (number of rods in each row will be specified on RADGEN.8.)
21-25	IFIRST	Flag for triangular array in rectangular canister; = 1 -- top row is a long row. = -1 -- top row is a short row. = 0 -- not a triangular array in a rectangular canister

**RADGEN.4** DIA, (XR(I),I=1,4) Read only if (IGEOM < 2 and NCOL > 0)  
*or* IGEOM = 2.  
 Format(5F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	DIA	Rod outside diameter (in.).
6-10	XR(1)	Clearance distance (in.) between the top row of rods and canister wall NOTE: for IGEOM = 2, this value is used for all six wall gaps for a triangular array of rods in a hexagonal canister.)



11-15	XR(2)	Clearance distance (in.) between the right column of rods and the canister wall. (NOTE: for triangular arrays in a rectangular canister, this is the distance between the wall and the end rod of a long row.)
16-20	XR(3)	Clearance distance (in.) between the bottom row of rods and the canister wall.
21-25	XR(4)	Clearance distance (in.) between the left column of rods and the canister wall. (NOTE: for triangular arrays in a rectangular canister, this is the distance between the wall and the end rod of a long row.)

**RADGEN.5** JBEG(I), NLINE(I), I=1,NROWS Read only if NCOL = 0  
Format(16I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	JBEG(I)	Index number of first rod in the Ith row.
6-10	NLINE(I)	Number of rods in the Ith row.

Input line RADGEN.5 is read with eight pairs of (JBEG,NLINE) entries per line, until NROWS pairs have been specified. The last line may have fewer than eight pairs, if NROWS is not an even multiple of eight.

**RADGEN.6** NWALL Read only if IGEOM = 3  
Format(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NWALL	Number of wall surfaces in the enclosure.

**RADGEN.7** X(I), Y(I), PERIM(I), I=1,NWALL Read only if IGEOM=3  
Format(3F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	X(I)	left endpoint coordinate for the Ith surface
6-10	Y(I)	right endpoint coordinate for the Ith surface
11-15	PERIM(I)	perimeter of Ith surface, (inches)

Input line RADGEN.7 is read NWALL times.

**RADGEN.8** EMROD, EMWALL  
Format(2F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	EMROD	Emissivity of rod surfaces (NOTE: all rod surfaces are assumed to have the same emissivity).
6-10	EMWALL	Emissivity of wall surfaces, if all walls have the same emissivity.

**RADGEN.9** EMW(I), I=1, NWALL Read only if EMWALL = 0.0  
Format(12F5.0)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	EMW(I)	emissivity of wall surface I

Input line RADGEN.9 is read with 12 entries per line until NWALL values have been specified. The last line may have fewer than 12 entries, if NWALL is not an even multiple of 12.

**RADGEN.10** TITLE Read only if IMODE = 1  
Format(16A4)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-64	TITLE	Title for identification of lumped view factor set.

**RADGEN.11** ILUMP  
Format(I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	ILUMP	Flag for view factor lumping; = -1 -- no lumping of view factors = 0 -- user specified lumping. = 1 -- automatic lumping for half symmetry on diagonal (square pitch). = 2 -- automatic lumping for eighth symmetry (square pitch). = 3 -- automatic lumping for sixth symmetry (triangular pitch).

**RADGEN.12** NGRP, LNWALL Read only if ILUMP > -1  
Format(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	NGRP	Number of groups for user-specified lumping.
6-10	LNWALL	Number of wall surfaces in lumped view factor set.

**RADGEN.13** IDUMY, NTOT Read only if NGRP > 0  
Form at(2I5)

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	IDUMY	Index number of Ith group. (NOTE: groups must be numbered from 1 to NGRP.)
6-10	NTOT	Total number of surfaces in this group.

**RADGEN.14** LLOC(J), AX(J), J=1,NTOT Read only if NGRP > 0  
Format(6(I5,F5.3))

<u>Columns</u>	<u>Variable</u>	<u>Description</u>
1-5	LLOC(J)	Index number of Jth surface in the Ith group
6-10	AX(J)	Fraction of the Jth surface that is contained in the Ith group.

Input line RADGEN.14 is read with six pairs of (LLOC,AX) entries per line, until all NTOT pairs of entries have been specified. The last line may have fewer than six pairs, if NTOT is not an even multiple of six.

Input lines RADGEN.13 and RADGEN.14 are read as a set NGRP times, to define all the lumping of all surfaces in the enclosure.

## 7.0 Validation for Cycle 2

This section presents the results of the validation of the code changes comprising the Cycle 2 release of COBRA-SFS. The size and complexity of the code make it virtually impossible to exercise every feature and all possible combinations of modeling options, but a concerted effort has been made to test out the features in a manner consistent with the expected usage of the code. The new capabilities added in Cycle 2 were explicitly tested by one case or another, and existing models from earlier cycles were also tested to make certain that code improvements have not compromised performance in any way.

The first test case is a benchmark test case, comparing the performance of the Cycle 2 version of the code with the previous release, Cycle 1. The purpose is to compare the results obtained using Cycle 2 with those of Cycle 1 for a case that both versions can solve. These results are presented in Section 7.1, in a code-to-code comparison. For the remainder of the test cases, the comparison is between Cycle 2 code calculations and experimental data. Section 7.2 gives the results for three single-assembly cases, and Section 7.3 presents results for two multiple-assembly storage casks.

The test cases presented in Sections 7.1 through 7.3 were all run on a SUN SPARCstation 2. Additional testing was performed on various alternative platforms, including workstations and a mainframe computer system. In addition to the SUN SPARCstation, the code has been run on IBM's RS6000, an INTEL Pentium workstation, the DEC workstation, the HP9000, and the Apple-Macintosh workstation. Cycle 2 of the code has also been run on a CRAY mainframe, under the uniCOS operating system. Essentially the same version of the code was run on all platforms; the only conversion changes were the time and date routines, and the *loc* function for clearing common blocks and writing the restart dump file.

Three representative test cases were run on the alternative platforms. These were the Mitsubishi single-assembly case, the TN24 unconsolidated cask in vertical orientation, and the PSN VSC-17 ventilated concrete cask model. Section 7.4 compares the results obtained on the various platforms for these three cases.

### 7.1 Benchmark Test: Cycle 1 to Cycle 2

The purpose of the benchmark test is to verify that Cycle 2 of COBRA-SFS produces essentially the same answers as Cycle 1 when presented with the same input. The COBRA-SFS model of the Mitsubishi electrically heated assembly was selected for this illustration, mainly because it is a relatively small problem that uses many code capabilities which might be called into play in a typical cask analysis. It is, therefore, a good test case to verify that improvements in Cycle 2 have not degraded the overall performance of the code for its primary application.

This section is concerned with the comparison of results between Cycle 2 and Cycle 1 for the test case. For this purpose, it is necessary only to know that the assembly is an electrically heated model of a 15 x 15 PWR fuel rod array enclosed in a square canister with nitrogen back-fill. The test

assembly was oriented horizontally, and sealed at both ends (the Mitsubishi assembly is described in detail in Section 7.2.1, in which the Cycle 2 calculational results are compared to the data).

The COBRA-SFS model consisted of a single assembly with the geometry represented by 225 rods, 256 subchannels, and 8 wall nodes. The boundary conditions were defined by specifying a uniform temperature on the outside of the wall nodes, and specifying a zero flow at the top and bottom of the assembly (with ITDP = 4 on OPER.1; see Section 6.2.11).

The same input file was used when executing both the Cycle 1 and the Cycle 2 versions of the code. Only the case title was changed between the two cases, for easier identification in examining the output. In both cases the calculation was run for 20 iterations, and produced a predicted peak clad temperature of 734.8°F. After 20 iterations, both code versions displayed convergence errors of 0.0000 for flow, fluid energy, and rod energy. The total energy error was 1.0000, which in this context means that exactly the same amount of energy was removed at the wall boundary as was added to the system by the heat generating rods.

In the output of fluid conditions, the enthalpies, temperatures, densities, and flow rates in the fluid nodes comprising the subchannels were identical between the two cases. The pressure was slightly different on some subchannels; Cycle 1 predicted 0.0000008 psi, while Cycle 2 predicted 0.0000000 psi. All rod temperatures were identical between the two cases, except for two instances. In one case, a rod was predicted to be at 616.8°F in one node with Cycle 1, while Cycle 2 predicted 616.9°F at that location. In the other case, Cycle 1 predicted 628.3°F, while Cycle 2 predicted 628.4°F for that particular node.

The individual subchannel flow rates were identical between the two cases, to the seven decimal places displayed on the COBRA-SFS output. The total flow rate (i.e., the sum of the inlet flow rates for all subchannels) was predicted in the Cycle 1 calculation to be  $-0.490(10^{-13})$  lbm/sec; Cycle 2 predicted  $0.212(10^{-11})$  lbm/sec. Both values are well within the convergence criteria for the boundary condition of zero flow. This difference is not in any sense significant.

This comparison gives some confidence that the Cycle 2 changes have not introduced significant errors into the solution. However, it is advisable for users of Cycle 1 to perform their own benchmark validation on Cycle 2 of models developed for Cycle 1, as part of installation and check-out of the new version. The results obtained with Cycle 2 should not differ substantially from the results with Cycle 1. The cause of any significant differences should be resolved before attempting to use the new features and models included in Cycle 2.

## 7.2 Single-Assembly Test Cases

Three single-assembly test cases were selected as part of the validation of the Cycle 2 changes to COBRA-SFS. These cases provide relatively simple problems for analysis that nevertheless contain many of the same elements as large multi-assembly cask models. All three assemblies were electrically heated models of PWR fuel in a square canister. Section 7.2.1 describes the validation test for Pacific Northwest Laboratory's (PNL's) Single-Assembly Heat Transfer Test (SAHTT) bundle;

Section 7.2.2 describes the validation of Cycle 2 with the Mitsubishi bundle, and Section 7.2.3 discusses the results obtained for a test assembly built by British Nuclear Fuels, Ltd. (BNFL). In each case, input models of the assemblies were run with Cycle 2 of the code, and the results compared with experimental data.

### 7.2.1 SAHTT 15 x 15 PWR Test Assembly

The SAHTT was developed as part of work at the PNL to investigate heat transfer characteristics of spent light water reactor fuel under dry storage conditions (Bates 1986). The test section, illustrated in Figure 7.1, was an electrically heated model of a 15 x 15 PWR spent fuel assembly, with simulated fuel rods 0.42 in. (1.07 cm) in diameter on a nominal pitch of 0.562 in. (1.4275 cm). Control rod thimbles were modeled with nine unheated rods that had an outer diameter of 0.5 in. (1.27 cm).

The axial heated length was 144 in. (3.658 m). The bundle was oriented horizontally, and back-filled with air or helium. Instrumentation included 98 thermocouples, to measure cladding temperatures and canister surface temperatures. The power generation rate was axially uniform, with total power of 0.5 kW or 1.0 kW, depending on the test.

The COBRA-SFS input consisted of a detailed subchannel model, with 256 subchannels, 225 rods, and 8 slab nodes modeling the canister wall. A cross-section of the model is shown in Figure 7.2. The locations of the rods modeling control rod thimbles are noted on the diagram. Four other rods that modeled fuel rods (identified as 61, 92, 104, and 151 in the diagram) had zero power, due to heater malfunction during testing. These were included in the COBRA-SFS model geometry input as zero-power rods.

To reduce problem size and save computation time, the bundle was modeled axially by taking a slice three nodes wide through the center of the bundle. The axially uniform power distribution in the test resulted in an essentially uniform temperature distribution for the full length of the bundle, and the ends were sufficiently well insulated for losses to be minimal. As a result, the axial temperature profile was essentially flat, with only a slight dip on the profile near the ends.

The most important Cycle 2 feature included in this model is lateral momentum coupling for the cross-flows. This allows the code to calculate the effect of natural convection in the lateral direction, due to the horizontal orientation of the bundle. In Cycle 1, natural convection in the lateral direction could not be modeled, due to the assumption of no crossflow coupling for momentum.

Boundary conditions on the calculations were taken from measured temperatures on the top, bottom, and sides of the test section. At the top, the boundary temperature was 215.9°C (420°F); at the bottom, it was 208.9°C (408°F); on the sides, 209.9°C (410°F). Radiative heat transfer was modeled using rod-to-rod and rod-to-wall exchange factors, assuming a surface emissivity of 0.2 for the walls and 0.6 for the rods. The grey body view factors were calculated using the auxiliary code RADGEN, (Rector 1987; see Section 6.3 for a description of this code).

The heat transfer coefficient correlations used in this input model are based on data obtained for flow through tube bundles or arrays of heated rods. The internal matrix subchannels modeling the test section falls within this data base, but the applicability of the correlations to the subchannels that

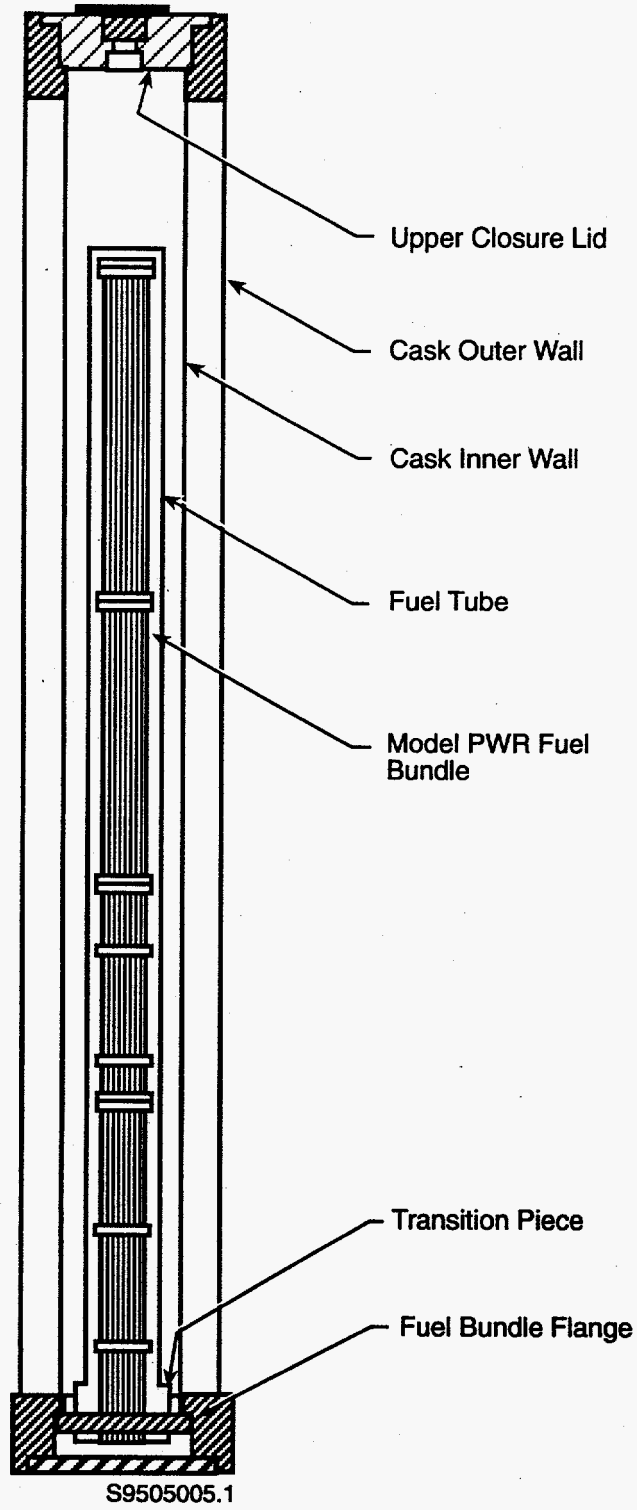
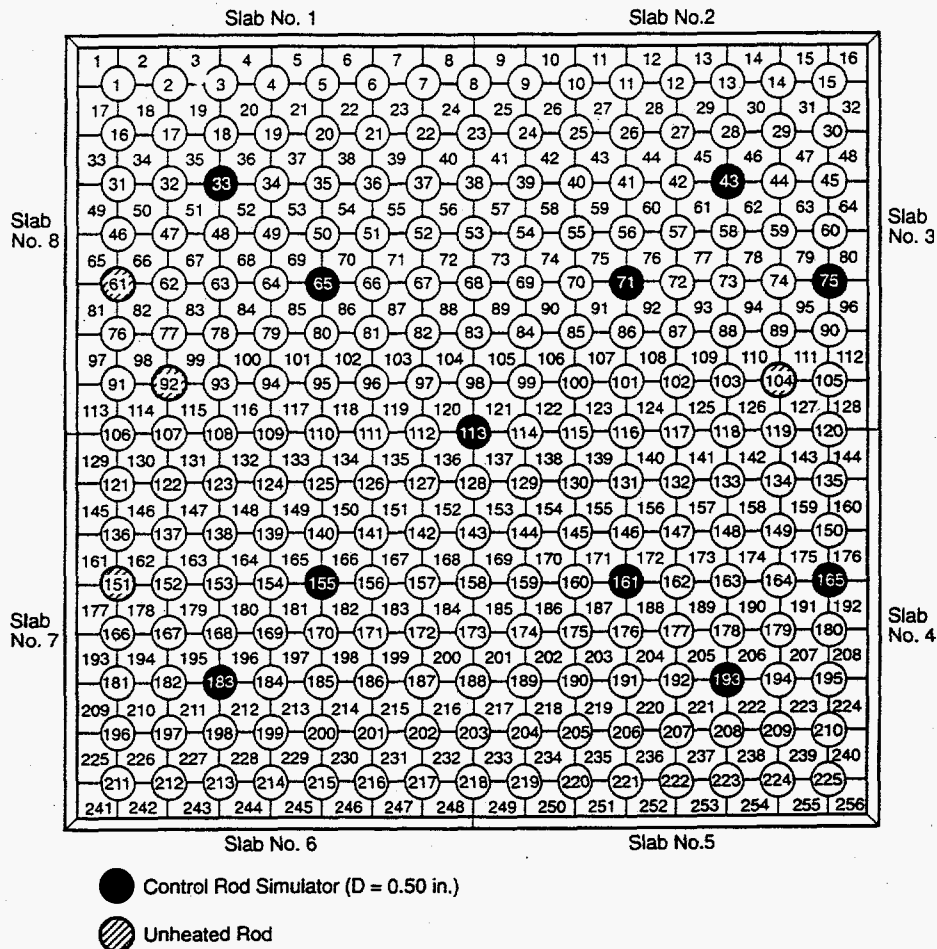


Figure 7.1. Diagram of SAHTT Assembly



S9505005.2

Figure 7.2. COBRA-SFS subchannel model of SAHTT Assembly

see the canister wall is not well validated. For simplicity, it was assumed that the subchannel correlations could be used with the wall channels, but this is an area where it might be profitable to do further work.

In the laminar flow regime, the heat transfer correlation was defined as

$$H_{\text{laminar}} = \frac{k}{D_e} (0.83 \text{Re}^{0.33} \text{Pr}^{0.33})$$

In the turbulent flow regime, the heat transfer correlation is

$$H_{\text{turbulent}} = \frac{k}{D_e} (0.33 \text{Re}^{0.6} \text{Pr}^{0.3})$$



In both correlations,  $k$  is the thermal conductivity of the fluid,  $D_c$  is the subchannel hydraulic diameter, which is also the characteristic dimension for the Reynolds number. As explained in Section 6.2.8, the code calculates the local heat transfer coefficient in a node as the maximum of the laminar and turbulent correlation values.

With a zero flow boundary condition resulting in zero axial flow in the subchannels, the axial friction factor is not an important item in the input, but it is necessary to provide a reasonable value to be used in the solution of the axial momentum equation. The axial friction factor,  $f$ , was specified at a value typical for PWR rods in square array, as

$$f = \frac{100}{Re}$$

The lateral friction factor,  $f'$ , however, is quite important in this case, since this is the direction for natural circulation occurring in the horizontal assembly. The friction factor correlation for laminar flow in the lateral direction is as follows:

$$f' = \frac{55}{Re}$$

For turbulent flow in the lateral direction, the friction factor correlation is given by

$$f' = 0.47 Re^{-0.15}$$

For lateral flow, the characteristic dimension for the Reynolds number is the width of the gap for crossflow between adjacent channels.

The results obtained with the COBRA-SFS code using this model for Test Run No. 12 of the SAHTT series are presented in Figure 7.3. The graph in Figure 7.3(a) is a profile of the peak rod temperatures on a diagonal across the assembly cross-section (refer to Figure 7.2 for the indices of the rods lying on the diagonal). The graph in Figure 7.3(b) shows a profile from top to bottom of the assembly cross-section. In both plots, the COBRA-SFS results show excellent agreement with the measured temperatures. The code predictions are in general slightly above the measured temperatures, as shown in Table 7.1.

The measurement uncertainty is based on the reported total uncertainty of  $\pm 3^\circ\text{C}$  ( $\pm 37^\circ\text{F}$ ) in the rod temperature measurements in the tests. The reported total uncertainty in the boundary temperature measurement is  $\pm 5^\circ\text{C}$  ( $\pm 41^\circ\text{F}$ ). Given these measurement uncertainties, the COBRA-SFS results show that the code is correctly modeling the heat transfer phenomena within the bundle, taking into account both natural convection in the lateral direction, and radiative heat transfer from rod-to-rod and rod-to-wall.

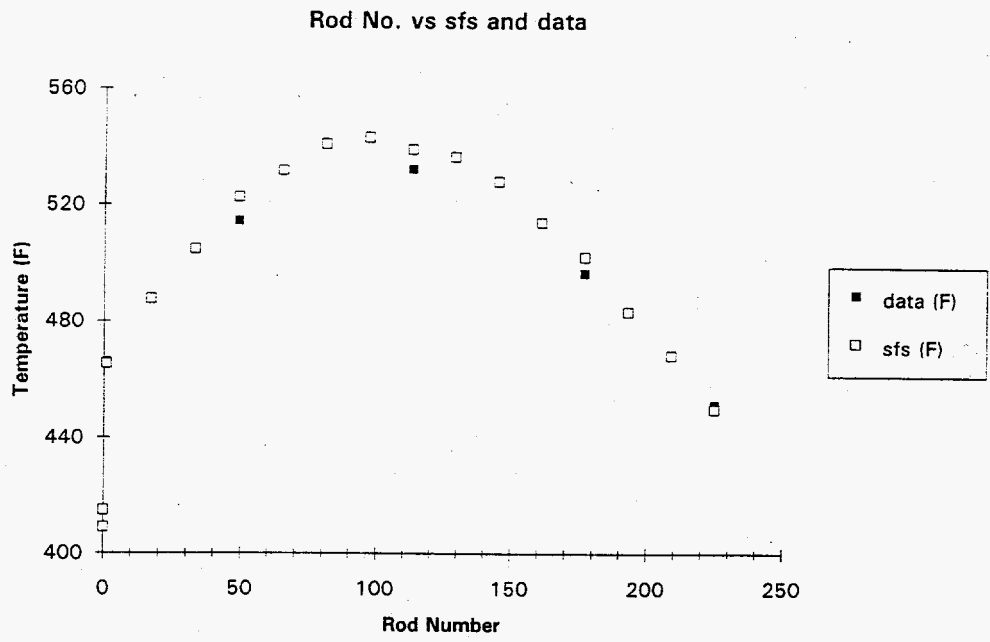


Figure 7.3(a). Comparison of COBRA-SFS Results with SAHTT Data: Rod Temperature Profiles on the Diagonal

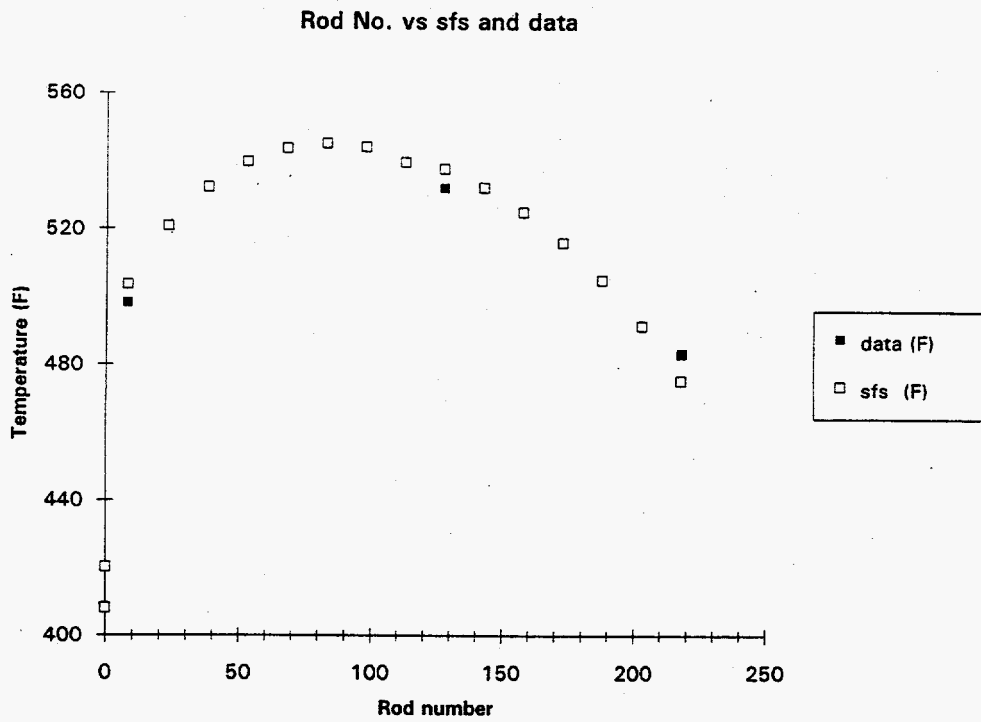


Figure 7-3(b). Comparison of COBRA-SFS Results with SAHTT Data: Rod Temperature Profiles; Top Face to Bottom Face

**Table 7.1.** Comparison of COBRA-SFS Predictions to Measured Temperatures for SAHTT Test No. 12

Rod number	COBRA-SFS predicted temperature (°C)	Measured temperature (°C)	Percent difference	Measurement uncertainty (%)
1	240.7	241.0	-0.12	±1.25
49	272.6	268.0	1.71	±1.12
129	280.4	278.0	0.86	±1.08
177	261.1	258.0	1.20	±1.16
8	261.9	259.0	1.12	±1.16
128	280.9	278.0	1.04	±1.08
218	246.3	251.0	-1.87	±1.20

### 7.2.2 Mitsubishi 15 x 15 PWR Test Assembly

An electrically heated test assembly modeling PWR spent fuel was developed by Mitsubishi Heavy Industries (Iriano et al. 1986). This assembly had essentially the same geometry as the PNL SAHTT assembly described in Section 7.2.1, except that the Mitsubishi cask was only 78.74 in. (2 m) long, rather than 144 in. (3.658 m). Also, all 21 rods in positions corresponding to the locations of control rod thimbles in a 15 x 15 fuel bundle were unheated. The unheated rods were of the same diameter (0.42 in. [1.07 cm]) as the heated rods. A diagram of the assembly geometry model is shown in Figure 7.4.

The Mitsubishi assembly was operated at powers up to about 2.5 kW, with a constant temperature of 200°C (392°F) on all boundaries. The test selected for comparison with COBRA-SFS calculations had nitrogen back-fill and total power input of 1.26 kW.

The Mitsubishi assembly was modeled for the COBRA-SFS calculations by making a few modifications to the input model developed for the SAHTT assembly. The full axial length was modeled in four axial nodes, rather than the center slice used for the SAHTT assembly. In addition, the subchannel geometry was modified slightly to correct for the difference in control rod diameter (0.5 in. [1.270 cm] in the SAHTT assembly and 0.42 in. [1.067 cm] in the Mitsubishi assembly). The total power input was specified as 1.26 kW in the Mitsubishi assembly model.

Even though the SAHTT and Mitsubishi assemblies had essentially the same geometry, the grey body view factors for radiative heat transfer were different, due to different rod diameters and the surface emissivities. The rest of the input, however, including the friction factor and heat transfer correlations, remained the same. The grey body view factors were calculated with RADGEN, using an estimated surface emissivity of 0.2 for both the rods and the walls. A constant temperature of 200°C (392°F) was specified on all boundary surfaces for the Mitsubishi assembly.

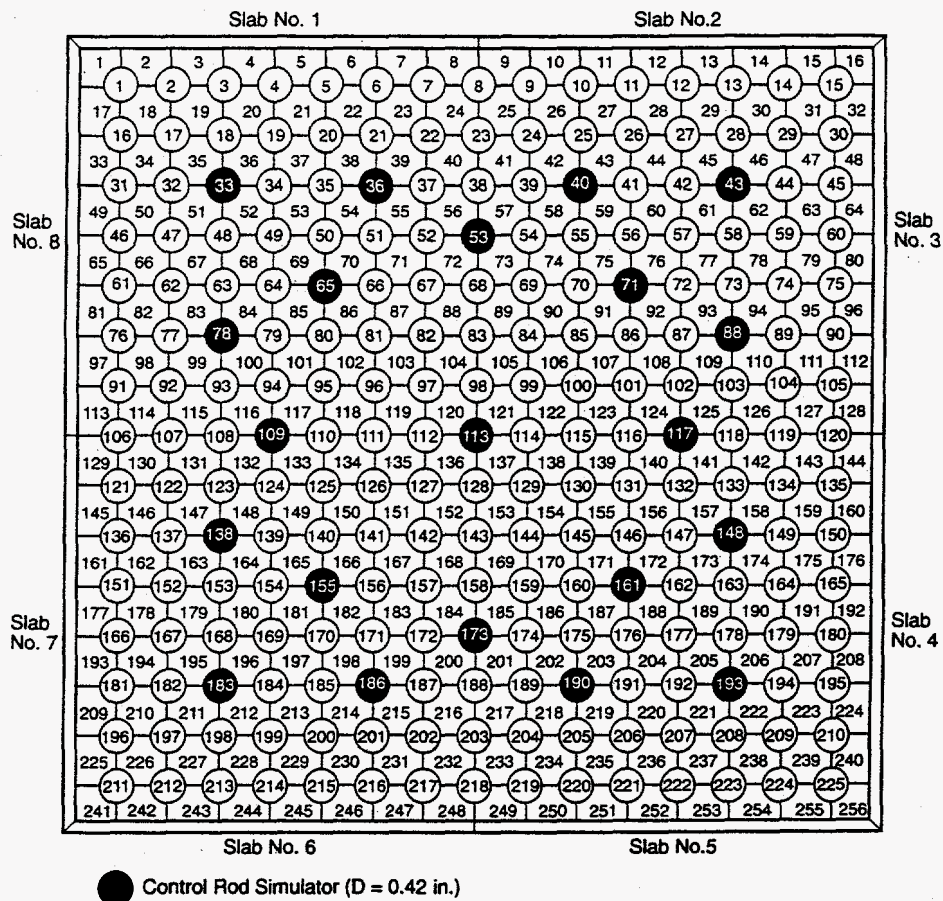
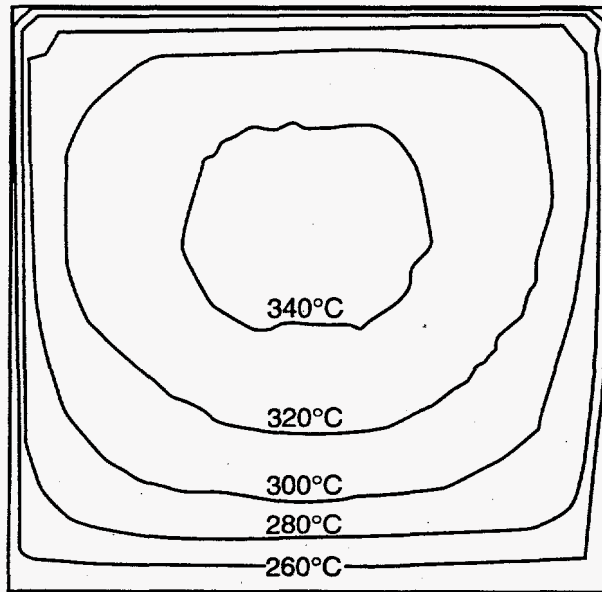


Figure 7.4. COBRA-SFS Subchannel Model of Mitsubishi Test Assembly

The results obtained with COBRA-SFS for the Mitsubishi assembly test case are difficult to assess against the measured rod temperatures, since the published reference (Iriano et al. 1986) does not include this information directly. The paper presents, instead, a comparison with the results of calculations using the computer code SICOH-3D, using a contour map of temperatures. This is reproduced in Figure 7.5(a), along with a similar contour map of the COBRA-SFS results for this case in Figure 7.5(b). The two mappings are very similar, and both show the expected shifting of the peak temperature to slightly above the center of the bundle, due to the effect of natural circulation in the lateral direction.

The closest estimate of the peak rod surface temperature that can be determined from the data presented in the reference is a value somewhere between 346°C (655°F) and 366°C (691°F). The peak rod temperature obtained with Cycle 2 of COBRA-SFS is 365°C, (689°F). The peak rod temperature obtained with Cycle 1 of COBRA-SFS (which cannot take into account the effects of natural convection in the lateral direction), was 390°C (735°F) for this case. The difference of 25°C (77°F) between the two results indicates how important natural convection can be in a problem of this type.

The ability to model natural circulation in the axial direction has been a feature of COBRA-SFS from the beginning. The new version, Cycle 2, adds the capability of modeling natural circulation in the lateral direction as well, for application to analysis of horizontal assemblies and casks.



S9509026.2

Figure 7.5(a). Mitsubishi Results for SICOH-3D Code

**Temperature (F) vs. Location**

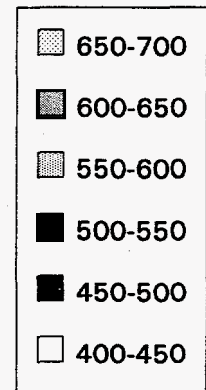
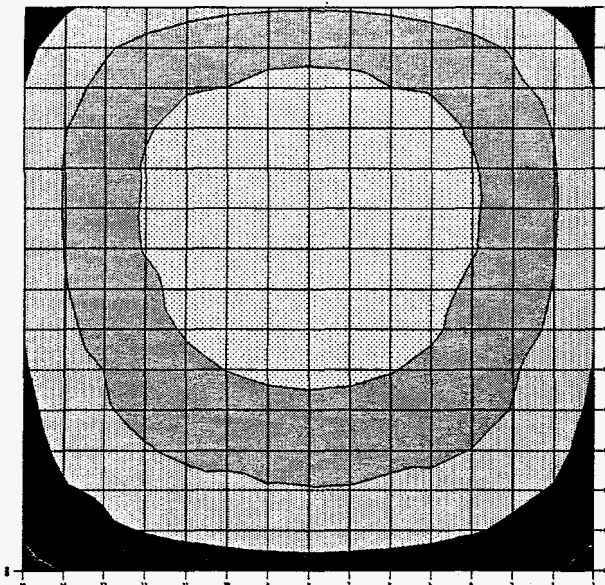


Figure 7.5(b). Mitsubishi Results for COBRA-SFS, Cycle 2

### 7.2.3 BNFL 16 x 16 PWR Test Assembly

The test assembly built by BNFL consisted of a 16 x 16 array of zircaloy-4 tubes bound by nine spacer grids within a square aluminum canister (Fry et al. 1983). The tubes modeled typical PWR fuel, having an outer diameter of 0.42 in. (1.067 cm), and an axial heated length of 144 in. (365.8 cm). Heating elements were inserted inside the tubes in the assembly, except for the 20 tubes in the locations normally occupied by unheated control rod thimbles in a fuel assembly. The assembly was held in place within the aluminum canister by means of guide rails, two per side, between the canister wall and the support grid frames.

The heating elements were designed to give an extremely flat axial power profile representative of spent fuel. The radial power distribution was also uniform across the assembly cross-section. Temperature measurements were obtained at these 224 locations: at eight axial locations on 26 rods, and at four axial locations on the four sides of the canister.

The total hemispherical emissivity of the inside wall of the aluminum canister was measured at the start and end of the test program. The average measured value at the completion of the tests was 0.38, with a range of 0.36 to 0.40. However, results of measurements of the emissivities of the heater rod surfaces were more equivocal. Measured values varied from 0.42 to 0.93, indicating a large range in surface emissivities for the rods, which may also have been varying in time during the test.

The COBRA-SFS model represented the BNFL assembly using subchannel geometry similar to the models of the SAHTT and Mitsubishi assemblies. A diagram of the model is shown in Figure 7.6. The COBRA-SFS input model consisted of 289 subchannels, 256 rods, and 8 slab nodes modeling the walls of the canister. The rods and subchannels were numbered in the same pattern as illustrated for the SAHTT and Mitsubishi assemblies in Figures 7.2 and 7.4, respectively. In the BNFL test assembly, the rods in positions corresponding to control rod locations had zero power input, but were the same diameter as the rods modeling the fuel rods (i.e.,  $d_{rod} = 0.42$  in. [1.067 cm]).

The same friction factor correlations that were used in the SAHTT and Mitsubishi models were also used in the model of the BNFL assembly. The heat transfer correlations were also the same, except that the leading coefficient on the Reynolds number in the correlation for the laminar flow range was 1.39 instead of 0.83. A zero-flow boundary condition was used in the calculation, with constant boundary temperature on the top, bottom and sides of the assembly. An adiabatic boundary was assumed at the inlet and outlet ends of the horizontal assembly. Based on the measured data, the boundary temperature on the top surface (slab nodes 1 and 2; refer to Figure 7.6), was 254°C (489°F). On the bottom surface (slab nodes 5 and 6), the boundary temperature was 244°C (471°F). On the upper half of the vertical sides, (slab nodes 3 and 8), the boundary temperature was 255°C (491°F), and on the lower half of the vertical sides (slab nodes 4 and 7), it was 250°C (482°F).

The grey body view factors for radiative heat transfer were calculated for the assembly using the RADGEN code. Surface emissivities were assumed to be 0.38 for the walls and 0.7 for the rods. The value of 0.38 is a good estimate of the wall emissivity, as obtained by post-test measurement. The value of 0.7 for the rods, however, is simply a rough average of the high and low measured values for the surface emissivities of the rods. The wide range of values obtained in post-test

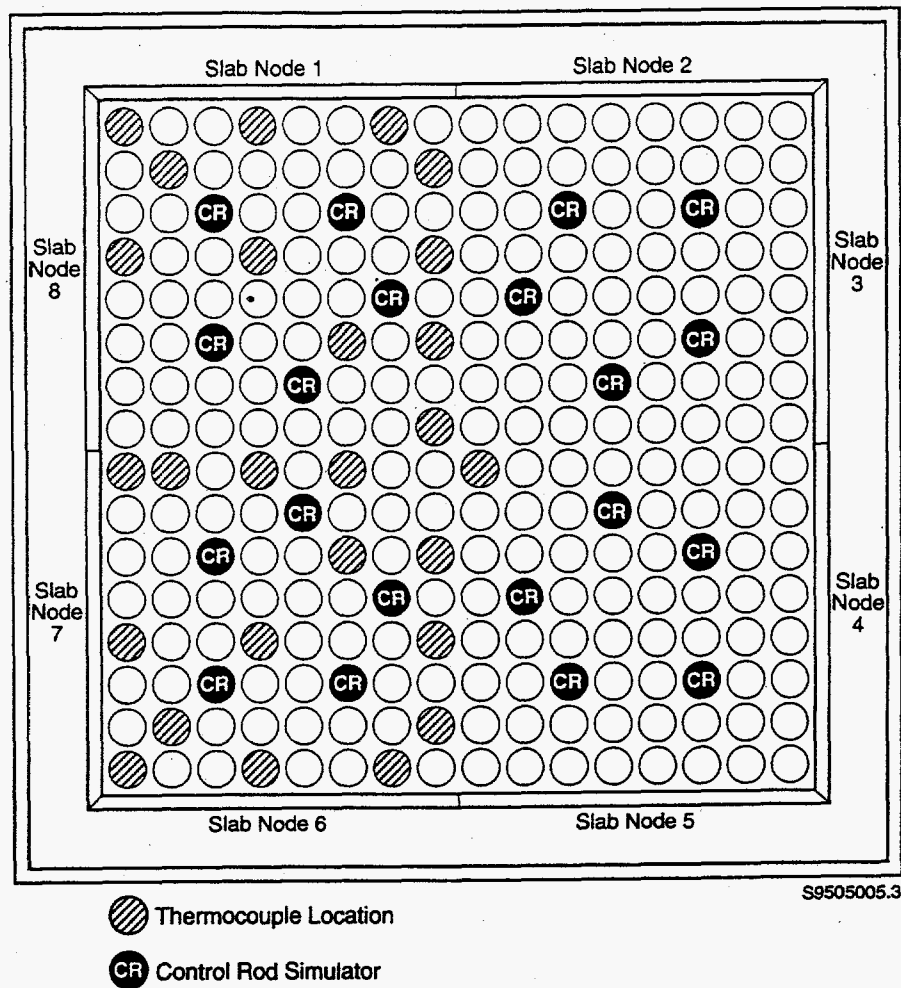


Figure 7.6. Cross-Section of BNFL Test Assembly

measurements indicate that the local emissivities differed substantially from the average. This is perhaps the greatest source of uncertainty in the model for this calculation, since radiative heat transfer was very significant in this test.

Results obtained with the COBRA-SFS model for BNFL test No. 007 are presented in Figure 7.7. In general, the code results predict temperature profiles across the assembly that are in good agreement with the measured data. The shape of the predicted and measured temperature distributions follow the same trends, and reach very nearly the same peak value. The code results are from 9°F (5°C) to 45°F (25°C) hotter than the measured values, but even the largest discrepancy is only about 5% of the total temperature difference between the peak rod temperature and the boundary.

Since radiative heat transfer is the dominant mode for this case, a large part of the difference between the predicted and measured temperatures can be attributed to the unknown variation in local rod surface emissivities. This result underscores the importance of accurate and reliable input data for COBRA-SFS calculations. The predicted temperatures obtained with the code cannot be more accurate than the data on which the calculation is based.

BNFL: air, high power case

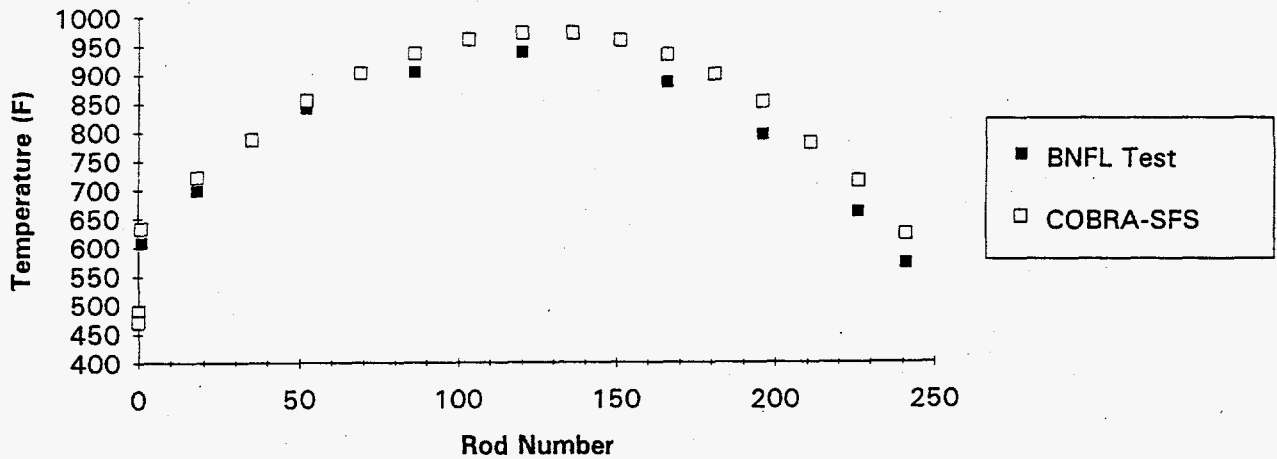


Figure 7.7(a). Comparison of COBRA-SFS, Cycle 2 Results with BNFL Data: Rod Temperature Profile on the Diagonal

BNFL: air, high power case

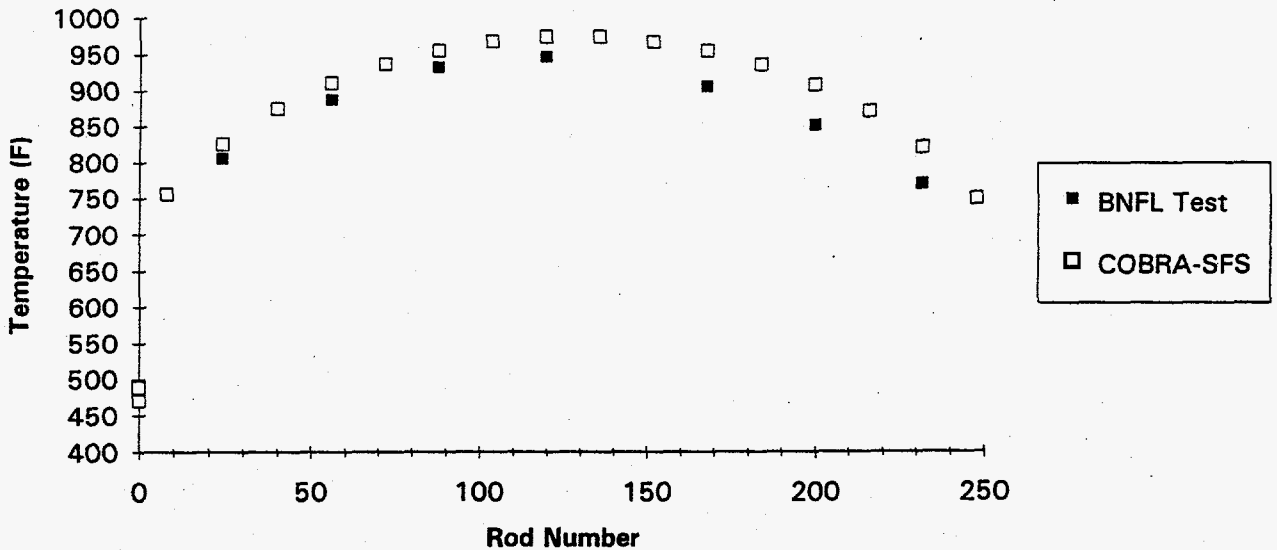


Figure 7.7(b). Comparison of COBRA-SFS Results with BNFL Data: Rod Temperature Profiles; Top Face to Bottom Face

### 7.3 Multiple-Assembly Shipping/Storage Cask Test Cases

The real utility of COBRA-SFS lies in its ability to model large complex spent fuel storage systems using subchannel analysis, so that convective, conductive, and radiative heat transfer modes can be accounted for in appropriate detail. Comparison with single assembly tests is a useful and necessary step in the process of validating the code, but the acid test is the code's ability to predict the temperature field within a multi-assembly shipping or storage cask.



COBRA-SFS calculations have been compared with data obtained in a large number of different storage and shipping cask designs. There are a number of factors to be considered in evaluating such comparisons, chief among them being the level of geometry modeling detail required to obtain adequate resolution of the problem for the flow and heat transfer solution, the definition of appropriate boundary conditions for a given test case, and the applicability of the flow and heat transfer models (e.g., friction factors, heat transfer correlations, and radiation exchange factors) used in the calculation. All of these factors must be considered in evaluating the results in comparison with the measured data.

Experience with a wide range of cask geometries has given some confidence about the level of geometric detail required for good results, and boundary conditions can be gleaned by diligent examination of the measured data. If these two aspects of the model preparation are done properly, that leaves evaluation of the flow and heat transfer models as the primary task. Many of the new capabilities in Cycle 2 of COBRA-SFS were developed specifically for modeling new cask designs. The two cask designs included here for validation of the new version are the TN24P cask with unconsolidated fuel, and the PSN concrete cask with consolidated fuel. These two cases are described in detail in Sections 7.3.1 and 7.3.2, respectively.

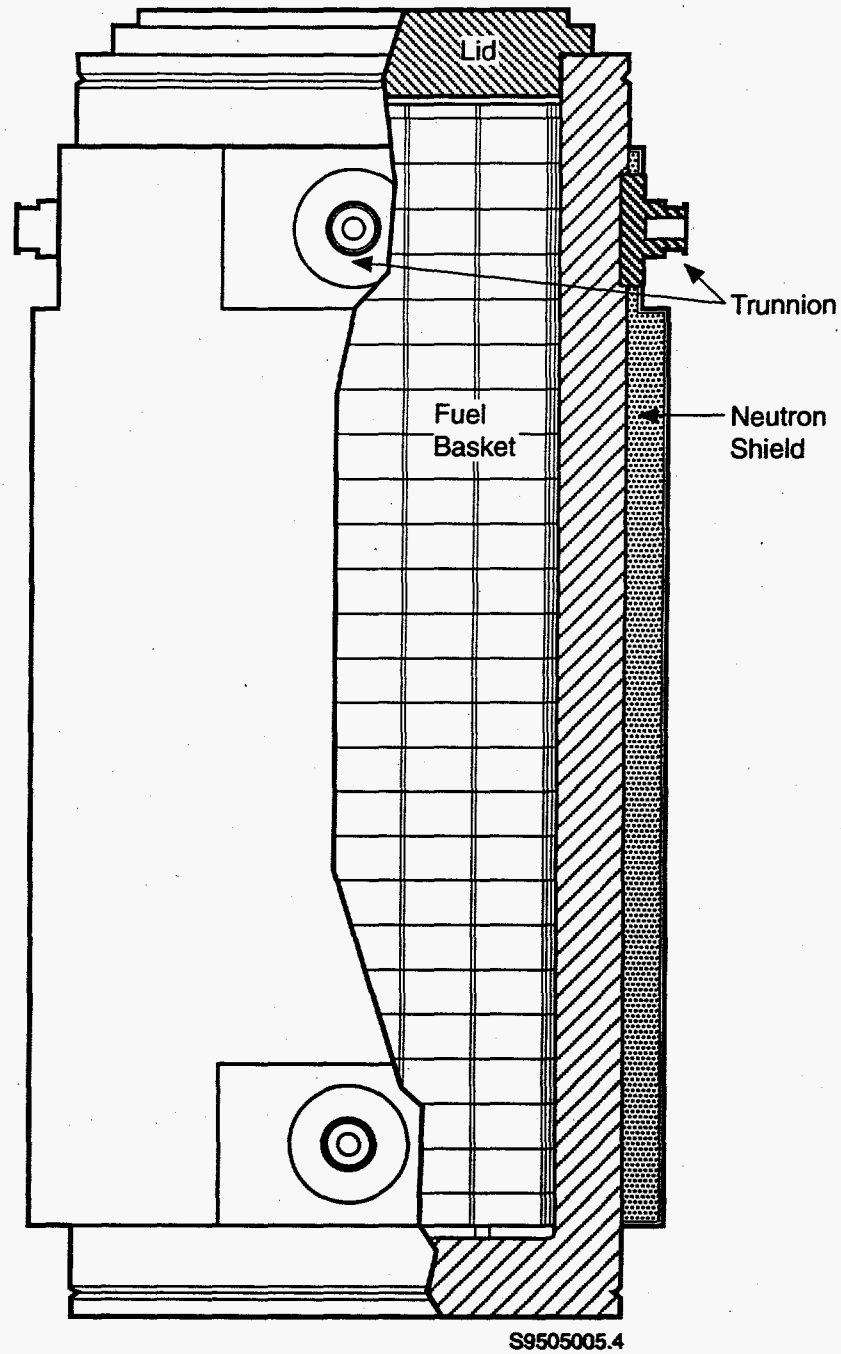
### **7.3.1 TN24P with Unconsolidated Fuel**

The TN24P spent fuel storage cask was constructed by Transnuclear, Inc. under a cooperative program between Virginia Power and the U.S. Department of Energy (DOE). Performance testing was conducted jointly by Virginia Power, PNL (operated for DOE by Battelle Memorial Institute), and by the Idaho National Engineering Laboratory (INEL) (operated for DOE by EG&G, Inc.). The Electric Power Research Institute (EPRI) also participated in the program, through a separate agreement with Virginia Power. The program is fully documented elsewhere (EPRI NP-5128, Transnuclear Inc. E-7455(rev. 0) and E7107, PNL-5777[vol. II]); this section presents only a summary description of the cask and the COBRA-SFS model. Results are presented for two cases: one with the cask in the vertical orientation, and the other with the cask horizontal. In both cases, the cask was back-filled with helium.

#### **7.3.1.1 TN24P Cask Description**

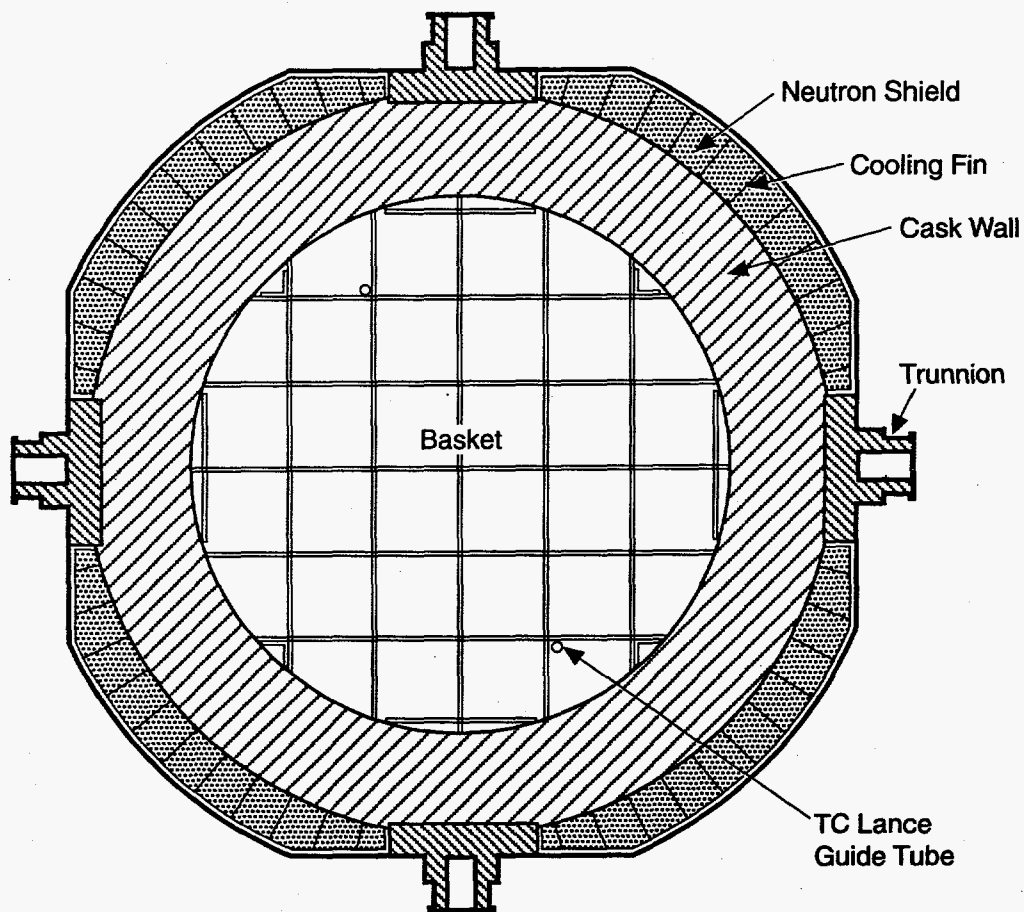
The TN24P spent fuel storage cask was designed to hold up to 24 spent fuel assemblies and dissipate heat loads of up to 24 kW. A cut-away diagram of the cask is shown in Figure 7.8. The cask body is of forged steel surrounded by a resin layer for neutron shielding, with a steel outer shell. The overall cask length is 16 ft (5.0 m), with an outer diameter of 7.5 ft (2.3 m). When loaded with unconsolidated spent fuel, it weighs approximately 100 tons.

Figure 7.9 shows a cross-section through the center of the cask. The spent fuel basket consists of stacked interlocking plates of aluminum and boron. For the "hot" tests, the cask was loaded with spent fuel from the Surry nuclear plant. This fuel was standard Westinghouse 15 x 15 PWR fuel, with nominal rod diameter of 0.420 in. (1.067 cm) in a square array with 0.563-in. (1.430-cm) pitch. Nominal fuel column length was 144 in. (365.8 cm), and the fuel burnup was approximately 30 GWd/MTU for each assembly. Decay heat rates in the fuel rods for the duration of the tests were calculated using ORIGEN2 (Croff 1980). The predicted power at the start of testing was 20.6 kW, and at the end of testing was 20.3 kW. The average power per assembly was approximately 850 kW.



**Figure 7.8.** TN24P PWR Spent Fuel Storage Cask

Six tests were run to assess the performance of the TN24P cask: three with the cask oriented vertically and three with the cask horizontal. In each orientation, tests were run with these three different back-fills: nitrogen, helium, and vacuum. A summary of the test matrix is given in Table 7.2.



S9505005.5

**Figure 7.9.** TN24P PWR Spent Fuel Storage Cask Cross-Section

The system was massively instrumented with thermocouples to obtain performance and boundary condition data during operation. These included 14 thermocouples attached to the inner wall and basket of the cask, 54 thermocouples in the fuel assemblies and basket, and 35 thermocouples on the cask exterior.

The thermocouple locations were selected for measurement redundancy, to provide measures of expected temperature symmetries within the cask, and to define axial, radial and temperature profiles

Table 7.2. TN24P Cask Test Matrix and Peak Temperatures

Run number	Orientation	Back-fill	Cask heat load (kW)	Ambient temperature	Estimated peak clad temperature
1	vertical	helium	20.6	18°C (64°F)	221°C (430°F)
2	vertical	nitrogen	20.6	20°C (68°F)	241°C (466°F)
3	vertical	vacuum	20.6	20°C (68°F)	290°C (554°F)
4	horizontal	helium	20.5	18°C (64°F)	215°C (419°F)
5	horizontal	nitrogen	20.4	21°C (70°F)	256°C (493°F)
6	horizontal	vacuum	20.3	19°C (66°F)	280°C (536°F)

for both the vertical and horizontal orientations. The peak clad temperature for each test was estimated using a cubic fit to the data from the thermocouples in the fuel assembly<sup>(a)</sup> and assuming a parabolic temperature distribution across the assembly.

COBRA-SFS calculations were performed for all six tests, but only two have been selected to present as part of the validation of Cycle 2. These are runs Nos. 1 and 4; the vertical and horizontal cases with helium back-fill. The COBRA-SFS models for these calculations are presented in the following subsection.

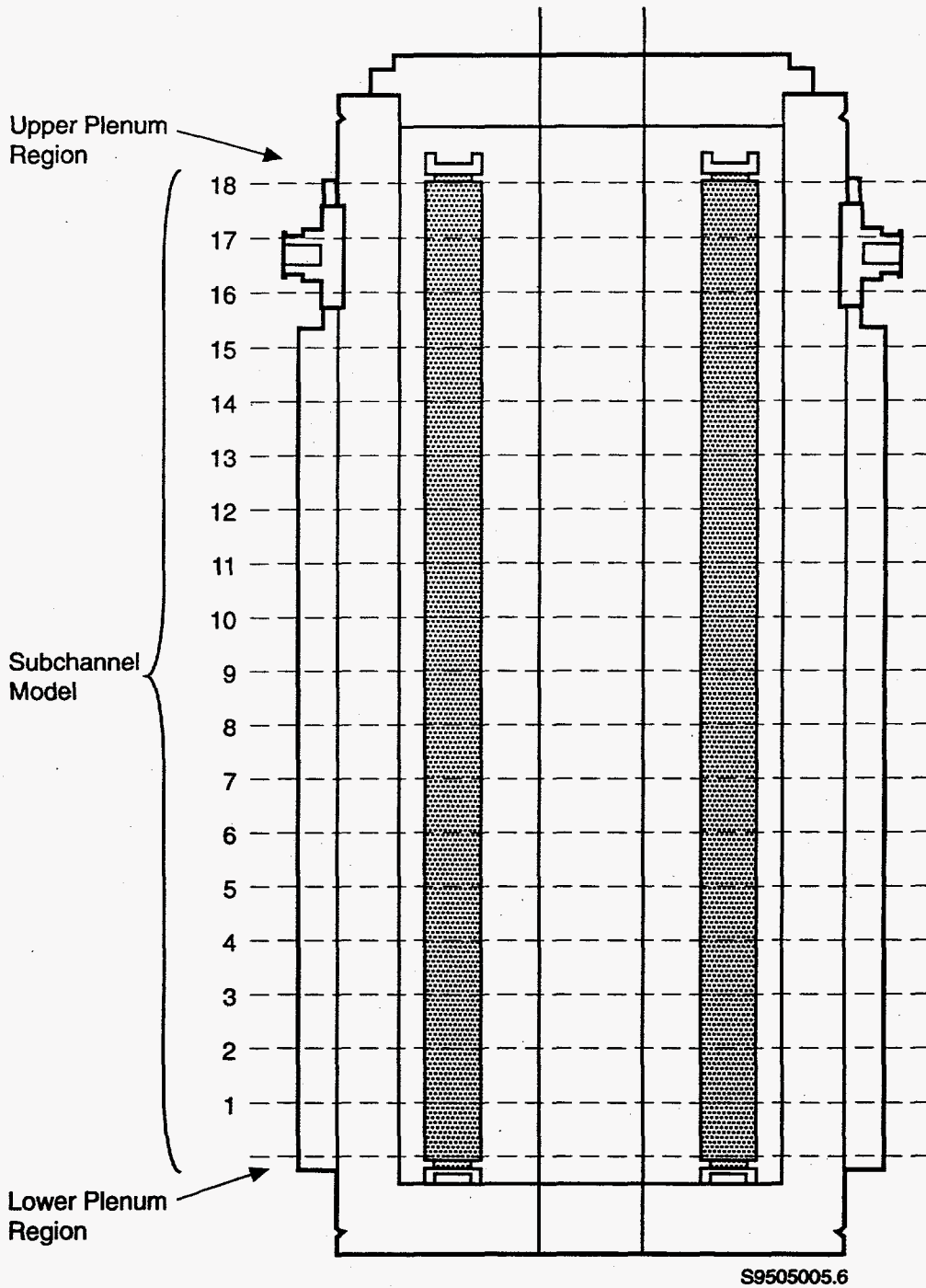
### 7.3.1.2 COBRA-SFS Model of the TN24P Cask

Two geometry models were needed to model the TN24P cask for COBRA-SFS. For the vertical tests, the cask could be represented with a one-eighth section of symmetry. With the cask in the horizontal position, however, the weight of the fuel assembly basket caused it to shift until it was in contact with the cask wall on the lower side. As a result, the only remaining plane of symmetry was the vertical one through the cask center. It was therefore necessary to model at least one half of the cask for the horizontal tests.

In both cases, however, the axial geometry modeling was the same, and consisted of 18 uniform axial nodes, as shown in Figure 7.10. Flow and heat transfer connections to the cask at the top and bottom were modeled using the optional plenum model in the code, to provide connections for natural recirculation within the cask and conductive heat transfer through the cask ends (refer to Section 6.2.10 input group BDRY).

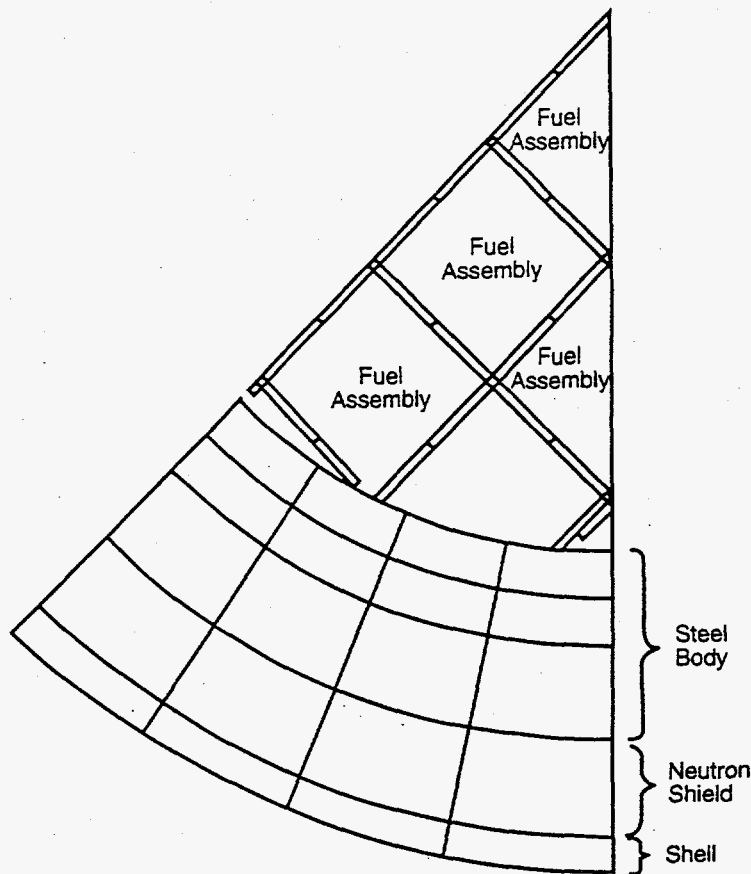
Figure 7.11 shows a cross-section of the model of a one-eighth section of symmetry of the cask. For simplicity, this diagram does not include the subchannel modeling of the individual fuel assemblies within the basket. The cask is represented with 51 slab nodes: 27 modeling the spent fuel

(a) Thermocouples in the fuel assemblies were inserted in the guide tube thimbles. There were no thermocouples on the fuel rods, so a direct measure of peak clad temperature was not obtained.



**Figure 7.10.** Axial Noding for COBRA-SFS Model of TN24P Cask

basket, 12 for the cask body, 4 for the neutron shield, and 8 for the cask outer shell. The outermost nodes of the cask shell are zero-thickness boundary nodes to represent the cask surface temperature, and are used to calculate the appropriate contribution of radiative and convective heat transfer to the environment.



S9505005.7

Figure 7.11. COBRA-SFS Model of One-Eighth Section of Symmetry of TN24P Cask

Within the spent fuel basket, the fuel assemblies were modeled using subchannel analysis. In the one-eighth section of symmetry, there are two types of fuel assemblies; half-symmetry assemblies and whole assemblies. The half-symmetry assemblies were modeled with a subchannel array, as illustrated in Figure 7.12. In the assemblies of this type, the model includes 136 subchannels and 120 rods. Figure 7.13 shows the lumped rod-and-channel arrangement used to model the whole assemblies. Modeling studies (Rector et al. 1986a) have shown that this type of modeling gives adequate resolution of temperature gradients within an unconsolidated spent fuel assembly. This model requires only 57 channels and 105 rods to represent the assembly.

A cross-section of the model for a one-half section of symmetry of the cask is shown in Figure 7.14. In this model, the geometry takes into account both the shifting of the spent fuel basket to contact the inner cask surface on the bottom, and the shifting of the fuel assemblies within the basket. The cask is represented with these 191 slab nodes: 95 for the spent fuel basket, 48 for the cask body, 16 for the neutron shield, and 32 for the cask outer shell. In this model, all of the fuel assemblies were whole assemblies, and were represented with the same lumped channel model as was used for the whole assemblies in the one-eighth section-of-symmetry model (refer to Figure 7.13). In the model for the vertical orientation, it was assumed that the fuel assemblies were centered within their respective basket cells. In this half-symmetry case, however, the flow areas of the channels between the rod array and the assembly canister walls were adjusted to account for the shifting of the fuel assembly within the basket due to the horizontal orientation.

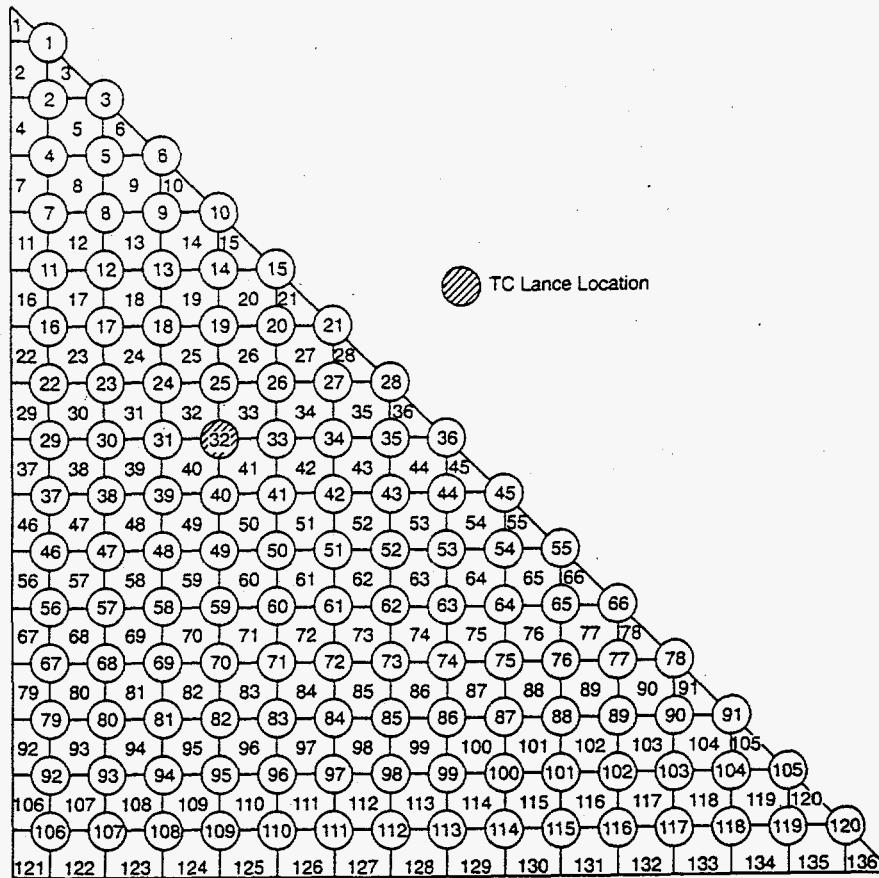


Figure 7.12. Subchannel Model of One-Half Section of Fuel Assembly

COBRA-SFS predictions of the flow and temperature field within the TN24P cask require solving for conductive, convective, and radiative heat transfer. Conduction heat transfer within and between the slab nodes was modeled using thermal resistances based on the material properties of the walls and internals of the cask. The flow field within the cask was determined by imposing boundary conditions of uniform pressure drop in all channels, and zero net flow for the system. The friction factor for the momentum solution was specified at a value typical for square array rods, as

$$f = \frac{100}{Re}$$

Convective heat transfer within the cask was modeled with a heat transfer coefficient determined for a Nusselt number of 3.66. This value comes from an analytical solution (Kays and Crawford 1980) of the energy equation for fully developed temperature and velocity profiles in a circular tube. Previous work (Wiles et al. 1986) has shown that this model gives the best overall results for nitrogen and helium back-fill at temperatures typical of spent fuel assemblies.

Radiative heat transfer within the cask was modeled using grey body exchange factors. The RADGEN code was used to calculate these view factors for the enclosures containing spent fuel rods. The option in RADGEN to sum view factors of lumped rods was used for the enclosures containing

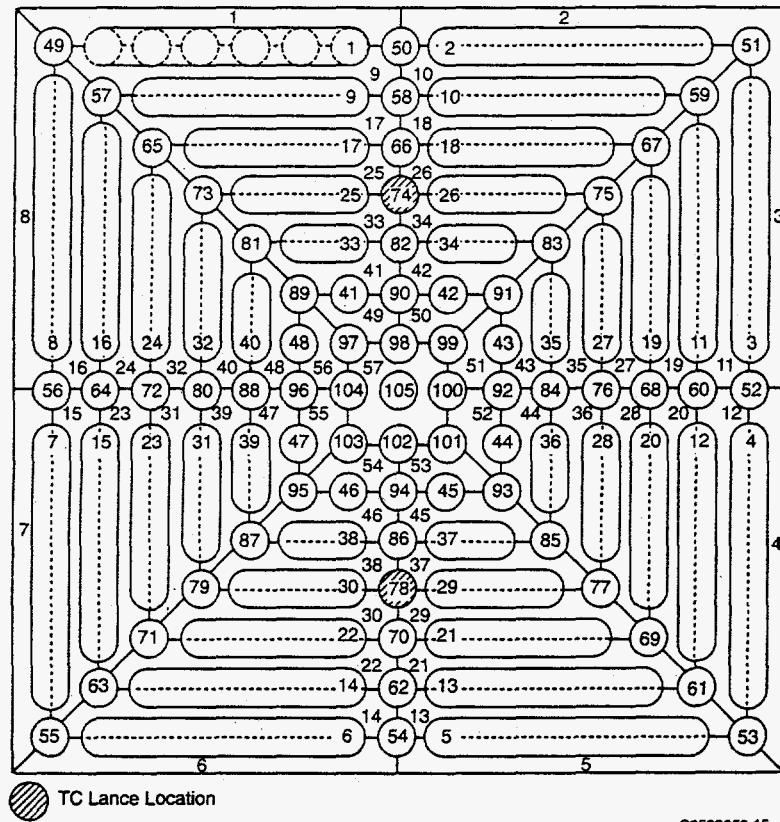


Figure 7.13. Lumped Channel and Rod Model of Fuel Assembly

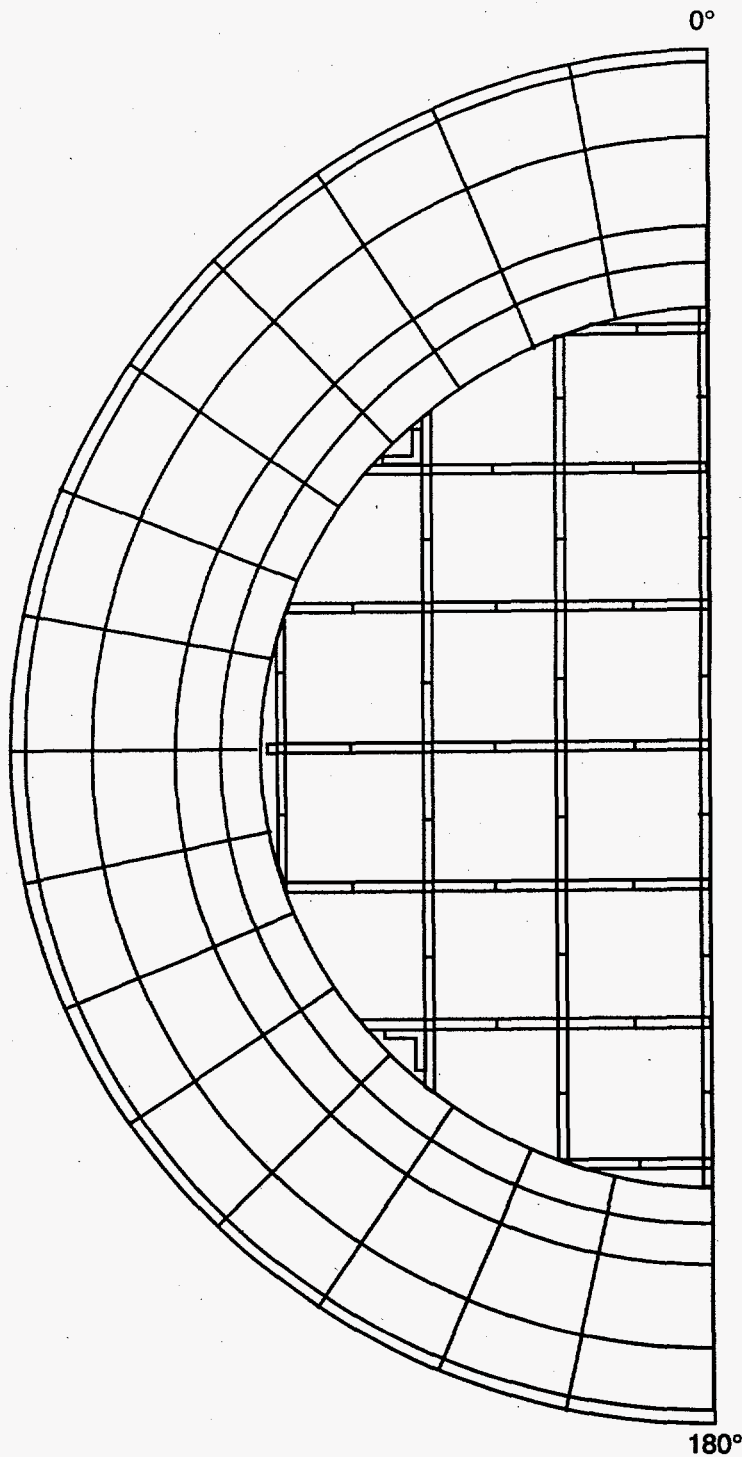
whole assemblies modeled as lumped channels and rods (see Section 6.3). For enclosures within the cask that did not contain rods, black body view factors calculated using the Hottel (Hottel and Sarofin 1967) crossed-string correlation method were specified by input along with the surface emissivities for the slab nodes comprising the open assembly walls. The grey body view factors were calculated automatically in COBRA-SFS.

Thermal boundary conditions on the exterior of the cask were specified by means of heat transfer coefficients on the sides, top and bottom. The outside of the cask was a smooth painted surface that was free to exchange heat with the environment by natural convection and radiative heat transfer. Separate heat transfer coefficients were defined for each region. Radiative heat transfer was calculated at all boundary surface nodes using the formula

$$q''_{\text{rad}} = \frac{1}{\frac{1}{\epsilon_1} + \frac{1}{\epsilon_2} - 1} \sigma(T_1^4 - T_2^4)$$

This is identical to the expression for radiative heat transfer between two parallel plates. The ambient air was assumed to be a black body at 18.25°C (65°F).





S9505005.8

Figure 7.14. COBRA-SFS Model of One-Half Section of Symmetry of TN24P Cask

The cask was placed upright on the bed of a rail car for the vertical tests, and natural convection with the ambient air occurred on the top and sides of the cask. The heat transfer coefficient for the side surface was specified using a Nusselt number for a natural convection from a vertical cylinder in air at 1 atm (Lindeburge 1981), as

$$Nu = 0.13(GrPr)^{1/3}$$

There was no natural convection directly on the bottom surface, but the rail car bed acted something like a cooling fin that exchanged heat with the ambient air by natural convection. The heat transfer coefficient was therefore formulated using the Nusselt number for free convection from a fin.

Heat transfer from the cask lid to ambient was modeled using an expression for the natural convection Nusselt number for a horizontal plate in air at 1 atm (Lindeburge 1981), as

$$Nu = 0.20(GrPr)^{1/3}$$

With the cask in the horizontal position, the Nusselt number for a vertical cylinder in air was used to approximate the heat transfer coefficient on both the ends and the sides.

### 7.3.1.3 COBRA-SFS Results with TN24P Cask Model

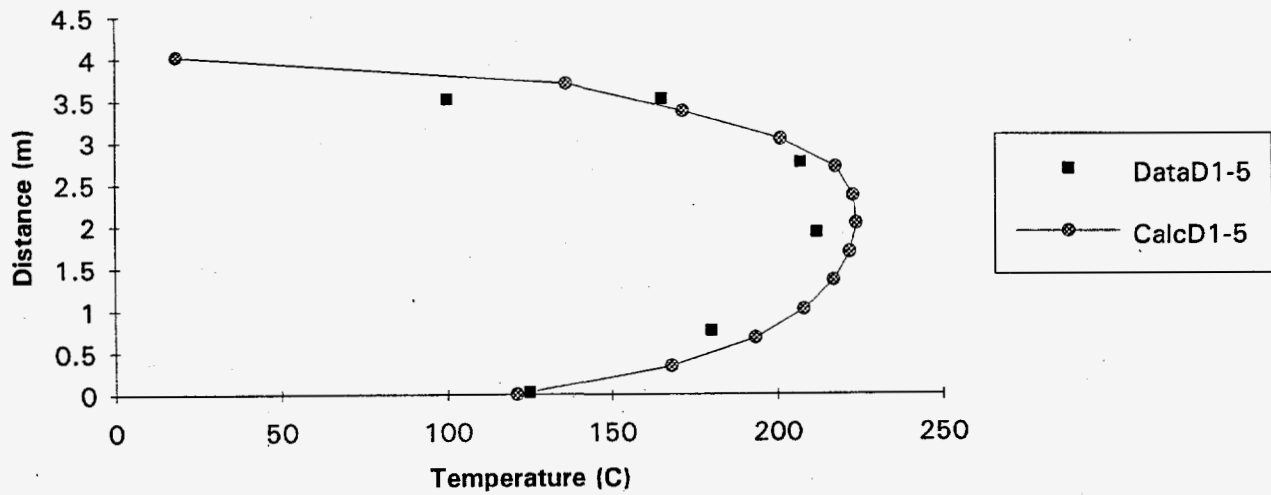
The results obtained with COBRA-SFS for the six tests in the TN24P cask are fully documented elsewhere (EPRI-NP-5128/PNL-6054). Sample results are presented here, to illustrate the performance of Cycle 2 in this analysis. Table 7.3 shows the peak clad temperatures predicted for the two tests selected for Cycle 2 validation. Note that the input files used for this testing were the pre-test models, so the Cycle 2 results compare most favorably with the reported pre-test predictions.

Sample data comparisons are shown in Figure 7.15 for the vertically oriented TN24P cask with helium back-fill (Run No. 1 in Table 7.3). The data in the graph of Figure 7.15(a) is from the thermocouple lance (DataD1-6) in the center assembly of the one-eighth section of symmetry. The calculated temperatures plotted on this graph are for rod No. 28 of the center assembly in the COBRA-SFS model. The data in Figure 7.15(b) are from the thermocouple lance (Data5-6) in the outer assembly

Table 7.3. Peak Temperature Predictions for TN24P Cask Tests

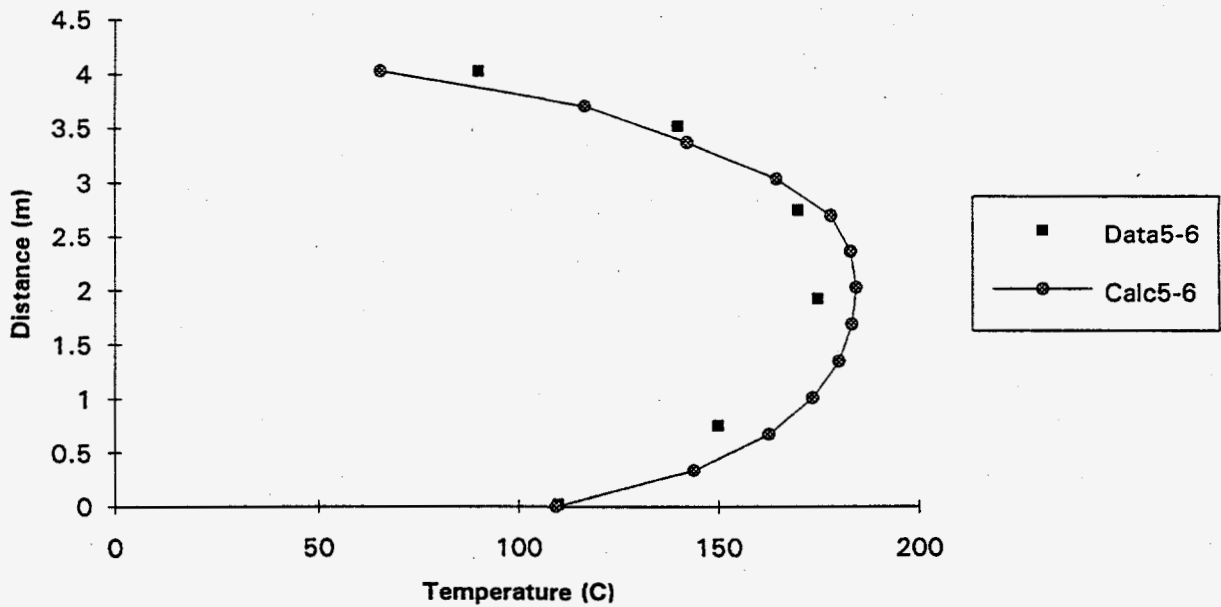
Run number	Orientation/ Backfill	COBRA-SFS predictions			Estimated peak clad temperature
		Peak clad temperature			
		Pre-Test	Post-Test	Cycle 2	
1	vertical, He	225°C (437°F)	220°C (428°F)	224°C (435°F)	211°C (411°F)
4	horizontal, He	218°C (424°F)	222°C (431°F)	220°C (428°F)	206°C (403°F)

### D1-5 Data vs COBRA-SFS Calculations



**Figure 7.15(a).** Comparison of COBRA-SFS Calculations with Measured Temperatures in the Center Assembly of the TN24P Cask (vertical orientation, helium back-fill)

### D5-6 Data vs COBRA-SFS Calculations



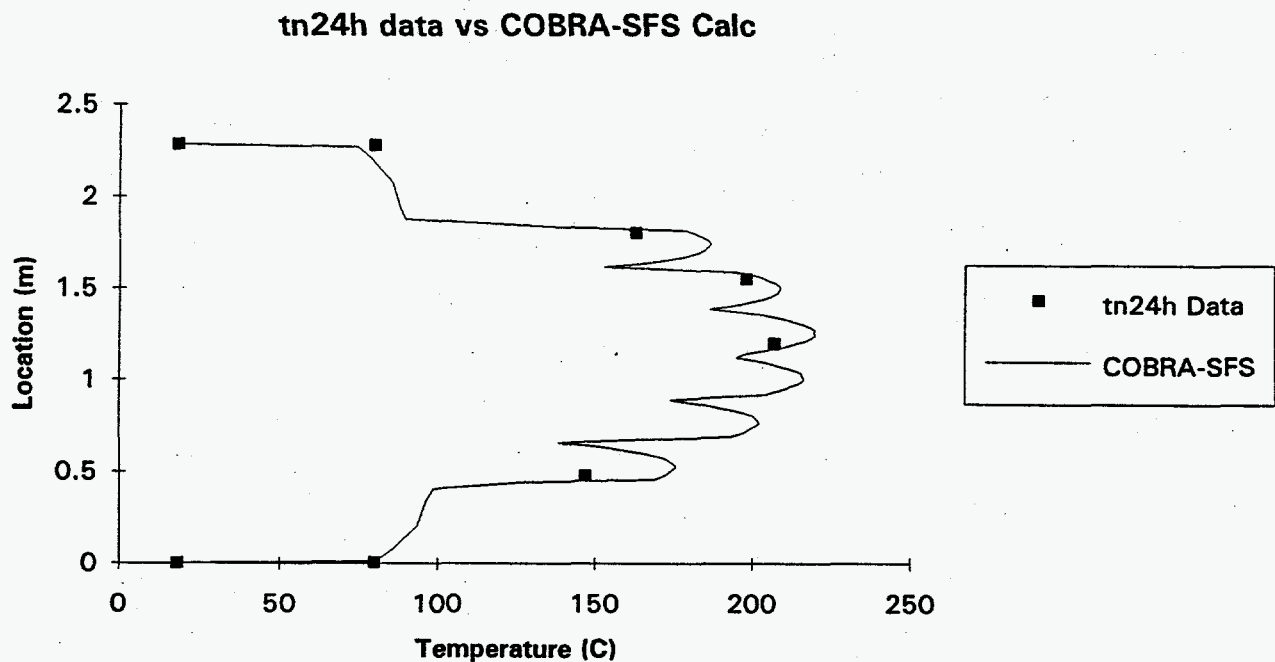
**Figure 7.15(b).** Comparison of COBRA-SFS Calculations with Measured Temperatures in the Outer Assembly of the TN24P Cask (vertical orientation, helium back-fill)

with one corner of the basket touching the inner surface of the cask shell. The calculated temperatures for the graph in Figure 7.15(b) are for rod No. 88 of the outer assembly.

In both comparisons, the COBRA-SFS calculations predict a slightly higher peak temperature than the data shows. However, these results are well within the uncertainty in the measurements, and the code results show the appropriate trends for the overall heat transfer solution.

Figure 7.16 shows a plot of the radial peak temperature profile through the centerline of the cross-section of the horizontal cask. The calculated temperatures are from the lower edge of the cask outer wall (location 0.0 on the graph) to the temperature at the upper edge of the cask outer wall (location 2.2835 m on the graph, corresponding to the cask outside diameter). The solid line on the graph represents the slab node temperatures through the thickness of the cask wall at the top and bottom, and the rod temperatures from the center row of rods in assemblies which lie on the vertical centerline of the cask cross-section (Figure 7.14).

The temperature data on the plot represent the thermocouple lance temperatures measured in the rod bundles corresponding to assembly Nos. 22, 1, 2, and 4 in the COBRA-SFS model. Note that these measured temperatures are not peak temperatures. The thermocouple lances were located within a guide tube thimble four rod rows from the bundle centerline. The agreement between the measured temperatures and the calculated values shown in Figure 7.16 is very good.



**Figure 7.16.** COBRA-SFS Calculation of Top-to-Bottom Peak Temperature Profile Through the Centerline of the Cross-Section of the Horizontal TN24P Cask, Compared with Measured Temperatures

### 7.3.2 PSN Concrete Cask

The PSN spent fuel storage cask was constructed by Pacific Sierra Nuclear Associates<sup>(a)</sup> for long-term storage of consolidated spent fuel from pressurized water reactors. Performance testing was conducted for DOE and PSN by PNL, and INEL. EPRI also participated in the program, as part of a cooperative program involving EPRI, DOE, PSN, and Wisconsin Electric Power Company, to evaluate horizontal modular storage systems for spent fuel.

Testing was conducted at INEL's Test Area North (TAN) cask testing facility. The pretest preparations, performance testing, and post-test activities are fully documented elsewhere (EPRI TR-100305). Only summary descriptions of the cask and COBRA-SFS model are presented here, along with results obtained with the test cases selected for validation of Cycle 2. These are the two cases with vents fully open, one with helium back-fill, and the other with nitrogen back-fill.

#### 7.3.2.1 PSN Concrete Cask Description

The VSC-17 spent fuel storage system designed by PSN is a passive container for storing up to 17 assemblies or canisters of irradiated nuclear fuel. The system consists of a Ventilated Concrete Cask (VCC), and a Multi-Assembly Sealed Basket (MSB). Diagrams of the cask structure are shown in Figure 7.17, with vertical and horizontal cross-sections. The concrete shell is a one-piece cylinder 20-in. (50-cm) thick, with a steel inner liner. The shell provides structural support, radiation shielding, and an annular path for natural convection cooling of the basket assembly. The cask weighs approximately 80 tons empty, and 110 tons when fully loaded with consolidated fuel.

The concrete of the cask bottom is an integral part of the cask shell, as shown in Figure 7.17, not a separate piece fastened to the concrete annulus forming the body of the cask. The cask bottom consists of a concrete slab 22-in. (56-cm) thick, with a steel plate on the inner surface that forms the bottom of the internal cavity. The cask lid consists of a steel weather cover capping the concrete annulus. The steel plate is bolted to the top of the steel liner over a sheet rubber seal. The lower side of the cask lid is formed by the lid of the basket assembly, which consists of two steel plates with neutron shielding material sandwiched between them.

The basket assembly is of pressure vessel grade steel, and consists of an outer shell, the shield lid, and the fuel guide sleeve assembly, as shown in Figure 7.18. There is sufficient clearance between the outer shell of the basket and the storage sleeve assembly to ensure that differential thermal expansion during operation will not result in load transfer between the two structures. Interior structural support for the basket is provided by three steel shells wrapped around the framework of the fuel guide sleeves at the top, bottom, and middle of the assembly, as shown in Figure 7.18. These basket support structures are welded to the outer edges of the framework of the fuel guide sleeves and to the inner surface of the basket outer shell.

---

(a) Pacific Sierra Nuclear Associates became Sierra Nuclear Corporation; in some of the documentation of the cask design and testing, SNC is used interchangeably with PSN in reference to the cask manufacturer.

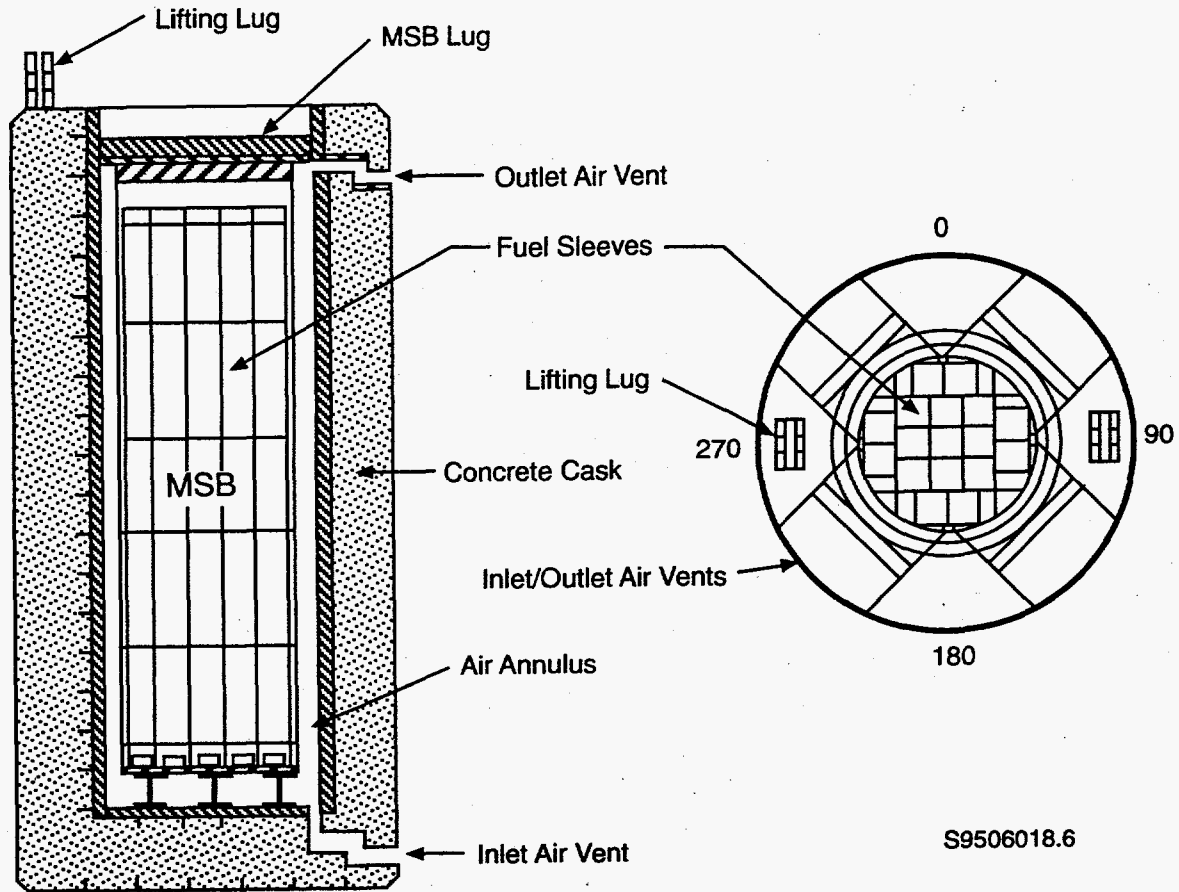
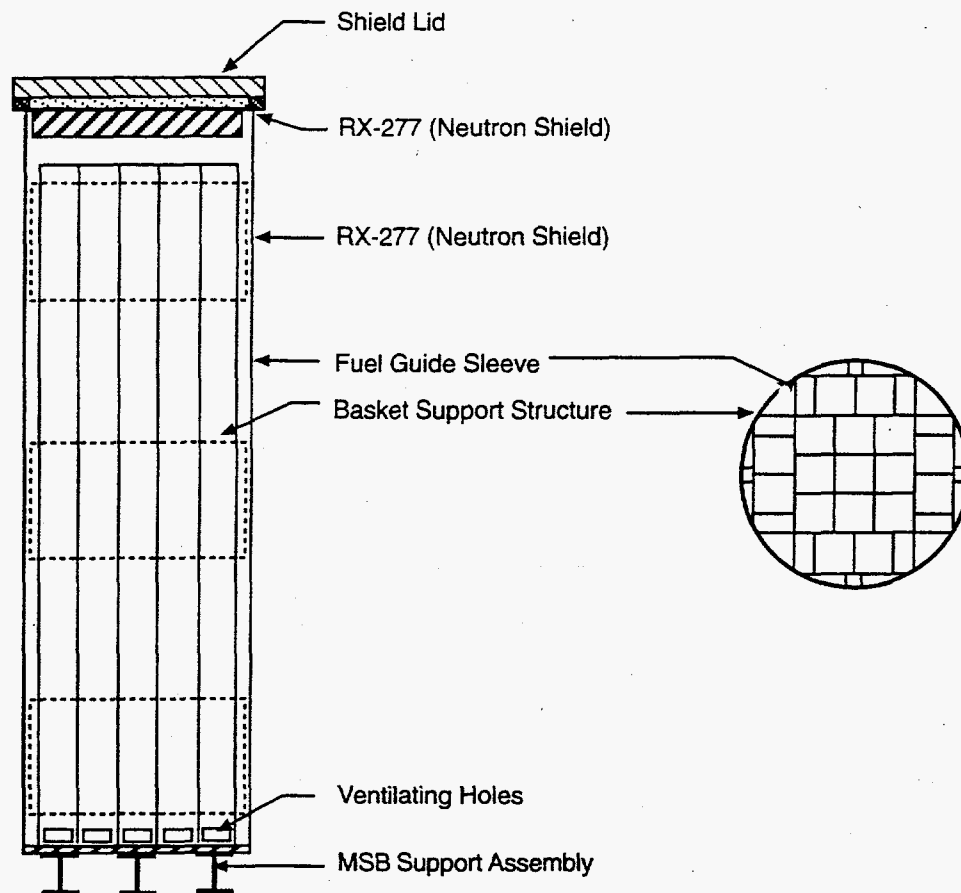


Figure 7.17. Diagram of PSN/VSC-17 Cask Structure

The basket lid is sealed with two o-rings, one of steel and one of elastomer, to provide a gas-tight seal to hold a back-fill of helium, nitrogen, or vacuum. The depth of the cask cavity is sufficient to accommodate BWR fuel; therefore, an MSB support assembly is required when the cask is loaded with PWR fuel, as illustrated in Figures 7.17 and 7.18. This support assembly allows free circulation of air from the vented annulus along the underside of the basket assembly within the cask cavity.

Decay heat generated in the spent fuel is transmitted through the basket wall to the air in the annulus between the basket and the concrete shell. Natural circulation brings cooling air in through the bottom vents and exhausts it to the environment through the top vents. The inlet and outlet vents are steel-lined penetrations arranged in a cruciform pattern, with the flow path constrained by a stepped structure to minimize radiation streaming, as illustrated in the cross-sectional diagrams in Figure 7.17.

The cask was instrumented with pressure transducers in the cask cavity, and a total of 98 thermocouples to measure thermal performance. Ten thermocouples were attached to the outer surface of the cask, five to the MSB lid, and two to the weather cover. Ten were embedded in the concrete,



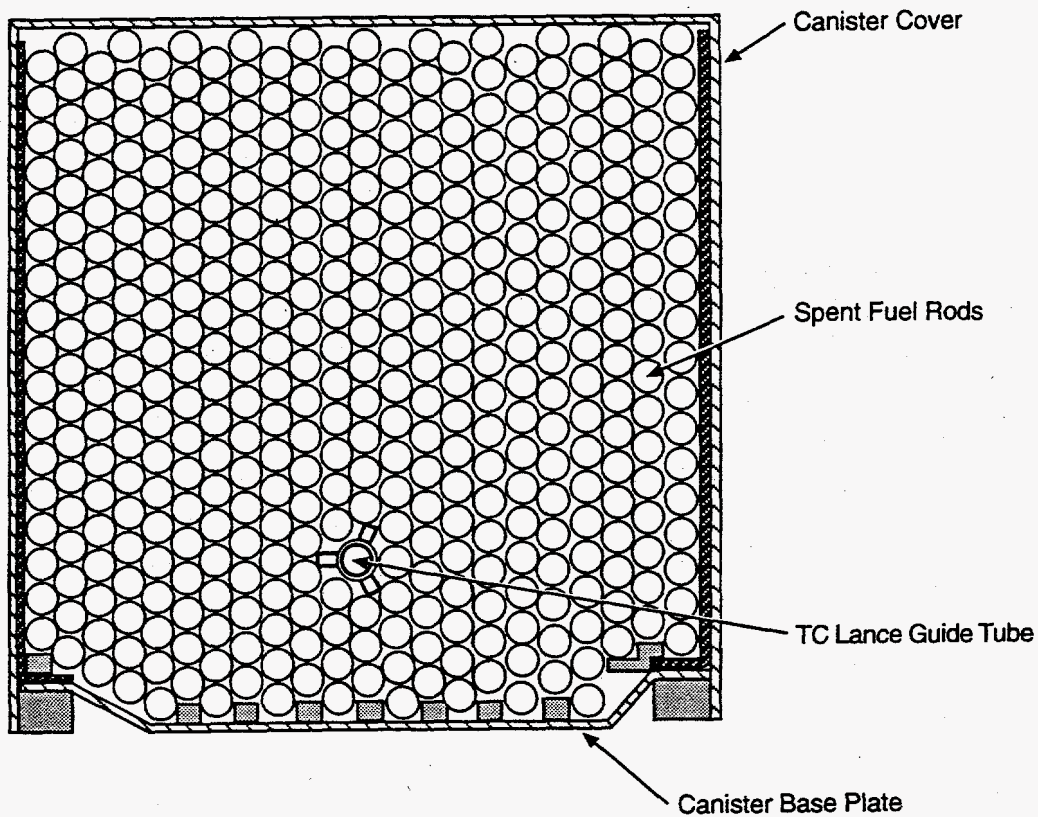
S9506018.5

Figure 7.18. PSN/VSC-17 Cask Multi-Assembly Sealed Basket

nine were attached to the outside surface of the basket shell, nine to the steel inner liner of the concrete shell, and one was installed in the center of each air inlet and outlet vent. The basket internals were instrumented with seven thermocouple lances containing a total of 42 thermocouples, which were inserted into selected guide tubes within six of the fuel canisters. The axial and radial locations of the thermocouple lances were selected to allow evaluation of temperature symmetry and axial and radial temperature profiles in the cask.

The cask was loaded with Westinghouse 15 x 15 PWR spent fuel from the Surry 2 and Turkey Point plants. The fuel had been removed from the normal 15 x 15 rod array and consolidated with the rods from two fuel assemblies in each canister. The canisters were of stainless steel, and consisted of a base and top locking cover, as shown in Figure 7.19. Each canister was loaded with 408 fuel rods, stacked on a triangular pitch. The top and bottom plates of the canister did not provide a gas-tight seal, but nevertheless severely limited the gas flow rate into or out of the canister.

Decay heat rates for the fuel were calculated from measured properties of the fuel using the ORIGEN2 computer code. Decay heat from the individual canisters ranged from 700 to 1050 watts, with an average of about 877 W per canister. The canister placement was selected to produce a



S9506018.4

Figure 7.19. Cross-Section of Consolidated Fuel Assembly

one-eighth symmetry in heat generation within the basket, and to produce the maximum fuel temperature in the center of the MSB. The ORIGEN2 code was also used to predict the axial decay heat profiles, which are required for the COBRA-SFS model of the system.

Performance testing of the VSC-17 cask consisted of six test runs with three internal environments and four cask venting configurations, ranging from vents fully open to vents closed. In all tests, the cask was oriented vertically. Table 7.4 summarizes the test conditions.

COBRA-SFS calculations were performed for all six tests, both pre- and post-test, and are fully documented elsewhere. For purposes of validating Cycle 2, only two test cases are summarized here. Both are with vents fully open, one with helium backfill, the second with nitrogen (Nos. 1 and 5 from Table 7.4). Essentially the same model was used for both cases, the only significant differences being in the fluid properties, the boundary conditions, and the constraints on the flow paths into the air annulus between the basket assembly and the concrete cask shell. The COBRA-SFS input model for the PSN/VSC-17 cask is described in the following subsection.

#### 7.3.2.2 COBRA-SFS Model of PSN Cask

Geometric and power distribution symmetries allowed modeling of the VSC-17 cask with a one-half section of symmetry. The axial geometry of the cask is modeled with 12 axial nodes, plus a lower plenum region and an upper plenum region which are modeled using the optional plenum model



**Table 7.4. PSN/VSC-17 Cask Test Matrix and Peak Temperatures**

Run number	Vent configuration	Back-fill	Cask heat load (kW)	Ambient temperature	Estimated peak clad temperature
1	all open	helium	14.9	21°C (70°F)	321°C (610°F)
2	2 inlet vents blocked	helium	14.9	23°C (73°F)	334°C (633°F)
3	all inlet vents blocked	helium	14.9	23°C (73°F)	378°C (712°F)
4	all inlet and outlet vents blocked	helium	14.9	22°C (72°F)	381°C (718°F)
5	all open	nitrogen	14.9	24°C (75°F)	376°C (709°F)
6	all open	vacuum	14.9	24°C (75°F)	397°C (747°F)

in the code. A diagram of the axial noding is shown in Figure 7.20. All channels, rods, and slabs see the same axial noding, but flow and heat transfer connections to the plena are determined by user input for specific channels and slab nodes.

The physical structure of the cask is modeled with 280 slab nodes representing the concrete shell, steel liner, basket shell and assembly, and the fuel canisters. Figure 7.21 shows a cross-sectional diagram of the COBRA-SFS model, with 10 slab nodes for the MSB outer shell, 30 for the steel liner, and 50 slab nodes modeling the cask concrete shell. The outermost ten of these nodes are zero-thickness surface nodes, for accurately calculating the contribution of radiative and convective heat transfer to the environment. Figure 7.22 shows a more detailed cross-sectional diagram of the COBRA-SFS model of the structures within the multi-assembly basket. The basket itself is modeled with 122 slab nodes, and the fuel canisters within the basket are modeled with 68 slab nodes.

Because of the close packing of the rods and the consequent flatness of the radial temperature gradient across the fuel array, the rods within each canister were modeled by lumping rings of rods into average rods with total power generation equal to the sum of all the rods each average rod represents. Figure 7.23 illustrates the rod lumping pattern for one assembly. All assemblies used this pattern, including those cut by the line of symmetry.

The flow subchannels within the fuel canisters were also lumped into average channels, consisting of all the subchannels enclosed between two radial rings of rods combined in the lumped rod model. The gap between the outermost rods and the canister wall was modeled with a single average channel on each wall, on the assumption that the close-packed triangular array of rods would remain centered within the canister. This assumption is somewhat unphysical, given the actual geometry of the canister, as illustrated in Figure 7.19, but it is conservative in that it presents the greatest possible thermal resistance for heat transfer between the rods and the wall of the canister. It also avoids the

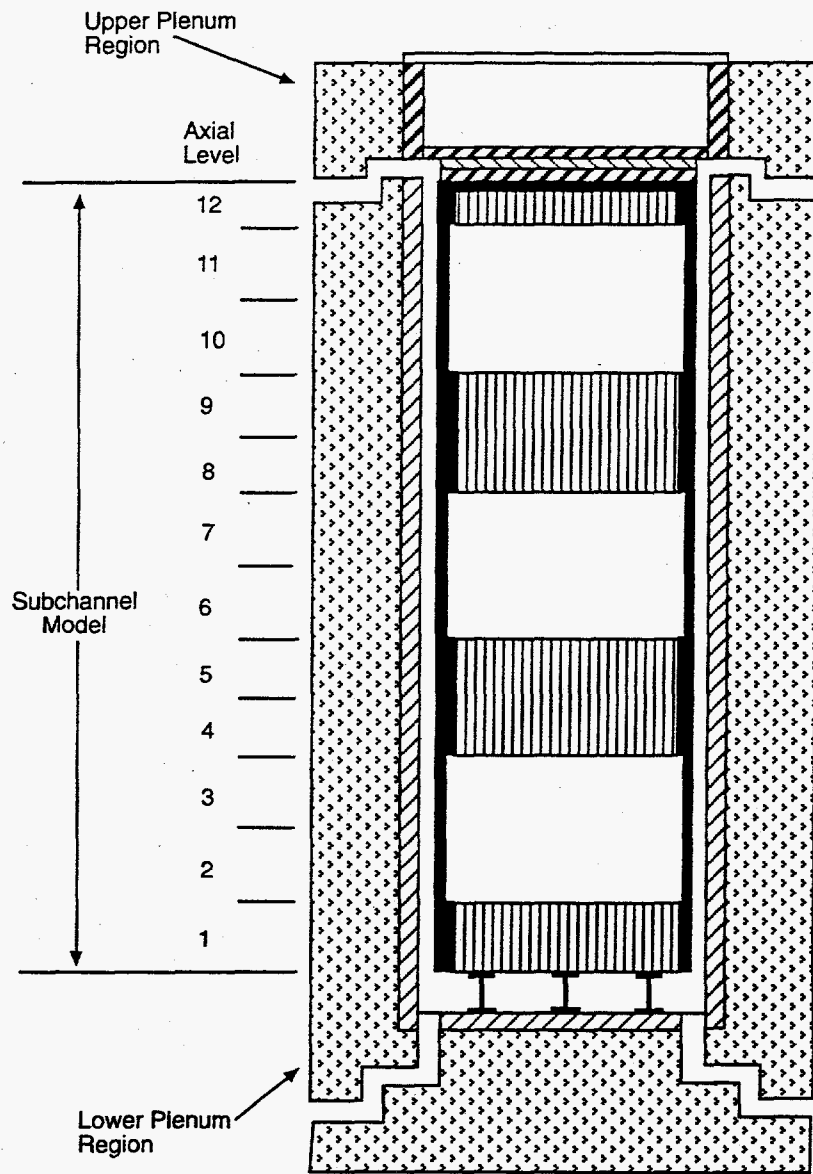
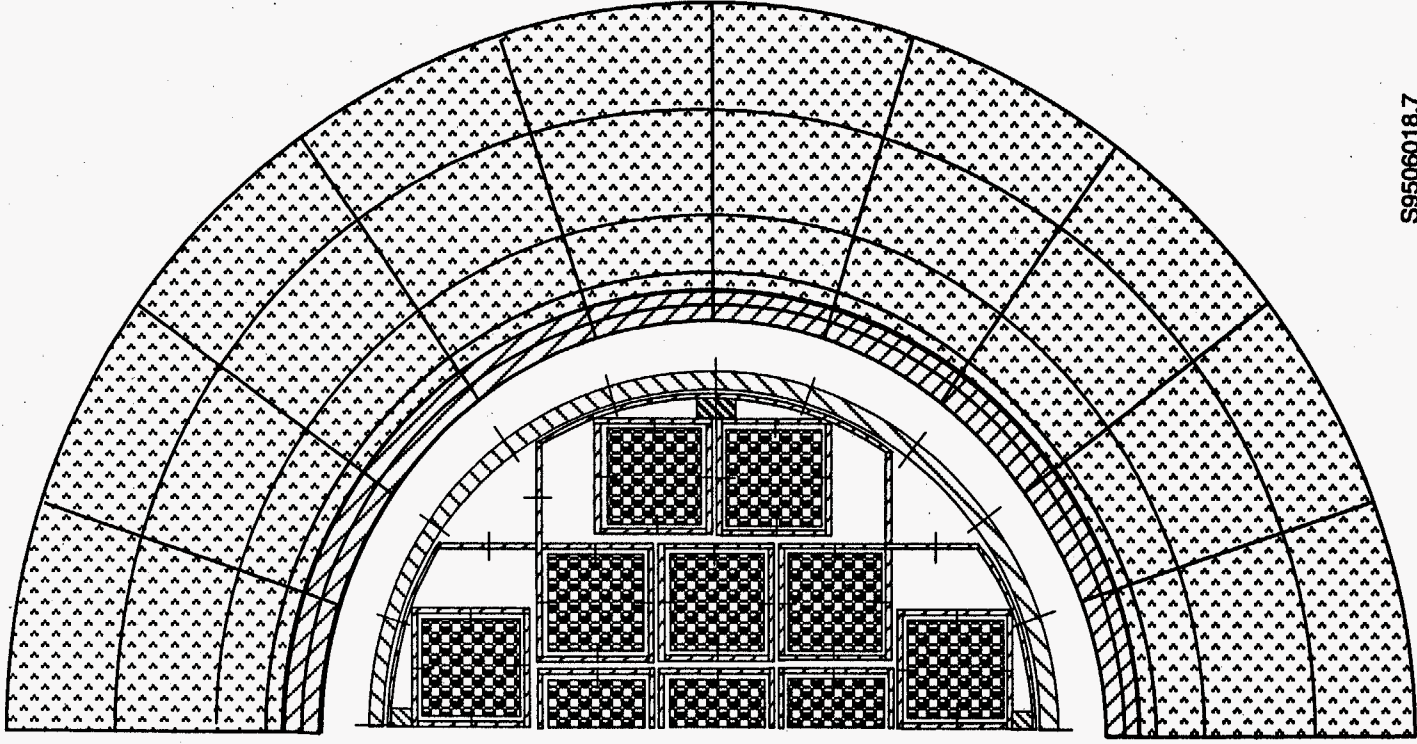


Figure 7.20. Axial Noding for COBRA-SFS Model of PSN/VSC-17 Cask

problem of having to determine the actual orientation of the fuel rods within the canister, and having to obtain estimates of appropriate values for contact resistance between the rods and the wall.

The axial decay heat power profile determined from ORIGEN2 calculations was applied to all fuel, and the radial power distribution was also determined from ORIGEN2 calculations that were based on gamma scans of spent fuel from the Turkey Point plant. This fuel was similar to the fuel actually loaded into the cask.

As with the TN24P cask, the COBRA-SFS predictions for the flow and temperature fields within the PSN/VSC-17 cask required solving for conductive, convective, and radiative heat transfer. Conduction heat transfer in the nodes modeling the solid structure of the cask was modeled using



S9506018.7

Figure 7.21. COBRA-SFS Model of PSN/VSC-17 Cask: Half-Section of Symmetry

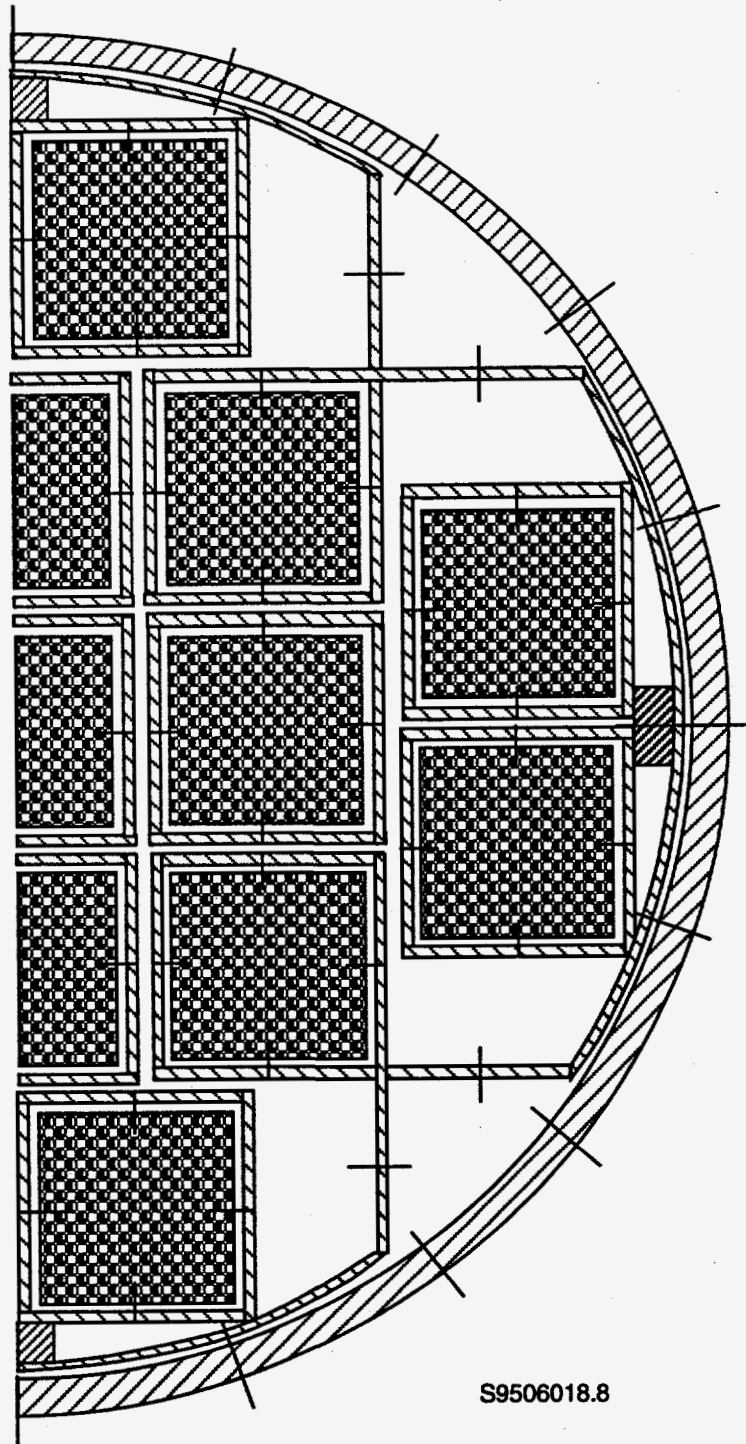
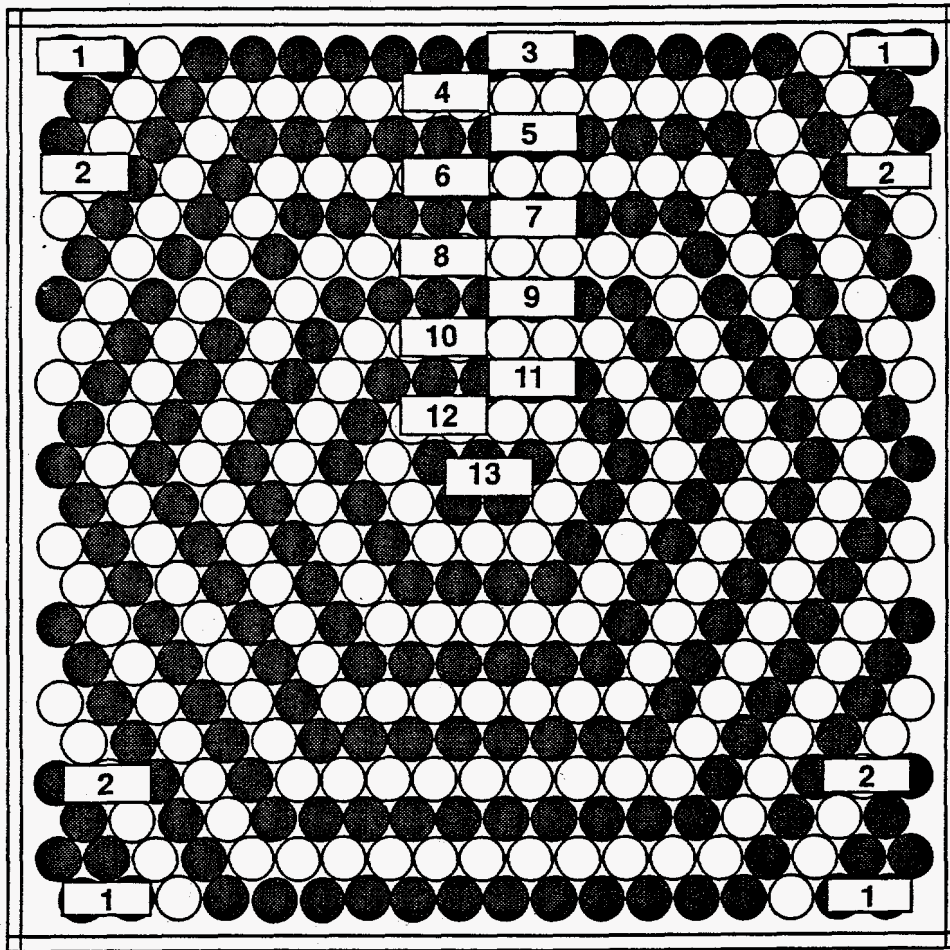


Figure 7.22. COBRA-SFS Model of Structures Within the MSB in the PSN Cask



S9506018.2

Figure 7.23. COBRA-SFS Model of Consolidated Fuel Assemblies in PSN Cask

thermal resistances based on the material properties of the cask body and internals. The flow field within the MSB was modeled by imposing an unspecified uniform pressure drop on all assemblies within the basket, and a zero net flow boundary condition. The flow boundary conditions on the assembly representing the annulus between the MSB and the concrete shell, which was open to the atmosphere through the cask vents, were modeled with a specified pressure drop corresponding to the gravity head of a column of air (at ambient temperature and pressure equal to the height of the distance between the cask inlet and outlet vents). For ambient conditions of 0.1014 MPa (14.7 psia) and 25°C (77°F), the pressure drop is 48.5 Pa (0.00703 psi).

Axial friction losses in the assemblies containing fuel rods were specified at a value typical for close-packed rods, as

$$f = \frac{40.0}{Re}$$

In the open assemblies within the basket, and in the annulus between the basket and the steel-lined concrete shell, the friction factor was specified at a value typical for low-speed flow in a duct, as

$$f = \max(f_i, f_e)$$

$$\text{where } f_i = 0.0850 \text{ Re}^{-0.25}$$

$$f_e = 96.0 \text{ Re}^{-1.0}$$

For the narrow gap between the fuel canisters and the supporting basket structure, it is assumed that the flow will always be laminar, and the laminar flow correlation given above for the vented annulus is used in those assemblies as well.

Form drag losses due to the structure of the end closures on the fuel canisters were modeled using a very large loss coefficient of  $9.0 \times (10^5)$  at the top and bottom of each assembly with fuel rods. For the open assemblies within the basket, the inlet and outlet losses were modeled with a more modest loss coefficient of 1.5. Inlet and outlet losses for the annular assemblies, which included the annuli between the fuel canisters and the basket cells, and the annulus that was vented to the atmosphere, were specified with a loss coefficient of 6.0. The channels comprising this annular assembly also included area variations to simulate the effects of the flow paths through the inlet and outlet ports.

Because of the extremely low flow rates in the channels formed by the close-packed rods within the assemblies, heat transfer between the rods and the fluid is modeled with a Nusselt number of 1.0, on the assumption that the dominant mode would be conduction. In the open assemblies within the basket, however, convection can play a more significant role in the heat transfer. For channels in these assemblies, the Nusselt number is defined using the relation

$$\text{Nu} = \max(\text{Nu}_i, \text{Nu}_e)$$

$$\text{where } \text{Nu}_i = 0.102 \text{ Re}^{0.914} \text{ Pr}^{0.4}$$

$$\text{Nu}_e = 4.80$$

For the natural recirculation of air through the assembly modeling the annulus vented to the atmosphere, the Dittus-Boelter heat transfer coefficient was used for turbulent flow, with a constant value for laminar flow. For these channels, the Nusselt number defined as

$$\text{Nu} = \max(\text{Nu}_i, \text{Nu}_e)$$

$$\text{where } \text{Nu}_i = 0.023 \text{ Re}^{0.8} \text{ Pr}^{0.4}$$

$$\text{Nu}_e = 7.54$$

Radiative heat transfer within the cask was modeled using grey body exchange factors calculated with the RADGEN code for the assemblies containing fuel rods. For the open assemblies within the basket, and for the annulus surrounding the basket, the black body view factors were specified in the COBRA-SFS input, and the code automatically calculated the grey body view factors for these enclosures. For radiative heat exchange with the environment from the cask exterior, the same models as those employed in the TN24P calculations were used.

### 7.3.2.3 COBRA-SFS Results for PSN Cask Model

The results obtained with COBRA-SFS for the tests performed in the PSN/VSC-17 cask are documented elsewhere. Sample results are presented here, to illustrate the performance of Cycle 2. Table 7.5 shows the peak clad temperatures in the fuel and in the concrete shell of the cask for the two validation tests.

For the helium case, the peak clad temperature predicted with COBRA-SFS is within 1°C of the measured value, and the peak temperature in the concrete is within 3°C of the measured value. This is a difference of less than 0.5% for the peak clad temperature, and a difference of about 4% for the peak temperature in the concrete. These results are well within the uncertainty in the measured values, and show that COBRA-SFS does an excellent job of predicting the heat transfer for conditions where conduction is dominant.

For the nitrogen case, the code overpredicts the peak clad temperature by 10°C, and also overpredicts the peak temperature in the concrete by 11°C. The predicted values are in reasonable agreement with the data, but the larger differences reflect the greater sensitivity of the results to modeling uncertainties when convective heat transfer is more important.

Figures 7.24 and 7.25 show calculated and measured axial temperature profiles in the PSN/VSC cask for the helium and nitrogen back-fill cases, respectively. The measured temperature profile from the thermocouple lance in the center assembly is compared to the calculated peak rod temperature. Measured temperatures from the multi-assembly basket are compared with the calculated temperatures in slab node 94, the slab node corresponding most closely with the thermocouple locations on the MSB. The measured temperatures on the cask surface within the vented annulus are compared to the calculated temperatures in slab node 83, modeling one segment of the cask shell.

**Table 7.5. Peak Temperature Predictions for PSN/VSC-17 Cask Tests**

Run number	Vent config./ Back-fill	Peak Clad Temperature		Peak Concrete Temperature	
		Measured	COBRA-SFS prediction	Measured	COBRA-SFS prediction
1	all open/helium	316°C (601°F)	317°C (602°F)	69°C (157°F)	72°C (162°F)
5	all open/nitrogen	366°C (691°F)	376°C (708°F)	72°C (161°F)	83°C (182°F)

Data vs COBRA-SFS Calculations (Helium)

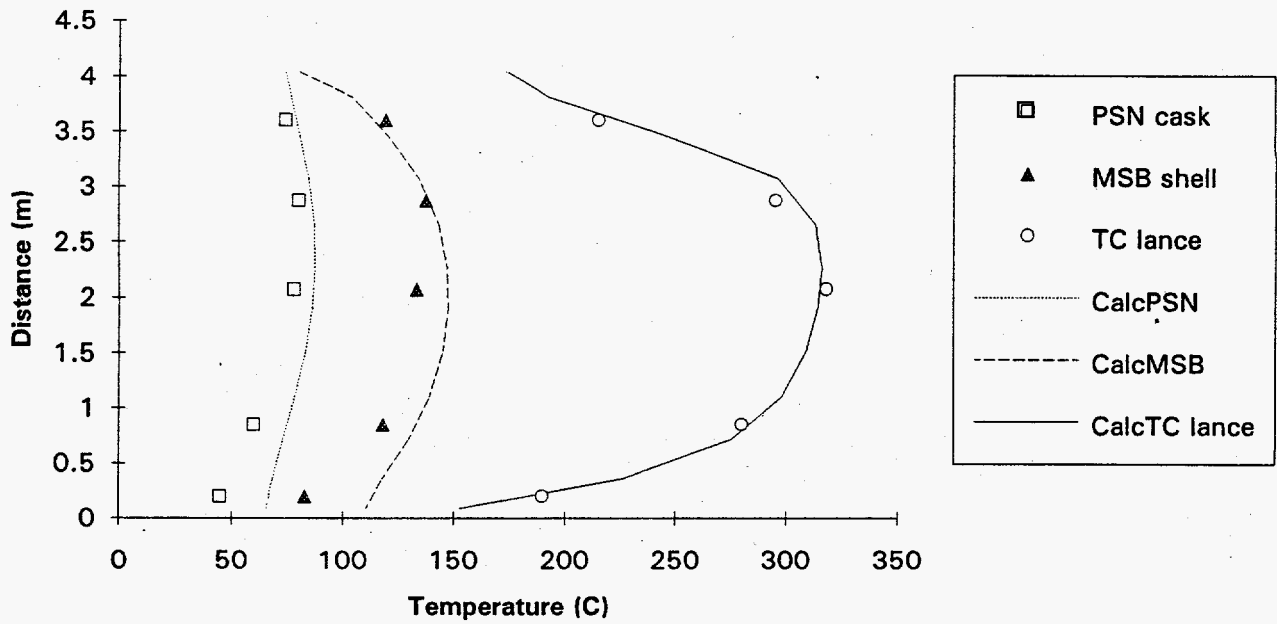


Figure 7.24. Comparison of COBRA-SFS Calculations with Measured Temperatures in the PSN/VSC-17 Cask; vertical orientation, vents open, helium back-fill

Data vs COBRA-SFS Calculations (Nitrogen)

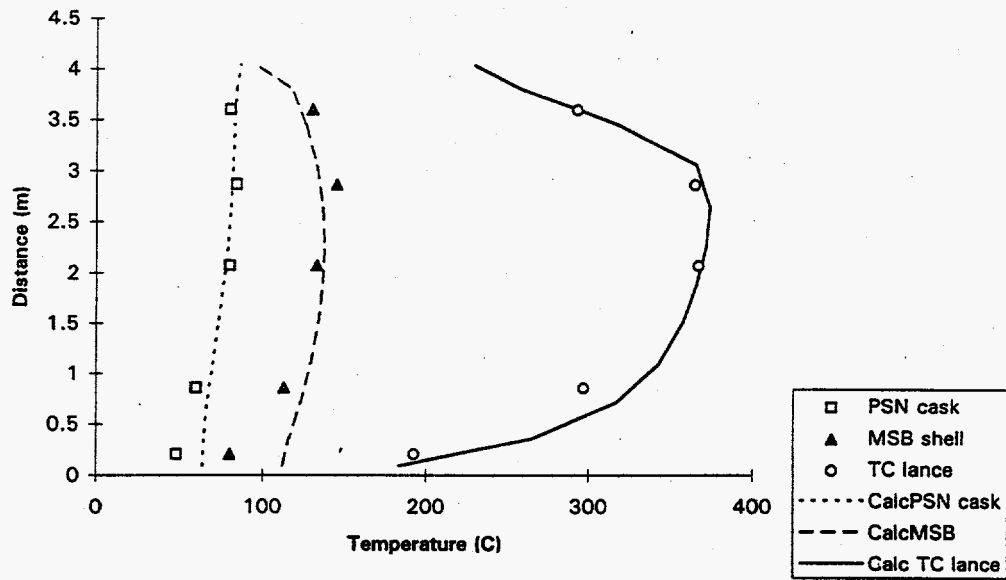


Figure 7.25. Comparison of COBRA-SFS Calculations with Measured Temperatures in the PSN/VSC-17 Cask; vertical orientation, vents open, nitrogen back-fill



For both the helium back-fill case (Figure 7.24) and the nitrogen back-fill case (Figure 7.25), the COBRA-SFS calculations are in good agreement with the measured temperatures. These results are well within the experimental and modeling uncertainties inherent in the comparison.

## 7.4 Validation on Alternative Platforms

In transferring the COBRA-SFS code to a new computing platform, the first concern is to determine that the code gives the same results as before. Also of interest, however, is how well the code runs on the new platform. In general, the main motivation for moving to a different platform is usually to gain some improvement in performance. For this code, improved performance means faster number-crunching.

The code was originally developed to run on large mainframe computers, such as the CDC and Cray machines. With the advent of powerful desktop workstations, however, the code has shown itself easily adaptable to the new computing environment (see Section 8.4). Versions of the code have been developed for the Sun SPARCstation, the IBM RS6000, an IBM clone with a Pentium processor, the SUN4 and DEC workstations, Hewlett-Packard's HP9000 workstation, and the Macintosh workstation.

Validation test cases have been run on these platforms to verify that the code gives the same results for the various versions. The run times for the validation cases on the various platforms are summarized in Table 7.6. The three test cases consist of models of one small test assembly and two relatively large multi-assembly casks.

The Mitsubishi test case is a model of a single-assembly test section, consisting of 256 sub-channels, 225 rods, and 8 slab nodes (see Section 7.2.2). The TN24P test case is the one-eighth section of symmetry model of the vertical cask with helium back-fill. This model contains 7 assemblies, with a total of 389 channels, 450 rods, and 51 slab nodes (see Section 7.3.1.2). The PSN test case is the half-section of symmetry model of the PSN/VSC-17 cask with nitrogen back-fill, vents fully open. This model contains 22 assemblies, with a total of 214 channels, 130 rods, and 249 slab nodes (see Section 7.3.2.2).

**Table 7.6. Run Time on Alternative Platforms (cpu seconds)**

Test case	Platforms						
	Workstations						CRAY (uniCOS)
	Sun	RS6000	Intel	DEC	HP9000	Macintosh	
Mitsubishi	818.09	176.21	249.64	107.80	217.00	366.00	80.10
TN24P	4599.66	988.69	641.26	641.26	1334.00	4131.00	2189.18
PSN cask	3494.54	714.33	482.17	482.17	980.32	1594.00	546.61

The only significant differences observed in these test case results are in the run times on the different platforms. As might be expected, the newer platforms are significantly faster than the older ones. It should be noted that the run times for all of the workstations compare quite favorably to the run times on the Cray mainframe.

## 8.0 Code Structure and Installation

This section describes COBRA-SFS from the programmer's standpoint. Section 8.1 describes the functional hierarchy in the code, and Section 8.2 presents a brief description of each subroutine. The dimension parameters used to define the sizes of the various arrays in the code are described in Section 8.3. This section also includes the dimension parameters for the RADGEN code. Platform-dependent coding is described in Section 8.4. Installation instructions and a description of the files that make up the code package are included in Section 8.5.

### 8.1 Functional Hierarchy of COBRA-SFS

The COBRA-SFS code is derived from COBRA-WC (George et al. 1980) and retains essentially the same structure. The main program COBRA, calls various routines to process the input and set up the problem, to perform the solution, and to process the output. At the beginning of a case, subroutine CLEAR is called to zero all common blocks. After the input has been processed, the boundary conditions are set (or updated for the new time step values in transient cases), and the solution is performed. After the solution is complete, the output routines are called to write information to the output files, and to save a restart file.

#### 8.1.1 Input Routines

In program COBRA, subroutine ECHO is called to write the input file verbatim to the output file, providing an exact record of the input for future reference in archived output files. The input is then processed, by execution of the driver routine for the input processing, subroutine SETUP. Subroutine SETUP calls subroutine SETIN, which reads the input group by group, from logical unit 2. The one exception is geometry input, group CHAN. Subroutine SETIN calls subroutine GROUP4 to process the input for group CHAN. After the input is read, subroutine SETUP calls subroutine SETOUT to process the input, fill the various arrays required for the problem description, and initialize the flow and energy field variables.

On a restarted case, the restart file is read from logical unit 9, in subroutine RESTRT. (This file would have been created in a previous run, in which the common arrays are written to logical unit 8 for each time step.) On a restart, additional input may also be read from the input file on logical unit 2, to define such things as changed boundary conditions, new forcing functions, or different models and correlations. This new input is processed through SETUP, as described in Section 6.0.

#### 8.1.2 Solution Routines

After the input is processed, control returns to the main program COBRA, and the calculation enters the time step loop. Arrays are filled with the specified boundary conditions, various control parameters are set, and the solution is initiated by calling subroutine RECIRC. Subroutine RECIRC is the driver routine for the solution, and also performs the solution of the mass continuity equation for each cell of the system, and the axial rebalancing.

Each external iteration of the solution procedure begins with an axial sweep of the system from the bottom to the top axial level. In this sweep, subroutine ENERGY is called to solve the fluid and wall node energy equations. If the plenum model is specified by user input, heat transfer to or from the upper and lower plena is calculated by calling subroutine EXPROP. Subroutine REHEAT is called to calculate the heat flux terms for heat transfer connections between rod, slab and fluid nodes, and subroutine HOTROD is called to solve for the temperature of the rod surface. This will include internal rod material temperatures, if the appropriate options have been specified and fuel material properties defined by the input. Subroutines MOMENT and DIFFER are called to solve the axial and lateral momentum equations for the fluid nodes. After the fluid energy equation has been solved, subroutine PROP is called to calculate new values for the fluid and material properties.

The continuity equation is then solved, with optional axial rebalancing, in a level-by-level and channel-by-channel internal iteration. The rebalancing is performed in subroutine ONED, and the pressure matrix is solved by calling a banded matrix solver in subroutine SOLVER. When the internal iteration on the continuity solution is completed, new pressures are calculated and the maximum energy, momentum, and continuity errors are evaluated. If these errors meet the convergence tests, the solution is complete. If not, the solution continues on to the next external iteration, but first checks for negative axial flows in the system. If any are found, subroutine ENERGY is called for a reverse sweep of the system, from the top axial level to the bottom.

At the completion of each external iteration, a summary line containing the current value of the peak clad temperature, its location, and all convergence errors is written to the output file. The solution continues iterating until all convergence criteria have been satisfied, or the maximum number of iterations is exceeded. Subroutine RECIRC then returns control to the main program, for processing of the output. If the problem is a transient, the solution is repeated for the next time step.

### 8.1.3 Output Routines

Main program COBRA calls subroutine RESULT to write the output data to logical unit 3. This output consists of fluid conditions (pressure drop, enthalpy, temperature, density, flow rate, mass flux, and velocity) for each node in all channels, flow conditions (mass flux and velocity) in each node of all gaps for crossflow between channels, temperatures for all rod nodes, slab nodes, and boundary nodes, and plenum fluid and solid node temperatures. The default is to print all information about all solid and fluid nodes, but the level of detail can be regulated by user-input, to specify output only for certain channels, rods, gaps, and walls.

In addition to the printer output file, COBRA also calls subroutine DUMPIT to save a restart file for the case, on logical unit 8. If the option has been specified by input, subroutine POST is also called from COBRA to write information to a file on logical unit 1 for post-solution plotting.

## 8.2 Subroutine Descriptions

The COBRA-SFS code consists of the main program COBRA and 32 subroutines. This subsection describes each subroutine in the code. The call list is given, with the definition of each variable in the list, along with a brief explanation of the routine's function. The first routine described is the main program COBRA; the subroutines are listed in alphabetical order.

### Main Program COBRA

This is the driver routine for the COBRA-SFS code. It calls the input routines to set up the problem, and contains the time step loop to drive the solution in both steady state and transient calculations. When the solution is complete, it calls the output routines to report the results of the calculation.

Program COBRA calls subroutines DATTIM, CLEAR, ECHO, RESTRT, RESULT, SETUP, CURVE, PROP, PREFIX, RECIRC, POST, and DUMPIT.

### Subroutine AREA

This subroutine fills the channel geometry arrays, based on the input provided in groups CHAN and VARY. It fills the axial cross-sectional area array for each node of each channel, and calculates the local hydraulic diameter using the input values for area and wetted perimeter. It also fills the gap width array with the input values, and calculates all area and gap-width variations specified by input in group VARY.

call list arguments: area(j,jx)

j - axial level index

jx - axial level index

Subroutine AREA calls CURVE, and is called by RECIRC and SETIN.

### Subroutine CLEAR

This subroutine sets all elements of the common blocks to zero. It uses the function LOCA to find the beginning and ending addresses of a common block, and sets to zero all addresses in between those endpoints. This subroutine is called at the beginning of each case.

call list arguments: none

Subroutine CLEAR calls function LOCA. Subroutine CLEAR is called by main program COBRA.

## Subroutine COLOC

This subroutine sets up the geometric locations of the boundaries of the fuel collocation nodes, to be used in the calculation of the fuel internal temperatures when the fuel collocation model is specified by input.

call list arguments: coloc(ia,isym,kft,kk,rmin,rmax,rcl,tcl)

ia - flag for noding geometry:  
= 1; planar geometry  
= 2; cylindrical geometry  
isym - flag for fuel geometry:  
= 2; solid fuel  
= 1; annular or plate fuel  
kft - fuel type index  
kk - absolute value of fuel type flag  
rmin - inner fuel radius (inches)  
rmax - outer fuel radius (inches)  
rcl - cladding radius (inches)  
tcl - cladding thickness (inches)

Subroutine COLOC is called by SETIN.

## Subroutine CURVE

This subroutine performs linear interpolation in tabular arrays. It is called to determine axial geometry variations, fluid and material properties, and boundary condition values.

call list arguments: curve(fx,x,f,y,n,j,isave)

fx - dependent variable for interpolation  
x - independent variable for interpolation  
f - dummy array for the dependent variable  
y - dummy array for the independent variable  
n - number of elements in the interpolation table  
j - error flag;  
= 10; indicates subroutine CURVE for output message  
isave - flag to use the same interpolation ratio as calculated  
in last call to subroutine CURVE

Subroutine CURVE is called by main program COBRA, and subroutines AREA, DIFFER, ENERGY, EXPROP, NETWORK, POST, PROP, REHEAT, RESULT, and SETIN.

## Subroutine DATTIM

This subroutine calls various system routines to determine the time and date of a calculation, and the cpu time used during execution. It contains platform dependent coding.

call list arguments: dattim(iopt,alph,xivar)

iopt - flag for target calculation;  
= 1; find elapsed cpu time  
= 2; find clock time  
= 3; find current date

alph - alphanumeric array containing formatting information for time  
and date printout

xivar - current clock time

Subroutine DATTIM calls system routines that provide date and time functions. It is called by main program COBRA, and subroutines ENERGY, MOMENT, POST, RECIRC, REHEAT, RESULT, and SETUP.

### Subroutine DIFFER

This routine is used to calculate the coefficients for the momentum solution and radial conduction in the fluid. It consists of four parts, which are called from different portions of the momentum and energy solution. Part 1 calculates the terms for the fluid energy equation. Part 2 calculates the crossflow resistance term and turbulent cross-flow mixing for the momentum solution. Parts 3 and 4 calculate the local loss coefficients in each node due to grids and other obstructions in the flow field. The Reynolds number dependencies for the momentum equations are also calculated for each node, based on local fluid conditions.

call list arguments: differ(ipart,j,jx)

ipart - flag for different parts of the subroutine

j - axial level index

jx - axial level index

Subroutine DIFFER calls CURVE, and is called by MOMENT and RECIRC.

### Subroutine DUMPIT

This subroutine writes out the solution arrays and calculated results to a binary file on logical unit 8, for use in a future restart. All information needed to obtain a complete output of the calculation is saved, along with all arrays needed to continue the solution for more iterations or more time steps.

call list arguments: dumpit(ntsav)

ntsav - index number of the time step being written to the  
dump file on lun 8.

Subroutine DUMPIT calls Function LOCA, and is called by main program COBRA.

### Subroutine ECHO

This subroutine writes a line-for-line listing of the input file to the output file.

call list arguments: none

Subroutine ECHO is called from main program COBRA.

### Subroutine ENERGY

This subroutine sets up and solves the combined rod and fluid energy equations at each axial level using the method of over-relaxation. New values of the fluid enthalpy are determined for each node at each axial level, and new slab node temperatures are also calculated. When the plenum model is used, the subroutine EXPROP is called to determine the upper and lower plenum temperatures.

call list arguments: energy(j, jp1, jm1, ipart)  
j - axial level index  
jp1 - index of next axial level  
jm1 - index of previous axial level  
ipart - flag for direction of solution sweep;  
= 1; from bottom to top  
= 2; from top to bottom

Subroutine ENERGY calls subroutines DATTIM, CURVE, and EXPROP. Subroutine ENERGY is called by RECIRC.

### Subroutine EXPROP

This subroutine sums the exit flows and enthalpies at the top and bottom of the channels when the plenum model is used, and calculates the average mixed fluid temperature for the upper and lower plenum. The regional temperatures for the plena are also calculated, based on heat transfer from the fluid, and conduction from slab nodes in contact with the plenum regions.

call list arguments: exprop(j, jx, l1)  
j - axial level index  
jx - axial level index  
l1 - plenum flag;  
= 1; lower plenum  
= 2; upper plenum

Subroutine EXPROP calls CURVE to determine local fluid properties, and is called by ENERGY.

### Subroutine FORCE

This subroutine calculates the forced diversion crossflow due to form losses seen by the axial flow in the channels.



call list arguments: force(j,jx)  
j - axial level index  
jx - axial level index

Subroutine FORCE is called by RECIRC.

#### Subroutine GROUP4

This subroutine reads the input for group CHAN in the processing of the input stream. It is called by subroutine SETIN, and calls subroutine LIMITS to check dimension parameters against the problem input.

call list arguments: group4(ierror)  
ierror - error flag

Subroutine GROUP4 calls subroutine LIMITS.

GROUP4 is called by SETIN.

#### Subroutine HOTROD

This subroutine solves for the temperature distribution in each fuel rod, using the fluid temperature and a boundary condition.

call list arguments: hotrod(j,jx)  
j - axial level index  
jx - axial level index

Subroutine HOTROD is called by RECIRC.

#### Subroutine LIMITS

Subroutine LIMITS is called during the processing of the input to check the problem input against the dimensions specified in the parameter statements in the compiled version of the code. If the input exceeds the allowed dimensions, an error message is written to the output file and execution terminates.

call list arguments: limits(num,minval,maxval,avar,apar,agroup,card,error)  
num - value specified by input  
minval - minimum value permitted for num  
maxval - maximum value permitted for num  
avar - alphanumeric identifier for variable num  
apar - alphanumeric identifier of parameter num is to be checked against  
agroup - alphanumeric identifier of current input group

card - input line number  
error - alphanumeric error flag;  
= no ; num is within permitted limits  
= yes ; num is outside permitted limits

Subroutine LIMITS is called by GROUP4 and SETIN.

### Subroutine LOAD

Subroutine LOAD is a generalized input processor that reads information from the input file for selected input lines that specify data in tabular form.

call list arguments: load(x,y,z,min,max,limit,step,image,card,lu)

x - first dummy input array  
y - second dummy input array  
z - third dummy input array  
min - number of elements in the input table  
max - maximum number of entries permitted in the input table  
limit - maximum number of table entries on a single line of input  
step - number of elements in a set for a table entry  
image - alphanumeric variable carrying format description; 12e5.0  
card - input line counter  
lu - input file logical unit number

Subroutine LOAD is called by SETIN.

### Function LOCA

This function uses a system routine to determine the address of a specific variable. The system routine is platform dependent. The variables correspond to the endpoints of the common blocks, and are used in subroutine CLEAR to find the beginning and end of memory locations to set to zero before initiating a new COBRA-SFS calculation.

call list arguments: loca(ae,ab)

ae - ending variable of a common block  
ab - beginning variable of a common block

Function LOCA is called by subroutines CLEAR, DUMPIT, and RESTRT. Function LOCA calls a system routine such as %loc, locf, or loc, depending on the platform version.

### Subroutine MIX

This subroutine calculates the turbulent cross-flow mixing parameter for the momentum equation, using a user-specified correlation.

call list arguments: mix(j)  
j - axial level index

Subroutine MIX is called by subroutines MOMENT and RECIRC.

### Subroutine MOMENT

This subroutine solves the linearized momentum equations for the axial and lateral directions. The flow in all channels is solved at each axial level by direct elimination, using the matrix solver in subroutine SOLVER. MOMENT is called twice from the solution driver routine, RECIRC. In the first call, the flow and flow derivatives are initialized for the first axial level only. In the second call, the flow and flow derivatives are calculated for the current axial level. The optional models for fluid-fluid shear are also included, when specified by user input.

call list arguments: moment(j,jx,jm1,jp1,ipart)  
j - axial level index  
jx - axial level index  
jm1 - previous axial level  
jp1 - next axial level  
ipart - flag for subroutine section;  
= 1; section for axial level 1  
= 2; section for axial levels  $j > 1$

Subroutine MOMENT calls subroutines DATTIM, DIFFER, MIX, and SOLVER.

Subroutine MOMENT is called by RECIRC.

### Subroutine NETWORK

This subroutine implements the network model, providing a specialized version of the pressure drop boundary condition. It provides initial flow and pressure values for all channels and assemblies, to satisfy the uniform pressure drop boundary conditions for the orifice losses specified by input.

call list arguments: network(iter)  
iter - solution section flag

Subroutine NETWORK calls CURVE, and is called by RECIRC and RESULT.

### Subroutine ONED

This subroutine solves a one-dimensional formulation of the continuity equations for each assembly, providing an approximate solution for the pressure and axial flows. This approximate solution is used as a rebalancing correction to the overall continuity solution in each fluid node.

call list arguments: oned(nass)  
nass - assembly index

Subroutine ONED is called by RECIRC.

### Subroutine POST

This subroutine writes data to an external file on logical unit 17. This data is organized for plotting COBRA-SFS output results, and consists of geometry data describing the system model, rod temperatures, slab temperatures, fluid temperatures, and local pressures. The information is written to the file for every time step, including the initial steady state.

call list arguments: post(nt,text)  
nt - time step number  
text - alphanumeric array containing case title

Subroutine POST calls DATTIM and CURVE. POST is called by main program COBRA.

### Subroutine PREFIX

This subroutine sets up the arrays governing the interaction terms used in the coefficient matrix for the combined rod and fluid energy equation.

call list arguments: none

Subroutine PREFIX is called by main program COBRA.

### Subroutine PROP

This subroutine sets up the properties tables for the fluid and solid material nodes. It also computes the local fluid density as a function of pressure and enthalpy or temperature, and calculates the local values of the heat transfer coefficient for convection and friction factors for flow losses and form drag.

call list arguments: prop(ipart,j,jx,jm1,jp1)  
ipart - flag for section of subroutine  
          = 1; fill the fluid properties arrays from the input tables  
          = 2; compute local fluid properties and parameters  
j - axial level index  
jx - axial level index  
jm1 - previous axial level  
jp1 - next axial level

Subroutine PROP calls CURVE to interpolate in the properties tables. Subroutine PROP is called by main program COBRA and subroutines RECIRC and SETIN.

### **Subroutine RECIRC**

This subroutine is the driver for the numerical solution. It calls **MOMENT** and **ENERGY** to solve the momentum and energy conservation equations. It solves the continuity equation for new pressures and flows, and tests for convergence on the energy, momentum, and continuity errors.

call list arguments: none

Subroutine **RECIRC** calls subroutines **DATTIM**, **AREA**, **PROP**, **FORCE**, **MOMENT**, **MIX**, **DIFFER**, **REHEAT**, **ENERGY**, **HOTROD**, **NETWORK**, **ONED**, and **SOLVER**. **RECIRC** is called by main program **COBRA**.

### **Subroutine REHEAT**

This subroutine computes the heat flux dependent terms for the rod energy equation. It also calculates the coefficients for heat transfer from the rods to the fluid, using the finite difference solution of the solid material energy equation.

call list arguments: reheat(j,jx,ipart)

j - axial level index

jx - axial level index

ipart - flag for direction of solution sweep;

= 1; solving from bottom to top

= 2; solving from top to bottom

Subroutine **REHEAT** calls **DATTIM**, **CURVE**, and **VARPRP**. Subroutine **REHEAT** is called by **RECIRC**.

### **Subroutine RESTRT**

This routine reads a restart file created in a previous code run in order to continue the calculation. This file was written to logical unit 8 by subroutine **DUMPIT** in an earlier run. Subroutine **RESTRT** reads this file from logical unit 9. The user has the option of simply printing the output for the original case, or may specify additional iterations or time steps in the solution. Some changes can be made in the input to modify such things as the problem boundary conditions, heat transfer connections, material properties, power distributions, or forcing functions. However, changes in the rod or channel geometry are not permitted on a restart.

call list arguments: restrt(ntstrt,istart)

ntstrt - time step number of restart

istart - iteration number to begin restarted calculation

Subroutine **RESTRT** calls Function **LOCA**, and is called from main program **COBRA**.

## Subroutine RESULT

This subroutine writes the calculation results to the output file for printing. Fluid conditions of pressure, temperature, flow rate, enthalpy, velocity, and mass flux are printed for every node of each channel and gap of each assembly. Temperatures are printed for each rod node and each slab node. Temperatures are printed for each region of the lower and upper plenum, and for all boundary nodes. The user may limit the amount of output by input specifications in group OUTF, but the default is to print all information for all fluid and solid material nodes and boundaries.

call list arguments: result(nt,text)

nt - time step number

text - alphanumeric array containing the case title specified by input

Subroutine RESULT calls subroutines DATTIM, NETWORK, and CURVE. RESULT is called by main program COBRA.

## Subroutine SETIN

This subroutine reads and processes the input data from all groups except group CHAN. It performs error checks on the input arrays against the dimension parameters, and provides diagnostic error messages if any parameter is too small for the particular case.

call list arguments: setin(igperr,error,tpost,ntpost)

igperr - flag for input processing;

= 0; print only new input data

= 1; print all input (including previously defined input on a restart)

= 2; print only input data from group OPER describing the operating conditions for the case

= 10; print all input data, then stop execution

error - flag for input error;

= yes ; input error encountered, halt execution

= no ; no input errors detected

tpost - time interval for post-processing plot dumps

ntpost - time step interval for post-processing plot dumps

Subroutine SETIN calls subroutine LOAD to read tabular input arrays, and subroutine LIMITS to check input values against array dimensions. It calls subroutine GROUP4 to process the input for group CHAN. It calls subroutines PROP, AREA, and COLOC to initialize property, geometry, and temperature arrays. It calls CURVE to interpolate in various tables. Subroutine SETIN is called by SETUP.

### Subroutine SETOUT

This subroutine processes the input data and writes it to the output file in an organized manner for each input group. It performs some additional dimension parameter checks, on array sizes not directly specified by input, but calculated from input parameters. It also writes to the output file a summary of the dimension parameters required for a given case, along with the value defined for the given parameter in the parameter statement.

call list arguments: none

Subroutine SETOUT is called by subroutine SETUP.

### Subroutine SETUP

This subroutine is the driver routine for input processing. It computes miscellaneous input-dependent parameters and sets up the array for form loss coefficient as a function of axial position. It also fills the temperature boundary node arrays, and checks the grey body view factors in tape10 for conservation of energy.

call list arguments: setup (text, tpost, ntpost)

text - alphanumeric array containing the case title  
tpost - time interval for post-processing plot dumps  
ntpost - time step interval for post-processing plot dumps

Subroutine SETUP calls subroutines DATTIM, SETIN, and SETOUT. SETUP is called by main program COBRA.

### Subroutine SOLVER

This subroutine solves a banded matrix by direct inversion. The matrix may be either symmetric or nonsymmetric, and must be of the form

$$[A] [x] = [B]$$

The matrix **[A]** is the banded coefficient matrix, **[x]** is the solution vector, and **[B]** is the resultant matrix.

For the symmetric case, the coefficient matrix is passed in an array dimensioned  $A(nrow[ibwl + 1 + ibwr])$ , where

nrow = number of rows in the matrix  
ibwl = number of nonzero columns to the left of the diagonal  
ibwr = number of nonzero columns to the right of the diagonal

The matrix diagonal appears in the  $A(nrow, ibwl + 1)$  location in the array.

For the nonsymmetric case, the coefficient matrix is passed in an array dimensioned A(nrow,[1+ibwr]). The matrix diagonal appears in the A(nrow,1) location of the array.

call list arguments: none

Subroutine SOLVER is called by subroutines MOMENT and RECIRC.

### Subroutine VARPRP

This subroutine calculates the variation of thermal properties for fuel rod nodes as a function of temperature, when the option for material properties that vary with temperature is specified by user input. The properties are calculated as a thermal storage term (density times the specific heat divided by the time derivative), and a conduction term (thermal conductivity divided by the node thickness). Values are calculated for each fuel rod node, and the density, specific heat, and thermal conductivity are a function of temperature.

call list arguments: varprp(rc,k,tz,nt,jx)

- rc - thermal storage term (density times specific heat divided by the time derivative) for the current temperature
- k - conduction term (thermal conductivity divided by the node thickness) for the current temperature
- tz - local fuel rod node temperature
- nt - fuel type index
- jx - axial level index

Subroutine VARPRP is called by subroutine REHEAT.

## 8.3 Dimension Parameters

Most of the variable arrays used in COBRA-SFS are stored in common blocks that are stored in COMDECKS. The COMDECKS are features of the WITNESS utility that work much like 'include' statements, allowing a single location for coding that can be inserted into many different subroutines when the compilable file is created. Appendix A contains a listing of the COBRA-SFS COMDECKS, and includes a brief definition of each variable. Array sizes in the COBRA-SFS common blocks are specified by means of parameter statements. This allows the user to tailor the storage requirements of the executable file to the size needed for a particular application. The dimension parameters are listed in Table 8.1, along with a description of the purpose each one serves, and the model input that is checked against it.



Table 8.1. Dimension Parameters for COBRA-SFS Code

Parameter	Definition
ie	width of energy and momentum coefficient matrix; $ie \geq 2*ibwl + 1$ , where <i>ibwl</i> is the maximum value of the difference between the index numbers of channels connected by lateral flow through a gap
ih	maximum number of second order connections to any channel (six for square arrays; nine for hexagonal arrays)
lir	maximum number of rods interacting with a given channel
lif	maximum number of thermal boundary connection types; limit for <i>nbtyn</i> on BDRY.1
lit	maximum number of assembly types
iu	maximum number of assemblies; limit for <i>nassem</i> on CHAN.1
liw	maximum number of loss coefficient forcing function profiles, and maximum number of friction factor and heat transfer correlations; limit for: <i>nflmc</i> ( <i>nass</i> ) on CHAN.6 <i>nhfvt</i> ( <i>nass</i> ) on CHAN.6 <i>npfvt</i> ( <i>nass</i> ) on CHAN.6 <i>mdflt</i> on CHAN.6 <i>nheat</i> on HEAT.1 <i>nfri</i> on DRAG.1 <i>nleff</i> on DRAG.1 <i>ngprfl</i> on OPER.1 <i>nqprfl</i> on OPER.1
lix	maximum number of flow separated regions within a single assembly
ma	maximum number of channels with area variations; limit for <i>nafact</i> on VARY.1
mc	maximum number of channels; limit for <i>nchanl</i> (sum of all values of <i>nchana</i> specified on CHAN.5)
mg	maximum number of lateral flow connections between channels; limit for <i>nk</i> (sum of all gaps in all assemblies, as defined by input on line CHAN.7)
mi	maximum number of connections to a channel (thermal plus flow)
mj	maximum number of assembly groupings in network model; limit for <i>nogrp</i> on DRAG.1
mk	maximum number of local axial flow blockages; limit for: <i>nblock</i> on DRAG.1 <i>nblokc</i> on DRAG.10
ml	maximum number of axial locations for gap and area variations; limit for: <i>naxl</i> on VARY.1 <i>ngxl</i> on VARY .1

Table 8.1. Cont'd

Parameter	Definition
lmm	maximum number of points in any input profile table; loss coefficient vs. Reynolds no. (nlcfp on DRAG.1) flow or pressure drop vs. time (ng on OPER.1) heat flux vs. time (nq on OPER.1) system pressure vs. time (np on OPER.1) inlet and exit enthalpies vs. time (nh and nhx on OPER.1)
mn	maximum number of fuel nodes in a rod; limit for $nc + 1$ (nc defined on RODS.1)
mo	maximum number of gaps coupled to any one gap for lateral momentum transport, plus one; (seven for square arrays, five for triangular arrays)
mp	maximum number of entries in each table of fluid properties, axial heat flux profile, and variable time step size; limit for:    nprop on PROP.1 nfprop (j) on PROP.2 nlcfp on DRAG.1 naxi(i) on OPER.16 itstep on CALC.1
mpp	total number of values permitted in interpolated property tables in 10°F increments; default is 4000, which allows 200 entries of 5 properties (specific heat, thermal conductivity, specific volume, viscosity and coefficient of thermal expansion) at each temperature in each of two tables.
mr	maximum number of fuel rods; limit for nrod (summation of all values specified for norods on RODS.2)
ms	maximum number of gaps that can have gap spacing variations; limit for ngaps on VARY.1
mt	maximum number of fuel types; limit for:    nfuel on RODS.1 nrodtp on RODS.1
mv	maximum number of axial heat flux profiles; limit for:    naflx(nass) on CHAN.6 naxp on OPER.1
mw	maximum number of slab nodes; limit for nwk on SLAB.1
mx	maximum number of axial nodes, plus one; limit for $ndx + 1$ (ndx is defined on CHAN.1)
my	maximum number of axial fuel type divisions; limit for knz on rods.7
id	maximum number of solid material property types; limit for:    nsprop on PROP.1 mat on SLAB.6
lip	maximum number of radial boundary regions; limit for nhsid(n) on BDRY.4

Table 8.1. Cont'd

Parameter	Definition
iq	maximum number of circumferential boundary nodes; limit for nwsid on BDRY.1
liz	maximum number of slab-to-slab and slab-to-fluid thermal connection types; limit for: nkss on SLAB.1 nksf on SLAB.1
mh	maximum number of slab-to-slab thermal connections for a solid structure node; limit for: nslab on SLAB.3
mq	maximum number of channels with local axial losses in any one assembly; limit for: nbl (i) on DRAG.3
nd	maximum number of fluid channels connected to a slab node
no	maximum number of boundary temperature types; limit for: nbtemp on BDRY.1
nv	maximum number of user-defined radiation groups (fvw array); limit for: nradg on RADG.1
lnz	maximum number of surfaces in any user-defined radiation group; limit for: nsurf on RADG.2 msid(iass) on RADG.9
la	number of surfaces for radiation group 1 on tape10
lb	number of surfaces for radiation group 2 on tape10
llc	number of surfaces for radiation group 3 on tape10
ld	number of surfaces for radiation group 4 on tape10
le	number of surfaces for radiation group 5 on tape10
lf	number of surfaces for radiation group 6 on tape10

Parameters to run Cycle 2 test cases --

parameter(ma=13,mc=700,mx=21,mg=1400,mpp=4000)  
parameter(mi=12,mj=1,mk=8,ml=20,lmm=10)  
parameter(mn=9,mp=20,mq=57,mr=1300)  
parameter(ms=15,mt=2,mv=1,mw=300,mh=11)  
parameter(my=2,nd=16,no=5,nv=5)  
parameter(lnz=80,id=6,ie=40,lif=20)  
parameter(lip=9,iq=20,lir=10,lit=7,iu=22)  
parameter(liw=6,lix=22,liz=40,ih=6)  
parameter(la=264,lb=113,llc=1,ld=1,le=1,lf=1)

The auxiliary program RADGEN, used to generate the grey body view factors for radiative heat transfer in rod arrays, also contains variable arrays dimensioned by means of parameter statements. RADGEN requires fewer parameters than COBRA-SFS, since it is concerned only with input related to the rod arrays. Table 8.2 lists the parameters and their definitions.

**Table 8.2. Dimension Parameters for RADGEN Code**

Parameter	Definition
nr	<p>maximum number of rows or columns of rods;</p> <p>a) for square arrays, the number of rows and columns is the same</p> <p>b) for rectangular arrays (rods on a square pitch, but with N rows and M columns, where <math>N \neq M</math>), nr must be greater than or equal to <math>\max(N,M)</math></p> <p>c) for triangular arrays, nr must be greater than or equal to the number of rows, since the array does not form well-defined columns</p>
np	maximum number of rods in an enclosure
nw	<p>maximum number of number computational wall surface segments making up an enclosure;</p> <p>a) for an enclosure containing rods, <math>nw = 4 * (nrows + ncol)</math></p> <p>b) for an enclosure without rods, the number of surfaces is specified by user input (on input line RADGEN.6 for batch mode, or Q12 for interactive mode).</p>
nd	<p>number of computational surface segments on each rod;</p> <p>nd=4 for square pitch</p> <p>nd=6 for triangular pitch</p>
nl	maximum number of lumped surfaces
nx	<p>maximum number of surfaces each surface sees;</p> <p>must be at least 18 for rod arrays (either square or triangular pitch)</p> <p>for enclosures without rods, nx must be equal to or greater than nw</p>
nf	maximum number of lumped surfaces that a given surface from the original view factor set can belong to

The default values of the dimension parameters in Cycle 2 of RADGEN are

parameter(nr=50,np=900,nw=500,nd=6,nl=220,nx=500,nf=1)

These values define arrays large enough to create the view factors for the problems in the COBRA-SFS Cycle 2 test case set, and therefore represent a rather large dimension set. The user should select values for the parameters that are consistent with the dimensions of the problem to be solved.

There are numerous dimension checks in Cycle 2 of the RADGEN code. If the input results in any of these dimensions being exceeded, an error message is written to the output file and execution is terminated.

## 8.4 Platform-Dependent Coding

The COBRA-SFS code is written in standard Fortran-77, and conforms to ASCII standards. It should be possible to successfully compile the code on any Fortran-77 compiler. However, the code has the feature of writing the time and date to the output file during execution, and this information is available only through system calls. Also, the cpu time is presented in the output, to provide the user with information on the execution speed of the code. Like the clock time and date, the cpu time is usually available only through a system call. Therefore, the date, clock time, and cpu time features are, by definition, platform-dependent. In addition, the procedure used in subroutines CLEAR, RESTRT, and DUMPIT to locate the addresses of the variables at the start and end of each common block uses a system routine usually called the "loc" function.

The COBRA-SFS code was originally developed on CDC and CRAY mainframe computers, using COS and uniCOS operating systems. Since the advent of powerful desktop UNIX workstations, it has been installed on the Sun SPARCstation, HP9000 workstation, the IBM RS6000 (risc) workstation, and the DEC workstation. It has also been installed on a number of DOS workstations; including the IBM ps2 workstation, and workstations with INTEL 486 and Pentium processors. It has also been installed on the APPLE MAC-II/MPW workstation.

To prevent a confusing multiplication of code versions, all platform versions are maintained in a single source. Platform-specific lines of coding are flagged with appropriate identifiers, and commented in or commented out of the compilable source at the user's discretion. Subroutine DATTIM and Function LOCA contain the platform dependent coding for all platforms in the Cycle 2 source coding. The following is a source listing of DATTIM and LOCA, showing the flagged program lines. The coding for a given platform is activated by removing the comment lines from the appropriate lines.

The platform flags used to identify the different sets of platform-specific coding are defined as follows:

<u>Platform</u>	<u>Identification Code</u>
IBM/ps2	ps2
INTELx86, (x > 2)	INT
IBM/unix	aix
SUN SPARCstation	Sun
SUN4 and DEC stations	DEC
CRAY with uniCOS system	uCS
HP-9000 workstation	HP9
APPLE MAC-II/MPW with AbSoft Compiler	MAC

An example of a platform-specific line for IBM/unix (aix) workstations is:

```
caix      integer ibuf(4)                                aix
123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
```

When the line is not active, column 1 contains a "c" and columns 2-4 contain the platform identification code. Columns 5-76 contain the Fortran coding for the line, and columns 77-79 repeat the platform identification code. To activate a line for a given platform, the first four columns are deleted.

For the above example, the line would appear in the aix version of the code as

```
integer ibuf(4)                                aix
12345678901234567890123456789012345678901234567890123456789012345678901234567890
```

The master source of the COBRA-SFS code is maintained under a configuration management utility call WITNESS. WITNESS is similar to the UPDATE and HISTORIAN utilities on CDC and IBM mainframes. The above example is from the WITNESS source of COBRA-SFS. The WITNESS source is a noncompilable source that is used to create the program library for configuration management. The program library is used to create the compilable source. In the process, any information beyond column 72 on a line of coding is lost in the compile file, so the user must rely on the source listing to identify all lines of platform dependent coding.

The following is a listing of the subroutine DATTIM. Note that this listing is for the Sun SPARCstation version of the code, and all platform dependent coding for other versions is commented out. The coding for all platforms appears in the source file.

```
subroutine dattim(iopt,alph,xivar)
c
c *****
c
c iopt = 1 -- find elapsed cpu time
c iopt = 2 -- find clock time
c iopt = 3 -- find current date
c
c *****
c
c this subroutine calls various system routines to determine
c the time and date of the run, and the cpu time used during
c code execution.
c
caixc*****
caixc the next eight lines are for the IBM UNIX (aix) workstations
caixc
caixc implicit double precision (a-h,o-z)
caixc integer ibuf(4)
caixc dimension idays(12)
caixc
caixc character*8 alph
caixc real tarray(2),xivar
caixc character*4 chx3
caixc
```

caix	integer iarray(3),iopt	aix
caixc		aix
caix	data idays/31,28,31,30,31,30,31,31,30,31,30,31/	aix
caixc		aix
caixc	*****	aix
cps2c	*****	ps2
cps2c	the next eight lines are for the IBM ps2 operating system	ps2
cps2c		ps2
cps2	character*8 alph	ps2
cps2	real tarray(2),xivar	ps2
cps2	character*4 chx3	ps2
cps2	integer iarray(3),iopt	ps2
cps2c		ps2
cps2	integer*2 ihr,imin,isec,i100th	ps2
cps2	integer*2 iyr,iday,imon	ps2
cps2	character*4 hour,minute,second	ps2
cps2	character*4 year,day,month	ps2
cps2c		ps2
cps2c	*****	ps2
cINTc	*****	INT
cINTc	the next six lines are for INTELx86 platforms (x > 2)	INT
cINTc	NOTE: this coding has been tested only on a Pentium processor	INT
cINTc		INT
cINT	character*8 alph	INT
cINT	real tarray(2),xivar	INT
cINT	character*4 chx3	INT
cINT	integer iarray(3),iopt	INT
cINTc		INT
cINT	integer*2 ihr,imin,isec,i100th	INT
cINT	integer*2 iyr,iday,imon	INT
cINTc		INT
cINTc	*****	INT
cHP9c	*****	HP9
cHP9c	the next three lines are for the HP9000 workstation	HP9
cHP9c		HP9
cHP9	character*8 alph	HP9
cHP9	real xivar	HP9
cHP9	integer iopt	HP9
cHP9c		HP9
cHP9c	*****	HP9
c	*****	Sun
c	the next four lines are for the SUN SPARCstation II	Sun
c		Sun
	character*8 alph	Sun
	real tarray(2),xivar	Sun
	character*4 chx3	Sun
	integer iarray(3),iopt	Sun
c		Sun
c	*****	Sun
cDECc	*****	DEC
cDECc	the next four lines are for the DEC workstation	DEC
cDECc		DEC
cDEC	character*8 alph	DEC
cDEC	real tarray(2),xivar	DEC
cDEC	character*4 chx3	DEC

cDEC	integer iarray(3),iopt	DEC
cDECc		DEC
cDECc	*****	DEC
cuCSc	*****	uCS
cuCSc	the next four lines are for the CRAY with uniCOS operating system	uCS
cuCSc		uCS
cuCS	character*8 alph	uCS
cuCS	real tarray(2),xivar	uCS
cuCS	character*4 chx3	uCS
cuCS	integer iarray(3),iopt	uCS
cuCSc		uCS
cuCSc	*****	uCS
cCOsc	*****	COS
cCOsc	the next four lines are for the CRAY with COS operating system	COS
cCOsc		COS
cCOS	character*8 alph	COS
cCOS	real tarray(2),xivar	COS
cCOS	character*4 chx3	COS
cCOS	integer iarray(3),iopt	COS
cCOsc		COS
cCOsc	*****	COS
c	*****	Sun
c	start of date/time logic for the SUN SPARCstation II	Sun
c		Sun
	if(iopt .eq. 1) then	Sun
	xivar = etime(tarray)	Sun
	endif	Sun
c		Sun
	if(iopt.eq.2) then	Sun
	call itime(iarray)	Sun
	write (alph(1:2),'(i2)') iarray(1)	Sun
	write (alph(4:5),'(i2)') iarray(2)	Sun
	write (chx3,'(i4)') iarray(3)	Sun
	alph(7:8) = chx3(3:4)	Sun
	alph(3:3) = ':'	Sun
	alph(6:6) = ':'	Sun
	endif	Sun
c		Sun
	if(iopt.eq.3) then	Sun
	call idate(iarray)	Sun
	write (alph(4:5),'(i2)') iarray(1)	Sun
	write (alph(1:2),'(i2)') iarray(2)	Sun
	write (chx3,'(i4)') iarray(3)	Sun
	alph(7:8) = chx3(3:4)	Sun
	alph(3:3) = '/'	Sun
	alph(6:6) = '/'	Sun
	endif	Sun
c		Sun
	return	Sun
c		Sun
c	end of date/time logic for the SUN SPARCstation II	Sun
c	*****	Sun
caixc	*****	aix
caixc	start of date/time logic for the IBM UNIX (aix) workstation	aix
caixc		aix





```

caix      nleap = 1 + nyear/4
caix      wall = iwall - 86400*(nyear*365 + nleap)
caix      nyear = nyear+70
caix      if(mod(nyear,4).eq.0) idays(2) = 29
caixc
caix      nday = wall/86400.0
caix      wall = wall - 86400.0*nday
caix      nday = nday+1
caix      do 1 m=1,12
caix          imon = m
caix          nday = nday - idays(imon)
caix          if(nday.lt.0) go to 2
caix 1      continue
caix 2      continue
caix      nday = nday + idays(imon) + 1
caix      write (alph,3) imon,nday,nyear
caix 3      format (i2,'/',i2,'/',i2)
caix      endif
caixc
caix      return
caixc
caixc
caixc end of date/time logic for the IBM UNIX (aix) workstation
caixc*****
cps2c*****
cps2c start of date/time logic for the IBM ps2 operating system
cps2c
cps2      if(iopt .eq. 1) then
cps2          call gettim(ihr,imin,isec,i100th)
cps2          xivar = 3600*ihr + 60*imin + isec + 0.01*i100th
cps2      endif
cps2c
cps2      if(iopt .eq. 2) then
cps2          call gettim(ihr,imin,isec,i100th)
cps2          ihr4 = ihr
cps2          imin4 = imin
cps2          isec4 = isec
cps2          call intoch(ihr4,hour,lenstr)
cps2          call intoch(imin4,minute,lenstr)
cps2          call intoch(isec4,second,lenstr)
cps2          if (minute.eq.' ') minute(3:4) = '00'
cps2          alph = hour(3:4)//'-'//minute(3:4)//'-'//second(3:4)
cps2      endif
cps2c
cps2      if(iopt .eq. 3) then
cps2          call getdat(iyr,imon,iday)
cps2          iyr4 = iyr
cps2          iday4 = iday
cps2          imon4 = imon
cps2          call intoch(iyr4,year,lenstr)
cps2          call intoch(iday4,day,lenstr)
cps2          call intoch(imon4,month,lenstr)
cps2          alph = day(3:4)//'-'//month(3:4)//'-'//year(3:4)
cps2      endif
cps2c

```



```

cps2     else if (itest(j).eq.4) then
cps2         string(i:i) = '4'
cps2     else if (itest(j).eq.5) then
cps2         string(i:i) = '5'
cps2     else if (itest(j).eq.6) then
cps2         string(i:i) = '6'
cps2     else if (itest(j).eq.7) then
cps2         string(i:i) = '7'
cps2     else if (itest(j).eq.8) then
cps2         string(i:i) = '8'
cps2     else if (itest(j).eq.9) then
cps2         string(i:i) = '9'
cps2     endif
cps2     j = j-1
cps2 100 continue
cps2     do 110 I=1,4
cps2         if (string(i:i).ne.' ') go to 120
cps2 110 continue
cps2 120 continue
cps2     lenstr = 5-i
cps2c
cps2 900 format(1x,'error...integer value less than zero in routine ',
cps2     *'intoch')
cps2 910 format(1x,'error...integer value greater than 999 in intoch')
cps2c
cps2     return
cps2c
cps2c end of date/time logic for the IBM ps2 operating system
cps2c*****
cINTc*****
cINTc start of date/time logic for INTELx86 platform (x > 2)
cINTc NOTE: this coding has been tested only on a Pentium processor
cINTc
cINT     if(iopt .eq. 1) then
cINT         call gettim(ihr,imin,isec,i100th)
cINT         xivar = 3600*ihr + 60*imin + isec + 0.01*i100th
cINT     endif
cINTc
cINT     if(iopt .eq. 2) then
cINT         call gettim(ihr,imin,isec,i100th)
cINT         write (alph(1:2),'(i2.2)') ihr
cINT         write (alph(4:5),'(i2.2)') imin
cINT         write (alph(7:8),'(i2.2)') isec
cINT         alph(3:3) = ':'
cINT         alph(6:6) = ':'
cINT     endif
cINTc
cINT     if(iopt .eq. 3) then
cINT         call getdat(iyr,imon,iday)
cINT         write (alph(1:2),'(i2.2)') imon
cINT         write (alph(4:5),'(i2.2)') iday
cINT         write (chx3,'(i4.4)') iyr
cINT         alph(7:8) = chx3(3:4)
cINT         alph(3:3) = '/'
cINT         alph(6:6) = '/'

```



```

cCOSc end of date/time logic for the CRAY-COS operating system          COS
cCOSc*****                                                                COS
cuCSc*****                                                                uCS
cuCSc start of date/time logic for the CRAY-uniCOS operating system    uCS
cuCSc                                                                uCS
cuCS      if(iopt .eq. 1) then                                          uCS
cuCS          call second(xivar)                                       uCS
cuCS      endif                                                         uCS
cuCSc                                                                uCS
cuCS      if(iopt .eq. 2) then                                          uCS
cuCS          call clock(alph)                                         uCS
cuCS      endif                                                         uCS
cuCSc                                                                uCS
cuCS      if(iopt .eq. 3) then                                          uCS
cuCS          call date(alph)                                          uCS
cuCS      endif                                                         uCS
cuCSc                                                                uCS
cuCS      return                                                         uCS
cuCSc                                                                uCS
cuCSc end of date/time logic for the CRAY-uniCOS operating system    uCS
cuCSc*****                                                                uCS
cHP9c*****                                                                HP9
cHP9c start of date/time logic for the HP-9000 operating system        HP9
cHP9c                                                                HP9
cHP9      if(iopt .eq. 1) then                                          HP9
cHP9          call second(xivar)                                       HP9
cHP9      endif                                                         HP9
cHP9c                                                                HP9
cHP9      if(iopt .eq. 2) then                                          HP9
cHP9          call time(alph)                                          HP9
cHP9      endif                                                         HP9
cHP9c                                                                HP9
cHP9      if(iopt .eq. 3) then                                          HP9
cHP9          call idate(mon,iday,iyr)                                  HP9
cHP9          write(alph,2) mon,iday,iyr                              HP9
cHP9 2      format(i2,'/',i2,'/',i2)                                    HP9
cHP9      endif                                                         HP9
cHP9c                                                                HP9
cHP9      return                                                         HP9
cHP9c                                                                HP9
cHP9c end of date/time logic for the HP-9000 operating system          HP9
cHP9c*****                                                                HP9
cMACc*****                                                                MAC
cMACc start of date/time logic for the Apple/Macintosh operating system MAC
cMACc**                                                                MAC
cMACc**          APPLE MAC-II/MPW with AbSoft Compiler                MAC
cMACc**          Quadra and PPC compatible                            MAC
cMACc**          Note: MPW include file OSUtils.inc required         MAC
cMACc**                                                                MAC
cMACc define a datetime record, structure and comments taken          MAC
cMACc from Inside MacIntosh, Vol 2, page 378                          MAC
cMACc                                                                MAC
cMAC      IMPLICIT NONE                                                MAC
cMAC      INCLUDE "OSUtils.inc"                                        MAC
cMAC      character *4 tempvar                                         MAC

```

```

cMAC      character *8 alph          MAC
cMAC      integer  *4 iopt          MAC
cMAC      real     *4 xivar         MAC
cMAC      RECORD /DateRec/ DateTime MAC
cMAC
cMAC
cMAC      xivar      = 0.0          MAC
cMAC      alph(1:8) = '12345678'   MAC
cMAC      tempvar   = ' '          MAC
cMAC
cMAC      go to (100,200,300),iopt  MAC
cMAC 300 continue                   MAC
cMAC      CALL GetTime(DateTime)   MAC
cMAC      write(alph(1:2),1000) DateTime.month MAC
cMAC      write(alph(4:5),1000) DateTime.day   MAC
cMAC      write(tempvar,1002) DateTime.year   MAC
cMAC      alph(7:8) = tempvar(3:4)           MAC
cMAC      alph(3:3) = '/'                   MAC
cMAC      alph(6:6) = '/'                   MAC
cMAC      if (alph(4:4).eq.' ') alph(4:4) = '0' MAC
cMAC      go to 999                       MAC
cMAC 200 continue                       MAC
cMAC      CALL GetTime(DateTime)   MAC
cMAC      write(alph(1:2),1000) DateTime.hour   MAC
cMAC      write(alph(4:5),1000) DateTime.minute MAC
cMAC      write(alph(7:8),1000) DateTime.second MAC
cMAC      if (alph(4:4).eq.' ') alph(4:4) = '0' MAC
cMAC      if (alph(7:7).eq.' ') alph(7:7) = '0' MAC
cMAC      alph(3:3) = ':'                   MAC
cMAC      alph(6:6) = ':'                   MAC
cMAC      go to 999                       MAC
cMAC 100 continue                       MAC
cMAC      CALL GetTime(DateTime)   MAC
cMAC      xivar = 3600.0 * DateTime.hour   MAC
cMAC      xivar = xivar + (60.0 * DateTime.minute) MAC
cMAC      xivar = xivar + DateTime.second  MAC
cMAC      go to 999                       MAC
cMAC 1000 format(i2)                    MAC
cMAC 1002 format(i4)                    MAC
cMAC**
cMAC**
cMAC*****
cMAC 999 return                          MAC
cMAC*****
cMAC** End of MPW Absoft Compiler Timing call MAC
cMAC*****
cMACc end of date/time logic for the Apple/Macintosh operating system MAC
cMACc*****
end

```

The following is a listing of the function LOCA, showing all platform dependent coding for the *loc* function used in determining the locations of the beginning and ending variables in the common blocks for subroutines CLEAR, DUMPIT, and RESTRT. This listing is from the Sun SPARCstation version, with the coding for all other platforms commented out.

```

function loca(ae, ab)
c
c   save
c
c
c *****
c
c   this function returns the number of
c   "words" between ae and ab.
c
c   this is needed by subroutines clear and
c   dumpit.
c
c   arguments -
c   ae   ending variable in common
c   ab   beginning variable in common
c
c   loca is called by clear and dumpit
c
c   loca calls system functions %loc, locf or loc
c
c *****
c
c   dimension fl(2)
c
c   vax (i.e.; 32-bit machine)
c   loca=(%loc(ae) - %loc(ab)) / 4
c
c   ibm
c   loca=( loc(ae)- loc(ab))/(loc(f1(2))-loc(f1(1)))
c
c   cdc
c   loca=(loc(ae) - loc(ab))
c
c *****
c loca function for SUN SPARCstation II
c
c   loca=(loc(ae) - loc(ab)) / 4 + 1
c
c *****
cDECc *****
cDECc loca function for SUN4 and DEC workstations
cDECc
cDEC   loca=(loc(ae) - loc(ab)) / 4 + 1
cDECc
cDECc *****
cHP9c *****
cHP9c loca function for HP-9000 workstation

```

Sun  
Sun  
Sun  
Sun  
Sun  
Sun  
DEC  
DEC  
DEC  
DEC  
DEC  
DEC  
HP9  
HP9



```

cHP9c
cHP9      loca=(loc(ae) - loc(ab)) / 4 + 1
cHP9c
cHP9c*****
cMACc*****
cMACc loca function for Apple-Macintosh workstation
cMACc
cMAC      loca=(loc(ae) - loc(ab)) / 4 + 1
cMACc
cMACc*****
cINTc*****
cINTc loca function for INTELx86 platforms (x > 2)
cINTc
cINT      dimension f1(2)
cINTc
cINT      loca = (loc(ae) - loc(ab)) / (loc(f1(2)) - loc(f1(1)))
cINTc
cINTc*****
cps2c*****
cps2c loca function for IBM/ps2 workstation
cps2c
cps2      loca=(loc(ae) - loc(ab)) / (loc(f1(2)) - loc(f1(1)))
cps2c
cps2c*****
caixc*****
caixc loca function for IBM/unix workstation
caixc
caix      dimension f1(2)
caixc
caix      loca=(loc(ae) - loc(ab)) / (loc(f1(2)) - loc(f1(1)))
caixc
caixc*****
cCOSc*****
cCOSc loca function for CRAY with COS operating system
cCOSc
cCOS      loca=(loc(ae) - loc(ab))
cCOSc
cCOSc*****
cuCSc*****
cuCSc loca function for CRAY with uniCOS operating system
cuCSc
cuCS      loca=(loc(ae) - loc(ab))
cuCSc
cuCSc*****
c
      return
      end

```

```

HP9
HP9
HP9
HP9
MAC
MAC
MAC
MAC
MAC
MAC
INT
INT
INT
INT
INT
INT
INT
ps2
ps2
ps2
ps2
ps2
ps2
aix
aix
aix
aix
aix
aix
COS
COS
COS
COS
COS
uCS
uCS
uCS
uCS
uCS

```

## 8.5 Installation Guide

Versions of COBRA-SFS, Cycle 2, have been developed or validated on eight different platforms, including the following seven workstations: the SUN SPARCstation 2; the IBM RS6000; the Hewlett-Packard HP9000; the DEC workstation; pc's with INTEL-386, -486, and Pentium processors; IBM/ps2, and the APPLE MAC-II/MPW. All eight versions of the code are essentially identical. The only differences are in the logic for the date, clock time, and cpu time subroutine DATTIM, and the LOCA function, which is used to find the beginning and ending addresses in memory for the common blocks (see Section 8.4 for a complete listing of platform dependent coding). The code transmittal package for COBRA-SFS, Cycle 2, as described in Section 8.5.1 and Section 8.4.2, gives step-by-step installation instructions.

### 8.5.1 Code Transmittal Package

The master version of the COBRA-SFS code is maintained under a configuration management utility called WITNESS. WITNESS is similar to the UPDATE and HISTORIAN utilities on CDC and IBM mainframes, and the master source files for both COBRA-SFS and RADGEN are ASCII text files in WITNESS/UPDATE format. These files are not compilable. The WITNESS source file is used to create a WITNESS program library, which assigns line numbers and identifiers to each line of coding. This program library cannot be edited, but it can be used to create (with or without modifications) compilable files, new source files, and list files, by running the WITNESS program with appropriate input.

The WITNESS source files for both COBRA-SFS and RADGEN are included in the code transmittal package. This is done mainly to give the user a source that can be converted fairly easily to an alternative configuration management utility. For users who do not have such a utility, or do not choose to use one for this code, the transmittal package also includes compilable files for each platform.

These compilable files contain the appropriate platform-specific coding, and each has been tested on the workstation for which it is designed. In most cases, a version should compile on its particular platform without errors. However, differences in operating systems and fortran compiler versions may necessitate some minor changes in a particular installation.

There is only one compilable source for RADGEN. Since it has no platform dependent coding, it can be expected to compile successfully on any platform that has adequate memory and a Fortran-77 compiler.

Input files for two test cases are included in the transmittal package, for reference in verifying code installation on any platform. The Mitsubishi test case is a single-assembly model (refer to Section 7.2.2 for a complete description), and the TN24 test case (see Section 7.3.1) is a full cask model, including both upper and lower plena.

The output files for the mitsubishi and TN24 test cases are from the Sun SPARCstation version. All other platforms gave essentially the same results in the validation testing of Cycle 2. Any significant differences between the results obtained on another platform and the Sun output files should be investigated carefully and corrected before using the code in any of the applications.

Three test cases are included for verification of the installation of RADGEN. These are the same cases as the sample problems presented in Section 6.3. As with the COBRA-SFS output files, the output files for the RADGEN test cases are from the Sun SPARCstation II. All other platforms should give the same results. Table 8.3 lists the COBRA-SFS code package files along with each of their descriptions.

**Table 8.3. COBRA-SFS Code Package**

File Name	Description
sfs2_Sun.src	UPDATE/WITNESS source of COBRA-SFS, cycle 2 (Sun version)
sfs2_XXX.f	compilable sources for all 8 platforms; XXX stands for platform identifier code
second.c	subroutine in C programming language for time calculation on the HP9000 version; object must be linked at compile time (see Section 8.5.2)
mitsubishi_cycle2.inp	input file for single-assembly Mitsubishi test case
mitsubishi.tape10	tape10 file created using RADGEN for Mitsubishi test case
mitsubishi_cycle2.out	output file for single-assembly Mitsubishi test case (Sun version)
tn24_vertical.inp	input file for multi-assembly TN24P cask test case
tn24v.tape10	tape10 file created using RADGEN for TN24P vertical cask model
tn24_vertical.out	output file for multi-assembly TN24P cask test case
radgen_cycle2.src	UPDATE/WITNESS source of RADGEN, cycle 2
radgen_cycle2.f	compilable source (suitable for all platforms)
testcase1.inp	input for RADGEN test case 1: 3 x 3 square array in a square canister
testcase1.out	output for RADGEN test case 1: 3 x 3 square array in a square canister
testcase2.inp	input for RADGEN test case 2: triangular array in a square canister
testcase2.out	output for RADGEN test case 2: triangular array in a square canister
testcase3.inp	input for RADGEN test case 3: hexagonal array in a hexagonal canister
testcase3.out	input for RADGEN test case 3: hexagonal array in a hexagonal canister

## 8.5.2 Installation Instructions

The first step in the installation of COBRA-SFS should be to convert a copy of the WITNESS source files of COBRA-SFS and RADGEN to a form acceptable to the code configuration management (CCM) utility on the user's computer. The second step would then be to create a compilable file for each code using the CCM utility, with the appropriate platform coding activated and all coding for other platforms deactivated. However, if the user's platform does not have a CCM utility, or there is some reason for not bringing COBRA-SFS under its aegis, then the first step in the installation of the code is to copy the appropriate compilable version from the transmittal package, and run it through the Fortran-77 compiler.

The transmittal package includes compilable files for eight different platforms, identified as follows:

<u>Platform</u>	<u>Compilable file name</u>
IBM/ps2	sfs2_ps2.f
INTELx86, (x > 2)	sfs2_INT.f
IBM/RS6000 (aix)	sfs2_aix.f
SUN SPARCstation	sfs2_Sun.f
SUN4 and DEC stations	sfs2_DEC.f
CRAY with uniCOS system	sfs2_uniCOS.f
HP-9000 workstation	sfs2_HP9.f
APPLE MAC-II/MPW with AbSoft Compiler	sfs2_MAC.f

The compilable file for the RADGEN code is radgen\_cycle2.f, and is suitable for all platforms with standard Fortran-77 compilers.

Compilation of the code can be accomplished on most platforms with the single command:

```
f77 sfs2_XXX.f -o sfs2_XXX.x
```

Note that XXX corresponds to the platform identifier mnemonic shown in the file name list above. Similarly, the command to compile the RADGEN code is

```
f77 radgen_cycle2.f -o radgen_cycle2.x
```

Mainly for convenience of transmittal, the entire code is contained in a single compilable file. It is also possible to split the compilable file into separate files for each subroutine, using the 'fsplit' utility available on most unix workstations. With the code split into separate subroutines, a script file can be used to compile the subroutines individually, then create an object library and executable. An example of such a script file for the Sun SPARCstation is as follows:

```
#  
# The following makefile is to generate the following files:  
#  
# 1) cobrasfs2.a  a library containing all *.o files  
#  
# 2) cobrasfs2.x  the master version of the executable  
#  
# This makefile assumes that "witness" has already  
# created the *.f files  
#
```

```

# NOTE: If a new routine is added the list of routines
#       under OBJS must be modified accordingly
#
# This makefile is for use on the master directory only
#
# Author: Annette Koontz
#       (plagarized by J. M. Cuta, 7/12/89)
#
#
.SUFFIXES: .o .f
#
OBJS =   cobra.o \
        area.o clear.o coloc.o curve.o dattim.o differ.o \
        dumpit.o echo.o energy.o exprop.o force.o group4.o \
        hotrod.o limits.o load.o loca.o mix.o moment.o \
        network.o oned.o post.o prefix.o prop.o recirc.o \
        reheat.o restrt.o result.o setin.o setout.o setup.o \
        solver.o varprp.o
#
cobrasfs2.x: cobrasfs2.a
#
# load step
#
        ar x cobrasfs2.a cobra.o
        f77 cobra.o cobrasfs2.a -o cobrasfs2.x
#
cobrasfs2.a: $(OBJS)
#
# build the library
#
        rm -f cobrasfs2.a
        ar rv cobrasfs2.a $(OBJS)
        ranlib cobrasfs2.a
.f.o:
#
# Compile
#
        f77 -c $*.f
#

```

For the validation assessments for Cycle 2, the codes was compiled without optimization, as indicated in the sample command lines given above. Optimization can significantly decrease execution time, however, and might be advisable for users who expect to run a large number of cases, or very large problems. Some care must be exercised, however, to make certain that code optimization does not affect code results. Output from the optimized version should be verified against the output from an unoptimized executable. If the optimized version does not give the same results as the unoptimized version, it is probably not a good idea to optimize the code with that particular compiler.

Platform-specific coding for the HP9000 version calls a subroutine named "second" to obtain the cpu time. This subroutine is in the C programming language, and so cannot easily be included in the COBRA-SFS Fortran source file. This subroutine is included in the transmittal package as the file "second.c." It is a very simple routine, consisting of the following lines:

```
#include <sys/times.h>
second (s)
float *s;
{
  struct tms tbuf;
  times (&tbuf);
  *s = 0.01 * (tbuf.tms_utime + tbuf.tms_stime);
  return;
}
```

Compilation of the HP9000 version of COBRA-SFS requires compiling "second.c" with the C compiler, then linking the resulting object file, "second.o", in the compilation and load step for the COBRA-SFS compile file. This can be done with the following command lines:

```
cc +O3 -c second.c
f77 +e sfs2_HP9.f second.o -o sfs2_HP9.x
```

The final step in the installation of the code is to run the test cases for COBRA-SFS and RADGEN. The input files for two COBRA-SFS and three RADGEN test cases are included in the transmittal package. Note that the input files are the same for all platforms; there are no platform-specific changes to the code input stream.

To verify the installation, the executable version of COBRA-SFS created in the compilation step above (sfs2\_XXX.x) is run with the test case input file (mitsubishi\_cycle2.inp or tn24\_vertical.inp) copied to a file named "input". The view factor input for each case (mitsubishi.tape10 or tn24v.tape10), must be copied to the file named "tape10." The code will produce an output file of the results, in the local file "output." The transmittal tape includes output files produced on the SUN SPARCstation (mitsubishi\_cycle2\_Sun.out and tn24\_vertical\_Sun.out). The results obtained with the users installation of COBRA-SFS should be compared with these output files from the transmittal tape. There should be no significant differences.

Similarly, the executable version of RADGEN (radgen\_cycle2.x) should be run with the input file (testcase1.inp, testcase2.inp, or testcase3.inp) assigned to the local file name "input." If RADGEN has been compiled for interactive mode (i.e., ibatch = 0 in the main program), the test cases are run by responding to Q1 with "n," to decline interactive input (see Section 6.3.2.1). The RADGEN results are written to file "tape10," and should be compared with the appropriate output file in the transmittal package (testcase1.tape10, testcase2.tape10, or testcase3.tape10). There should be no significant differences between these files and the results obtained on the new platform.

### 8.5.3 Redimensioning COBRA-SFS and RADGEN

The dimensions of arrays in both COBRA-SFS and RADGEN are defined using parameter statements, as described in Section 8.3. The values specified for the dimension parameters in the master version of the code are large enough to run all the validation test cases reported in Section 7.0. These parameters will probably be sufficient for most applications of the code. However, if the user needs a larger version, it will be necessary to change the parameter statements in the code.

This will be much easier if the code is under some sort of configuration management utility. In the WITNESS source file, the dimension parameters are included in a COMDECK called SPECS that appears in nearly every subroutine in the code. It is possible to change the parameter statements by editing the compilable file (sfs2\_XXX.f), but the changes must be propagated without error through every occurrence of the parameter statements in the code. A CCM utility will generally make this simple. If it has to be done by editing the compilable file, however, the chance for error can approach certainty.

With the RADGEN code, the problem is somewhat less onerous since there are only five subroutines in the code, in addition to the main program. However, the same caveat regarding the necessity for consistent changes applies. A parameter statement changed in one subroutine must be altered in exactly the same manner in all other subroutines.

Once a new compilable file has been created, either by invoking a CCM utility or by editing a copy of the original compilable file, the code can be compiled as described above to create a new executable. This executable will have the redimensioned arrays, and will be able to run the new cases.

## 9.0 References

- Bates, J. M. 1986. *Single PWR Spent Fuel Assembly Heat Transfer Data for Computer Code Evaluations*. PNL-5571, Pacific Northwest Laboratory, Richland, Washington.
- Cox, R. L. 1977. *Radiation Heat Transfer in Arrays of Parallel Cylinders*. ORNL-5239, Oak Ridge National Laboratory, Oak Ridge, Tennessee.
- Creer, J. M., T. E. Michener, M. A. McKinnon, J. E. Tanner, E. R. Gilbert, and R. L. Goodman. 1987. *The TN-24P PWR Spent-Fuel Storage Cask: Testing and Analyses*. EPRI-NP-5128/PNL-6054, Electric Power Research Institute, Palo Alto, California.
- Croff, A. G. 1980. *ORIGEN-2 -- A Revised and Updated Version of the Oak Ridge Isotope Generation and Depletion Code*. ORNL-5621, Oak Ridge National Laboratory, Oak Ridge, Tennessee.
- Cuta, J. M., D. R. Rector, and J. M. Creer. 1984. *Comparisons of COBRA-SFS Calculations with Data from Simulated Sections of Unconsolidated and Consolidated BWR Spent Fuel*. EPRI-NP-3764, Electric Power Research Institute, Palo Alto, California.
- Cuta, J. M., and J. M. Creer. 1986. *Comparisons of COBRA-SFS Calculations to Data from Electrically Heated Test Sections Simulating Unconsolidated and Consolidated BWR Spent Fuel*. EPRI-NP-4593, Electric Power Research Institute, Palo Alto, California.
- Fry, C. J., E. Livesey, and G. T. Spiller. 1983. "Heat Transfer in a Dry, Horizontal LWR Spent Fuel Assembly." In *Proceedings of Seventh International Symposium*, New Orleans, Louisiana. Oak Ridge National Laboratory, Oak Ridge, Tennessee. From Packaging and Transportation of Radioactive Materials (PATRAM'83) Symposium.
- George, T. L., K. L. Basehore, C. L. Wheeler, W. A. Prather, and R. E. Masterson. 1980. *COBRA-WC: A version of COBRA for Single-phase Multi-assembly Thermal-Hydraulic Transient Analysis*. PNL-3259, Pacific Northwest Laboratory, Richland, Washington.
- Gosman, A. D., R. Herbert, S. V. Patankor, R. Potter, and D. B. Spalding. 1973. "The SABRE Code for Prediction of Coolant Flows and Temperatures in Pin Assemblies Containing Blockages." Presented at International Meeting on Reactor Heat Transfer, Karlsruhe, Germany. HTS/73/47, Imperial College of Science and Technology.
- Hirt, C. W., and J. L. Cook. 1972. "Calculating Three-Dimensional Flows Around Structures and Over Rough Terrain." *Journal of Computational Physics*, 10:324-340.
- Hottel, H. C., and A. F. Sarofin. 1967. *Radiative Heat Transfer*. McGraw-Hill, Inc., New York.



Ingesson, L. and S. Hedberg. 1970. "Heat Transfer Between Subchannels in a Rod Assembly." Paper No. FC7.11. Presented at the Fourth International Heat Transfer Conference, Versailles, France.

Irino, M., M. Oohashi, T. Irie, and T. Nishikawa. 1986. *Study on Surface Temperatures of Fuel Pins in Spent Fuel Dry Shipping/Storage Casks*. IAEA-SM-286/139P, pp. 585-598.

Kays, E. U., and M. E. Crawford. 1980. *Convective Heat and Mass Transfer*, McGraw-Hill, Inc., New York.

Khan, E. U., W. A. Prather, T.L. George, and J.M. Bates. 1981. *A Validation Study of the COBRA-WC Computer Program for LMFBR Thermal-Hydraulic Analysis*, PNL-4128, Pacific Northwest Laboratory, Richland, Washington.

Lindeburge, M. R. 1981 *Mechanical Engineering Review Manual*, 6th ed. The Professional Engineering Program, San Carlos, California.

Lombardo, N. J., T. E. Michener, C. L. Wheeler, and D. R. Rector. 1986. *COBRA-SFS Predictions of Single-Assembly Spent Fuel Heat Transfer Data*. PNL-5781, Pacific Northwest Laboratory, Richland, Washington.

Lombardo, N. J., J. M. Cuta, T. E. Michener, D. R. Rector, and C. L. Wheeler. 1986a. *COBRA-SFS: A Thermal-Hydraulic Analysis Computer Code; Volume III: Validation Assessments*. PNL-6049, Vol. 3, Pacific Northwest Laboratory, Richland, Washington.

McKinnon, M. A., J. M. Creer, T. E. Michener, J. E. Tanner, E. R. Gilbert, and R. L. Goodman. 1986. *TN-24P PWR Spent Fuel Storage Cask Performance Testing and Analysis*. EPRI-5128, Electric Power Research Institute, Palo Alto, California.

McKinnon, M. A., R. E. Dodge, R. C. Schmitt, L. E. Eslinger, and G. Dineen. 1992. *Performance Testing and Analyses of the VSC-17 Ventilated Concrete Cask*. EPRI-TR-100305/PNL-7839, Electric Power Research Institute, Palo Alto, California.

Rector, D. R., C. L. Wheeler, and N. J. Lombardo. 1986. *COBRA-SFS: A Thermal-Hydraulic Analysis Computer Code, Volume 1: Mathematical Models and Solution Method*. PNL-6049, Vol. 1, Pacific Northwest Laboratory, Richland, Washington.

Rector, D. R., R. A. McCann, U. P. Jenquin, C. M. Heeb, J. M. Creer, and C. L. Wheeler. 1986b. *CASTOR-1C Spent Fuel Storage Cask Decay Heat, Heat Transfer, and Shielding Analyses*. PNL-5974, Pacific Northwest Laboratory, Richland, Washington.

Rector, D. R., J. M. Cuta, and N. J. Lombardo. 1986c. *COBRA-SFS Thermal-Hydraulic Analyses of the CASTOR-1C and REA 2023 BWR Storage Casks Containing Consolidated Spent Fuel*. PNL-5802, Pacific Northwest Laboratory, Richland, Washington.

Rector, D. R. 1987. *RADGEN: A Radiation Exchange Factor Generator for Rod Bundles*. PNL-6342, Pacific Northwest Laboratory, Richland, Washington.

Rector, D. R., and T. E. Michener. 1989. *COBRA-SFS Modifications and Cask Model Optimization*. PNL-6706, Pacific Northwest Laboratory, Richland, Washington.

Rogers, J. T., and N. E. Todreas. 1968. "Coolant Mixing in Reactor Fuel Rod Assemblies--Single Phase Coolants." In *Heat Transfer in Rod Assemblies*, ASME, pp. 1-56.

Rogers, J. T., and R. G. Rosehart. 1972. *Mixing by Turbulent Interchange in Fuel Assemblies: Correlations and Inferences*, ASME Paper No. 72-HT-53.

Rowe, D. S. 1973, *COBRA-IIIC: A Digital Computer Program for Steady-State and Transient Thermal-Hydraulic Analysis of Rod Bundle Nuclear Fuel Elements*. BNWL-1695, Battelle Northwest Laboratory, Richland, Washington.

Slattery, J. C., 1972, *Momentum, Energy, and Mass Transfer in Continua*, McGraw-Hill, Inc., New York.

Stewart, C. W., C. L. Wheeler, R. J. Cena, C. A. McMonagle, J. M. Cuta, and D. S. Trent. 1977. *COBRA-IV: The Model and the Method*. BNWL-2214, Battelle Northwest Laboratory, Richland, Washington.

Tong, L. S. 1968. "Pressure Drop Performance of a Rod Assembly." In *Heat Transfer in Rod Assemblies*, ASME, pp. 57-69.

Wheeler, C. L., R. A. McCann, N. J. Lombardo, D. R. Rector, and T. E. Michener. 1986. "HYDRA and COBRA-SFS Temperature Calculations for CASTOR-1C, REA-2023, CASTOR-V/21, and TN-24P Casks." In *Proceedings, Third International Spent Fuel Storage Technology Symposium and Workshop*, Vol. 1, S77-S98, CONF-960417, National Technical Information Service, Springfield, Virginia.

Wiles, L. E., N. J. Lombardo, C. M. Heeb, U. P. Jenquin, T. E. Michener, C. L. Wheeler, J. M. Creer, and R. A. McCann. 1986. *BWR Spent Fuel Storage Cask Performance Test, Volume II: Pre- and Post-Test Decay Heat, Heat Transfer, and Shielding Analyses*. PNL-5777, Vol. 2, Pacific Northwest Laboratory, Richland, Washington.

**Appendix A**

**Common Block Variables**

## Listing of Common Blocks in COBRA-SFS, Cycle 2

```

*comdeck geom
  dimension iloc(ln8)
c
  common /geom/ gbeg1(1), ik(mg), jk(mg), length(mg), gangle(mg),
1             ikw(nd,mw), z, ftm, theta, atotal, gelev, angle(5),
2             elev, sl, dx(mx), gk(lit), ibandw(iu),
3             an(mc), dhyd(mc), ilocs(ln1,mc), perim(mc), per(mc),
4             hperim(mc), dduct(iu,lix), abar(mc), endg1
c
  common /flags/ gbeg2(1), ierror, j1, iterat, nrc, nwk,
1             naah, naahpl, nspec(20), nanglt,
2             isolve, iass, nfcon, nscbc, nramph,
3             ischeme, ibwl, ibwr, ic, iwide, iclose, iclosa(iu),
4             isym, nrow, i3post, endg2
c
  common /const/ pi, dtgc, gc, idtgc, dti, sb
c
  common /barea/ gbeg3(1), afact(ml,ma), gfact(ml,ms), axl(ml), x(mx),
1             gapxl(ml), naramp, nafact, naxl, ngxl, ngaps, nch(ma),
2             ngap(ms), idarea(mc), idgap(mg), gapn(mg), endg3
c
  equivalence (iloc(1), ilocs(1,1))
c
  real idtgc, length
c
  level 2, /geom/
c
  level 2, /flags/
*comdeck fields
  common /field1/ fbeg1(1), wtemp(mg),
1             p(mc, mx), f(mc, mx), fold(mc, mx), h(mc, mx), hold(mc, mx),
2             rho(mc, mx), rhoold(mc, mx), rhobar(mc, mx), w(mg, mx),
3             wold(mg, mx), dfdp(mc, mx), dwdp(mg, mx),
4             twall(mw, mx), twold(mw, mx), ftyp(mr, mx), flux(mr, mx),
5             hsurf(mr, mx), trod(mn, mr, mx),
6             a(mc, mx), dhyd(mc, mx), gap(mg, mx), app(mc, mx),
1             wp(mg), dfdpd(mx), dfdpo(mx), dtii(mg),
2             dtjj(mg), endf1
c
  common /field2/ fbeg2(1), hah(ie, mc), saver(mc),
1             flow(mc), qprim(mc), ichan(ln6, mc), const(lib),
2             hfilm(mc), tmh(mc), ftemp(mc), fsp(mc),
2             tinlet(mc), finlet(mc), hinlet(mc),
3             fmomf(mc), dpk(mc), dpdx(mc), relxi(mc),
4             dhdx(mc), dfdx(iu), v(mc), cons(mx),
5             nach(mc), itypin(lit, 3), firstt, endf2
c
  common /bundle/ fbeg3(1), ipoint(iu, 8), ngap1, ngap2,
1             nchan1, nchan2, nassem, nchanl,
2             mm, it, ip, ia, iw, iz, ix, ir, if, nz, nk,
3             maxtyp, ndxp1, ndx, endf3
c
  common /fields3/ conk(mg), dpbar(lip*iq)
c
  common /fieldb/ term1(mc), wpp(mg)
c
  level 2, /field1/
c
  level 2, /field2/
c
c
c
c
*comdeck props

```

```

c
common /prop1/ pbeg1(1), pliq(mp,2), temliq(mp,2), hliq(mp,2),
1 conliq(mp,2), cpliq(mp,2), visliq(mp,2), vliq(mp,2),
2 bliq(mp,2), tvary(mp), varyk(mp), varycp(mp),
3 varyr(mp), ntnode, nsett, nfprop(2), nviscw, data(mpp),
4 tmelt(2), tboil(2), nfass(iu), endp1

c
common /prop2/ pbeg2(1), pref, lamnh(liw), nheat, nflmc(iu), ah1(liw),
1 ah2(liw), ah3(liw), ah4(liw), ah11(liw), ah12(liw),
2 ah13(liw), ah14(liw), aht1(liw), aht2(liw),
3 aht3(liw), aht4(liw), aht11(liw), aht12(liw),
4 aht13(liw), aht14(liw), ahf1(liw), ahf2(liw),
5 istar(mg), ntheat, nfheat, endp2

c
common /prop3/ pbeg3(1), visc(mc,3), viscw(mc), con(mc),
1 cp(mc), vp(mc), t(mc), vs cm(mc), beta(mc),
2 ntype(mc), amix(lit), bmix(lit),
3 endp3

c
common /matprp/ pbeg4(1), uwcp(nd,mw), uwall(nd,mw),
1 areacp(mw2), matypp(mw2), tpsave(mw2),
2 npwcon, con0(id), con1(id), con2(id), con3(id), cpsol(id),
3 rhesol(id), rdima(liz), rdimb(liz),
4 ngvt(id), conw(mw), matj(id,mx), mattyp(mw),
4 imat(id), pdna(iu), pdn, nsprop, endp4

c
common /alpha/ aname(id), alph
character*8 aname, alph

c
logical lamnh

c
c level 2, /prop3/
c *comdeck drags
c
common /drag/ dbeg1(1), dpa(iu), kij(mg), cij(mg), itran,
1 aa(liw), bb(liw), cc(liw), dd(liw), ee(liw),
2 aal(liw), bbl(liw), ccl(liw), lamnf(liw),
3 aat(liw), bbt(liw), cct(liw), ddt(liw), eet(liw),
4 aatl(liw), bbt1(liw), cctl(liw),
5 nfrict, ishear, nshear, kn(mg,2), ck(mg,4), endd1

c
common /bgrid/ dbeg2(1), factor(lit,mq,mx), ilce(lit,mq),
1 lcass(iu), lcff(mk), lcasst(mk), ifact(mq,mx),
2 nlevel(lit,mq), cd(lit,mq,mx), xcd(lit,mq,mx),
3 ifb(mc), nblock, flo(iu), allf, nbl(lit),
4 flog(mj), pwin(iu), pwout(iu), rain(iu),
5 raout(iu), rgin(mj), rgout(mj), netgrp(iu),
6 noutff(iu), dpt(iu), rmuinlt, nogrp, pdrop,
7 netwk, fpri(iu), dppri(iu), lblok(mk,2),
8 iblokt, rtin, hgin(mj), hgout(mj), hain(iu),
9 ghead(mc), dpcor(mc), walls(iu), ilcs(lit,mq),
1 nramp, ninff(iu), phead, haout(iu), ibloka(mc),
2 iblokc(mc), endd2

c
real kij
logical lcass
logical lamnf
c level 2, /bgrid/
c *comdeck time
c
common /timez/ tbeg1(1), dt, iprnte, iprnta, ttime,
1 etime, clock(8), maxt, ndt, ndtpl, itstep, endt1

```

```

c
common /outout/ tbeg2(1), output(12), print(14), tdummy(10),
1 tprint, trant, naxp, nout, npchan, npgap, i2, i3,
1 ipost, jhot, jpres, jrods, jslab,
2 npnode, nprod, printn(10), nskipt, nskipx, nq, ng, ig,
3 nh, ihh, np, im(ms), jm(ms), imp(10), jmp(10), i8,
4 kase, printc(mc), printg(mg), printr(mr), ipreb,
5 ncout, nrout, ngout, nwout, npwall, printw(mw),
* postr(mr), postw(mw), posth(mc), postp(mc),
6 conv(2), iotemp, endt2

c
common /chrvar/ h1, h2, h3, h4, h5, h6, h7, h8

c
character*1 h1, h2, h3, h6, h7
character*4 h4, h5
character*2 h8

c
common /tables/ tbeg3(1), yp(lmm), fp(lmm), yg(lmm), fg(lmm, liw),
1 fh(lmm), yq(lmm), fq(lmm, liw), yt(mp), yh(lmm),
2 fhx(lmm), yhx(lmm), ft(mp), y(mp, mv), axial(mp, mv),
3 naflx(iu), tmncvl(iu), recl(lmm), fflc(lmm, liw),
4 nax(mv), nlcfp, nolc, nlcff, nhx, endt3

c
common /bound/ tbeg4(1), hout, njump, ntries, hin(2), gin, tin, dps,
1 itdp, itdpa(iu), ngprfl, nqprfl, npfvt(iu), exith,
2 exitmu, exitt, negf(iu), negflo, fttotal, endt4

c
common /index/ tbeg5(1), nhigh(mc), nlow(mc), ncount(mr), ihigh,
1 ilow, ilowpl, npoint, ihalt, endt5

c
common /convrq/ tbeg6(1), qoh, accely, werry, accelc, ferror, qerror
1 , dampng, accelh, accrho, accelw, herror, qor, endt6

c
integer printc, printg, printn, printr, printw
logical print

c
*comdeck fuels
c
common /fuel/ flbeg1(1), kfuel(mt), kclad(mt), rfuel(mt), rclad(mt),
1 cfuel(mt), cclad(mt), tclad(mt), hgap(mt), powr(ij),
2 ntheta, nrdthta, ngax, nc, nrod, nrodtp, nfuelt, recall,
3 bx(2, mn, mr), zx(mn, mr), qtotal, scladmx,
4 scladmox, cladmx, acladm(2), rcladm(2), zcladm(2),
5 lr(mr, 6), dfuel(mt), phi(mr, 6), radial(mr), d(mr),
6 idfuel(mr), nhfvt(iu), sparev(20), qvoid(mr),
7 idtyp(mt), nzone(mt), iztyp(mt, my), zend(mt, my),
8 nclad(mr, 6), nwall(mr, 10), nrr(mr), nrw(mr), rcon, rrdim,
9 wcon, wrdim, delrad(mt), dodr(mn, mt), dsqr(mn, mt),
9 conr(mr), endfl1

c
common /repeat/ flbeg2(1), drod(mt), geomf(mt), dfueli(mt),
1 rcfuel(mt), rcclad(mt), r2kf(mt),
2 phitot(mr), ring(mr), rang(mr), rung(mr),
3 hgap1(mt), lri(lir, mc), troid(m7, mr, mx), rtran, endfl2

c
real kfuel, kclad
integer acladm, rcladm, zcladm
c level 2, /fuel/
c level 2, /repeat/
c
*comdeck walls
c
common /walls/ wbeg1(1), kside(iu, m11), uwal(mh1, mw1), qwal(mw),

```

```

1      kwal(mh,mw), icf(nd,mw), twsave(mw), rdimf(liz),
2      icon(mh,mw), rwal(liz), wid(liz),
3      nsidf(mw), nsidw(mw), relxk(mw), nkss, nksf,
4      wallxc(mw), nrpf, qslab(mw), naxk(mw), endw1
c
1      common /radi8/ wbeg2(1), rser(liz), rpar(liz), ema(liz), emb(liz),
2      arad(liz), radsav(2,mw), sumsav(iu,m11), t4(m6),
3      radgap(liz), emx(lnz), brad(lnz,m8), areas(nv,lnz),
4      fv1(la,la), fv2(lb,lb), fv3(llc,llc), fv4(ld,ld),
5      fv5(le,le), fv6(lf,lf), rratio(6,lnz), irad,msid(iu),
6      itypr(iu), fvw(nv,lnz,lnz), sumr(mr), emv(6,lnz),
      emrsav(6), radc(mr), sumrg(lnz,nv), endw2
c
1      common /walbnd/ wbeg3(1), cs1(lif), cs2(lif), ems(lif), ts(iq,lip,mx),
2      nzoneb(lip,iq), nbctyp(lip,iq,mx), zendb(lip,iq,mx),
3      nzonet(no), zendt(mx,no), bt(mx,no),
4      nbttyp(iq), nbtemp,qbsrc(iq), qpsrc(2,2),
5      tamb(iq,mx), sper(lip,iq), qside(mx),
      nhsid(iq), nwsid, iwsid(iq), awsid(iq), nbtyp, endw3
c
1      common /plena/ wbeg4(1), npr, iplenr(2), awps(lip), awpax(lip),
2      npsid(2), npax(2), tambp(2,2), nptyp(lip,2,2),
3      sperp(lip,2,2), ttop(lip,2,2), qtop(2), pexit, ifpln,
4      npw(2,2), cpsav(2), hsave(2), exittod(2), iunca(iu),
5      dxwall, dxplen(mw2), ipw(mw), ipintp(2,mw2),
6      hplen, endw4
c
1      common /redstf/ wbeg5(1), sldamp, nslgrp, bslgrp(6), eslgrp(6), endw5
2      integer bslgrp, eslgrp
c
1      level 2, /walls/
c
1      level 2, /radi8/
c
1      level 2, /walbnd/
c
1      level 2, /plena/
*comdeck rebal
c
1      common /rebal/ rbeg1(1), qcoef(iu), coefh(mc), coefw(iu,lnz),
2      caxld(mx2,3), qres(mx2), dtax(mx2),
3      qcon(iu), endr1
c
1      level 2, /rebal/
c

```

## COBRA-SFS, Cycle 2 Common Block Variable Definitions

### COMDECK GEOM

iloc(ln8)	array storing the number of gap connections seen by a given channel
common /geom/	
gbegl(1)	variable defining beginning address for common block <i>geom</i>
ik(mg)	index of lower-numbered channel of the pair connected by a given gap k
jk(mg)	index of higher-numbered channel of the pair connected by a given gap k
length(mg)	centroid length of a given gap k, units: ft
gangle(mg)	angle of cross-flow velocity vector in a given gap, with respect to the vertical; converted to radians from input value in degrees
ikw(nd,mw)	array storing index numbers of channels with thermal connections to a given slab node
z	total axial length of channels, units: ft
ftm	user-defined turbulent mixing parameter (dimensionless)
theta	channel orientation angle with respect to the vertical. units: degrees
atotal	sum of inlet axial cross-sectional area at all channels in an assembly. units: ft <sup>2</sup>
gelev	angle of lateral control volume length with respect to the vertical; units: degrees
angle(5)	array used to read orientation angle for lateral flow velocity vectors; units: input as degrees from the vertical
elev	channel orientation with respect to the vertical; units: degrees
sl	nominal ratio of gap width to centroid length (dimensionless)
dx(mx)	axial node length array, units: ft
gk(lit)	multiplier on lateral control volume length for thermal conduction in the fluid
ibandw(iu)	solution matrix bandwidth for a given assembly
an(mc)	nominal channel area, as specified by input; converted to units of ft <sup>2</sup>



dhydnc(mc)	nominal channel hydraulic diameter, units: ft
ilocs(ln1,mc)	array containing indices of all higher-numbered channels connected through gaps to a given channel
perim(mc)	nominal channel wetted perimeter, as specified by input; converted to units of ft
per(mc)	summation of perimeters of all slab nodes connected to a given channel; units: ft
hperim(mc)	nominal channel heated perimeter, as specified by input; converted to units of ft
dduct(iu,lix)	array storing index number of the last channel in a given assembly that sees other channels through crossflow connections
abar(mc)	fluid density in the node above the channel exit; units: lbm/ft <sup>3</sup>
endg1	variable defining ending address for common block <i>geom</i>
<b>common /flags/</b>	
gbeg2(1)	variable defining beginning address for common block <i>flags</i>
ierror	flag for table interpolation error
j1	flag for control of processing of output of input data; defined by user input
iterat	counter for external solution iterations
nrc	maximum number of rods interacting with a channel
nwk	total number of slab nodes specified by user input
naah	maximum number of thermal plus flow connections to a channel, plus 1
naahp1	maximum number of thermal plus flow connections to a channel, plus 2
nspec(20)	array to store required values of spec parameters for a given input
nanglt	number of pairs of channel orientation angles to be specified by user input
isolve	flag to solve the energy equation by a semi-implicit method if not using the fuel conduction solution, or if there are no rods in the problem
iass	index number of current assembly
nfcon	flag for radial conduction in fluid
nscbc	flag for single-phase turbulent mixing model

nramph            number of iterations over which heat generation is to  
                  be ramped into the solution

ischeme           flag for lateral momentum solution; specified by user  
                  input

ibwl             maximum difference between indices of channels  
                  connected by a gap; defines number of columns to the  
                  left of the diagonal for the banded matrix solution in  
                  subroutine SOLVER

ibwr             maximum difference between indices of channels  
                  connected by a gap; defines number of columns to the  
                  right of the diagonal for the banded matrix solution  
                  in subroutine SOLVER

ic                maximum difference between indices of channels  
                  connected by a gap, plus 1; diagonal locator for  
                  banded matrix solver in subroutine SOLVER:

-- for symmetrical matrix, index number of the  
    array column that contains the diagonal element  
    of the banded matrix;

-- for non-symmetrical matrix, the position of the  
    diagonal in the nrow x ibw matrix array

iwide            maximum width of solution matrix; defined by parameter  
                  ie

iclose           flag set to indicate zero flow boundary condition for  
                  all assemblies

iclosa(iu)       flag for zero flow boundary condition on a given  
                  assembly

isym             flag for symmetrical or nonsymmetrical matrix for  
                  banded matrix solution in subroutine SOLVER

nrow             number of rows in the banded matrix for the solver in  
                  subroutine SOLVER

i3post           logical unit number for post-processor output file  
                  "post"

endg2            variable defining ending address for common block  
                  flags

**common /const/**

pi                geometric constant  $\pi$ , defined as 355.0/113.0

dtgc             time step (sec) times the gravitational acceleration  
                  constant; units: ft/sec

gc                gravitational acceleration constant, 32.2 ft/sec<sup>2</sup>

idtgc            inverse of the time step times  $g_c$ ; units: sec/ft

dti                    inverse of the time step; units: sec<sup>-1</sup>  
sb                     Stefan-Boltzmann constant, 4.7611(10)<sup>-13</sup> Btu/sec-ft<sup>2</sup>-R<sup>4</sup>

**common /barea/**

gbeg3(1)               variable defining beginning address for common block  
barea  
afact(ml,ma)           channel axial area variation input array;  
(dimensionless)  
gfact(ml,ms)           gap width variation table input array; (dimensionless)  
axl(ml)                input array for axial locations of axial area  
variation input tables; (dimensionless)  
x(mx)                 array of axial level locations; units: ft  
gapxl(ml)             input array for axial locations of gap width variation  
input tables; (dimensionless)  
naramp                number of iterations over which to insert geometry  
variations into solution  
nafact                number of channels with axial area variation  
naxl                 number of axial locations in axial area variation  
tables specified by user input  
ngxl                 number of axial locations in gap width variation  
tables specified by user input  
ngaps                number of gaps with gap width variations  
nch(ma)              temporary storage array for index numbers of channels  
with area variations  
ngap(ms)             temporary storage array for index numbers of gaps with  
gap width variations  
idarea(mc)            array storing index number of area variation table  
used by a given channel i  
idgap(mg)            array storing index number of gap width variation  
table used by a given gap k  
gapn(mg)             nominal gap width for gap k, converted from input  
value; units: ft  
endg3                variable defining ending address for common block  
barea

**COMDECK FIELDS**

**common /field1/**

fbeg1(1)              variable defining beginning address for common  
block field1

wtemp(mg)	tentative cross-flow array, in level-by-level solution of axial momentum equation, units: lbm/sec-ft
p(mc, mx)	fluid pressure in each channel at each axial level; expresses as differential pressure relative to boundary exit pressure; units: lbf/ft <sup>2</sup>
f(mc, mx)	fluid flow rate in each channel at each axial level; units: lbm/sec
fold(mc, mx)	flow field from previous time step; units: lbm/sec
h(mc, mx)	fluid enthalpy in each channel in each axial node; units: Btu/lbm
hold(mc, mx)	enthalpy field from previous time step; units: Btu/lbm
rho(mc, mx)	fluid density in each channel in each axial node; units: lbm/ft <sup>3</sup>
rhoold(mc, mx)	density field from previous time step; units: lbm/ft <sup>3</sup>
rhobar(mc, mx)	average density in channel i between axial levels j and j+1, for transport quantities in lateral momentum equation solution; units: lbm/ft <sup>3</sup>
w(mg, mx)	cross-flow in each gap k at each axial level; units: lbm/sec-ft
wold(mg, mx)	cross-flow field from previous time step
dfdp(mc, mx)	derivative of flow with respect to pressure in each channel at each axial level; units:
dwdp(mg, mx)	derivative of crossflow with respect to pressure for each gap at each axial level; units:
twall(mw, mx)	temperature in each slab node at each axial level; units: °F
twold(mw, mx)	slab temperature field from previous time step; units: °F
ftyp(mr, mx)	real array storing integer values of array of material type indices in different axial fuel zones of a given axially varying fuel type table
flux(mr, mx)	surface heat flux for each rod in each axial node; units: Btu/sec-ft <sup>2</sup>
hsurf(mr, mx)	surface heat transfer coefficient for each rod in each axial node; units: Btu/sec-ft-°F
trod(mn, mr, mx)	rod temperature array, for each node of each rod in each axial node; units: °F
a(mc, mx)	axial flow area of each channel at each axial level; units: ft <sup>2</sup>
dhyd(mc, mx)	hydraulic diameter of each channel at each axial level; units: ft

gap (mg, mx)	gap width for each gap at each axial level; units: ft
app (mc, mx)	axial flow area, averaged over level j, j-1, and j+1, for each channel at each axial level j; units: ft <sup>2</sup>
wp (mg)	turbulent cross-flow for momentum exchange, in each gap k for the current level in the level-by-level solution of the axial momentum equation; units: lbm/sec-ft
dfdpd (mx)	summation of dfdp arrays for all channels at all levels, in the one-dimensional solution in subroutine ONED
dfdp0 (mx)	difference of dfddp arrays for all channels at level j, in the one-dimensional solution in subroutine ONED
dtii (mg)	turbulent momentum transport term for gap k when the donor channel is channel ii
dtjj (mg)	turbulent momentum transport term for gap k when the donor channel is channel jj
endf1	variable defining ending address for common block field1
common /field2/	
fbeg2 (1)	variable defining beginning address for common block field2
hah (ie, mc)	storage array for coefficient matrix in energy and momentum solution
saver (mc)	array to return the residual from the matrix solution of the axial momentum equation
flow (mc)	array to store summation of all cross-flows into or out of a given channel, for the energy solution
qprim (mc)	local heat deposited in channel i, calculated in each node; units: Btu/sec-ft
ichan (ln6, mc)	array storing index numbers of rods seen by a given channel
const (lib)	array storing the source term vector in the matrix formulation for the solution of the momentum equations
hfilm (mc)	array storing the local value of the heat transfer coefficient for a given channel; units: Btu/s-ft <sup>2</sup> -°F
tmh (mc)	array storing the local fluid temperature error for the calculation of the spatial derivative of enthalpy for the energy equation
ftemp (mc)	array storing the previous iteration values of the flow rate in each channel at a level, for the momentum and energy equation solutions

fsp(mc) friction loss in channel i, calculated in each axial node; (dimensionless)

tinlet(mc) input array for specifying channel inlet temperature for each channel; units: °F

finlet(mc) input array for specifying channel inlet flow rate for each channel; units: lbm/sec or dimensionless, depending on user specified input options

hinlet(mc) input array for specifying channel inlet enthalpy for each channel; units: Btu/lbm

fmomf(mc) array storing tentative axial flows solved for in the momentum solution at each channel at an axial level

dpk(mc) array storing the term for local pressure loss due to friction and form drag, in the momentum equation solution; units: (lbm/ft<sup>3</sup>)<sup>-1</sup>

dpdx(mc) array storing derivative of pressure with respect to axial length in each channel at a given axial level

relxi(mc) array storing the value of the relaxation term for each channel at a given axial level in the solution of the fluid energy equation

dhdx(mc) derivative of enthalpy with respect to axial length for each channel at a given axial location

dfdx(iu) derivative of axial flow rate with respect to axial length for each channel at a given axial level

v(mc) array storing local value of fluid specific volume for each channel at a given axial level; units: ft<sup>3</sup>/lbm

cons(mx) array storing the total axial continuity error at a given axial level in the one-dimensional rebalancing solution; also, array storing the continuity error at a given axial level for a given channel in the channel-by-channel solution of the continuity equation

nach(mc) array storing the index number of the assembly that a given channel resides in

itypin(lit,3) array storing identifying information for a given assembly type; i.e., type index number, number of channels, and number of gaps

firstt time returned for first call to cpu timing routine at the beginning of a calculation

endf2 variable defining ending address for common block *field2*

**common /bundle/**

fbeg3(1) variable defining beginning address for common block *bundle*

ipoint(iu,8)	array storing identification information for each bundle, including assembly index number, index numbers of first and last channels in the assembly, and index numbers of first and last gaps in the assembly
ngap1	global index number of first gap in a given assembly
ngap2	global index number of last gap in a given assembly
nchan1	global index number of first channel in a given assembly
nchan2	global index number of last channel in a given assembly
nassem	total number of assemblies
nchanl	total number of channels
mm	variable to store value of parameter lmm; maximum number of points in input profile tables
it	variable to store value of parameter lit; maximum number of assembly types
ip	variable to store value of parameter lip; maximum number of radial boundary regions
ia	variable to store value of parameter mr; maximum number of fuel rods
iw	variable to store value of parameter liw; maximum number of loss coefficient forcing function tables
iz	variable to store value of parameter liz; maximum number of slab-to-slab and slab-to-fluid connection types
ix	variable to store value of parameter lix; maximum number of flow separated regions within a single assembly
ir	variable to store value of parameter lir; maximum number of rods that interact with a given channel
if	variable to store value of parameter lif; maximum number of thermal boundary connection types
nz	variable to store value of parameter lnz; maximum number of surfaces in any user-defined radiation group
nk	total number of gaps
maxtyp	maximum number of assembly types in a given case
ndxp1	total number of axial levels
ndx	number of axial nodes
endf3	variable defining ending address for common block <i>bundle</i>

common /fields3/

conk (mg) array storing local value of fluid thermal conduction in the lateral direction for a given gap at an axial level

dpbar (lip\*iq) array storing plenum node temperatures written to the output file for the plenum temperature summary

common /fieldb/

term1 (mc) array storing input values for inlet mass flow fraction for each channel in an assembly

wpp (mg) array storing the sum of the turbulent cross-flow term and the fluid conduction term for the lateral momentum equation solution in each gap at a give axial level

COMDECK PROPS

common /prop1/

pbeg1 (1) variable defining beginning address for common block prop1

pliq (mp,2) input array for fluid property tables; pressure (psia)

temliq (mp,2) input array for fluid property tables; temperature (°F)

hliq (mp,2) input array for fluid property tables; enthalpy (Btu/lbm)

conliq (mp,2) input array for fluid property tables; thermal conductivity (Btu/h-ft-°F)

cpliq (mp,2) input array for fluid property tables; specific heat (Btu/lbm-F)

visliq (mp,2) input array for fluid property tables; viscosity (lbm/ft-h)

vliq (mp,2) input array for fluid property tables; specific volume (ft<sup>3</sup>/lbm)

bliq (mp,2) input array for fluid property tables; coefficient of thermal expansion, (1/°F)

tvary (mp) input array for temperature-varying material properties tables; temperature (°F)

varyk (mp) input array for temperature-varying material properties tables; thermal conductivity (Btu/h-ft-°F)

varycp (mp) input array for temperature-varying material properties tables; specific heat (Btu/lbm-°F)

varyr (mp) input array for temperature-varying material properties tables; density (lbm/ft<sup>3</sup>)



ntnode	number of entries in the temperature dependent material properties table for fuel type 1
nsett	total number of fluid properties tables; (maximum is 2)
nfprop(2)	number of entries in each input fluid property table
nviscw	flag for hot wall correction to fluid viscosity calculation
data(mpp)	data array for interpolated table(s) of fluid properties, created in 10°F increments from input properties table(s)
tmelt(2)	starting temperature for each fluid property table interpolated in 10°F increments from the input table values; defined as 10°F below the first entry in the user-specified input table
tboil(2)	ending temperature for each interpolated fluid property table; defined at the last entry in the user-specified input table
nfass(iu)	index number of fluid properties table to be used in a given assembly
endp1	variable defining beginning address for common block <i>prop1</i>
<b>common /prop2/</b>	
pbeg2(1)	variable defining beginning address for common block <i>prop2</i>
pref	system pressure; units: psia
lamnh(liw)	logical variable; set to .true. if laminar coefficients are defined by user input as non-zero for a given heat transfer correlation
nheat	number of heat transfer coefficient correlation specified by input in group HEAT
nflmc(iu)	index of heat transfer correlation to be used in a given assembly
ah1(liw)	Reynolds number multiplier coefficient for heat transfer correlation(s) for turbulent axial flow
ah2(liw)	Reynolds number exponent for heat transfer correlation(s) for turbulent axial flow
ah3(liw)	Prandtl number exponent for heat transfer correlation(s) for turbulent axial flow
ah4(liw)	additive coefficient for heat transfer correlation(s) for turbulent axial flow
ahl1(liw)	Reynolds number multiplier coefficient for heat transfer correlation(s) for laminar axial flow

ahl2(liw)	Reynolds number exponent for heat transfer correlation(s) for laminar axial flow
ahl3(liw)	Prandtl number exponent for heat transfer correlation(s) for laminar axial flow
ahl4(liw)	additive coefficient for heat transfer correlation(s) for laminar axial flow
aht1(liw)	Reynolds number multiplier coefficient for heat transfer correlation(s) for turbulent lateral flow
aht2(liw)	Reynolds number exponent for heat transfer correlation(s) for turbulent lateral flow
aht3(liw)	Prandtl number exponent for heat transfer correlation(s) for turbulent lateral flow
aht4(liw)	additive coefficient for heat transfer correlation(s) for turbulent lateral flow
ahtl1(liw)	Reynolds number multiplier coefficient for heat transfer correlation(s) for laminar lateral flow
ahtl2(liw)	Reynolds number exponent for heat transfer correlation(s) for laminar lateral flow
ahtl3(liw)	Prandtl number exponent for heat transfer correlation(s) for laminar lateral flow
ahtl4(liw)	additive coefficient for heat transfer correlation(s) for laminar lateral flow
ahf1(liw)	Rayleigh number multiplier coefficient for heat transfer correlation(s) for free convection axial flow
ahf2(liw)	Rayleigh number exponent for heat transfer correlation(s) for free convection axial flow
istar(mg)	in-line function to define donor channel for cross-flow in a given gap
ntheat	number of forced convection heat transfer correlations specified by user input
nfheat	number of free convection heat transfer correlations specified by user input
endp2	variable defining beginning address for common block prop2
<b>common /prop3/</b>	
pbeg3(1)	variable defining ending address for common block prop3
visc(mc,3)	viscosity of a fluid node; units: lbm/ft-s
viscw(mc)	viscosity in a node at a heated wall; units: lbm/ft-s

con(mc)	thermal conductivity in a fluid node; units: Btu/s-ft-°F)
cp(mc)	specific heat of the fluid in a node; units: Btu/lbm-°F)
vp(mc)	specific volume of a fluid node; units: ft <sup>3</sup> /lbm
t(mc)	temperature in a fluid node; units: °F
vscm(mc)	local value of the heat flux profile for a given axial level; units: Btu/s-ft <sup>2</sup>
beta(mc)	array storing the local coefficient of thermal expansion in a fluid node at a given axial level
ntype(mc)	array storing index number of axial friction factor correlation for a given channel
amix(lit)	multiplicative coefficient for lateral turbulent mixing correlation(s)
bmix(lit)	Reynolds number exponent for turbulent mixing correlation(s)
endp3	variable defining beginning address for common block <i>prop3</i>
 <b>common /matprp/</b>	
pbeg4(1)	variable defining beginning address for common block <i>matprp</i>
uwcp(nd,mw)	sum of the thermal conductances for heat transfer between a fluid node and a wall node divided by the specific heat of the fluid node, at a given axial level
uwall(nd,mw)	sum of the thermal conductances for heat transfer between a fluid node and a wall node, at a given axial level
areacp(mw2)	heat transfer area for axial connections between a given slab node and a plenum axial region; units: input in in <sup>2</sup>
matypp(mw2)	array storing index numbers of material type for connections between a given slab node and a plenum axial region
tpsava(mw2)	array to save the previous iteration value of the plenum fluid temperatures, in the energy equation solution
npwcon	total number of slab nodes with thermal connections to the plenum
con0(id)	leading additive coefficient for correlation(s) for temperature-dependent solid material thermal conductivity

con1(id) multiplicative coefficient on temperature for temperature-dependent solid material thermal conductivity  
 con2(id) multiplicative coefficient on temperature-squared term for temperature-dependant solid material thermal conductivity  
 con3(id) multiplicative coefficient on temperature-cubed term of temperature-dependant solid material thermal conductivity  
 cpsol(id) solid material properties table input array; specific heat (Btu/lbm-°F)  
 rhosol(id) solid material properties table input array; density (lbm/ft<sup>3</sup>)  
 rdimal(liz) geometry factor for the lower-numbered node of any pair of nodes connected by a given thermal connection type  
 rdimb(liz) geometry factor for the higher-numbered node of any pair of nodes connected by a given thermal connection type  
 nqvt(id) index number of transient heat rate forcing function to be used for user specified solid material properties  
 conw(mw) array storing current value of boundary heat flux correlation for each boundary slab node at a given axial level  
 matj(id,mx) array storing index numbers of fuel material types for profiles of axially varying fuel material type  
 mattyp(mw) array storing index of material type for each slab node  
 imat(id) solid material type index number  
 pdna(iu) array storing heat generation rate for each assembly  
 pdn average heat generation rate; units: input as Mbtu/h-ft<sup>2</sup>  
 nsprop number of solid material properties tables specified by input  
 endp4 variable defining ending address for common block *matprp*

common /alpha/

aname(id) alphanumeric identifier for solid material properties table  
 alph alphanumeric variable used to store date and time information for printout to output file(s)

## COMDECK DRAGS

### common /drag/

dbeg1(1) variable defining beginning address for common block  
*drag*

dpa(iu) array storing pressure drop boundary condition for  
each assembly; units: input in psi

kij(mg) loss coefficient for lateral flow in gap connections

cij(mg) array storing the combined form drag and friction loss  
term for the lateral momentum equation in each gap at  
a given axial level

itran flag for lateral convection heat transfer correlations

aa(liw) first multiplicative coefficient on the Reynolds  
number for turbulent flow friction factor  
correlation(s)

bb(liw) first Reynolds number exponent for turbulent flow  
friction factor correlation(s)

cc(liw) second multiplicative coefficient on the Reynolds  
number for turbulent flow friction factor  
correlation(s)

dd(liw) second Reynolds number exponent for turbulent flow  
friction factor correlation(s)

ee(liw) additive coefficient for turbulent flow friction  
factor correlation(s)

aal(liw) multiplicative coefficient on the Reynolds number for  
laminar flow friction factor correlation(s)

bb1(liw) Reynolds number exponent for laminar flow friction  
factor correlation(s)

ccl(liw) additive coefficient for laminar flow friction factor  
correlation(s)

lamnf(liw) logical variable; set to .true. if laminar  
coefficients are defined as non-zero for a given  
friction factor correlation

aat(liw) first multiplicative coefficient on the Reynolds  
number for lateral wall friction factor correlation(s)

bbt(liw) first Reynolds number exponent for lateral wall  
friction factor correlation(s)

cct(liw) second multiplicative coefficient on the Reynolds  
number for lateral wall friction factor correlation(s)

ddt(liw) second Reynolds number exponent for lateral wall  
friction factor correlation(s)

eet(liw) additive coefficient for lateral wall friction factor correlation(s)  
 aatl(liw)) multiplicative coefficient on the Reynolds number for laminar flow lateral wall friction factor correlation(s)  
 bbt1(liw) Reynolds number exponent for laminar flow lateral wall friction factor correlation(s)  
 cctl(liw) additive coefficient for laminar flow lateral wall friction factor correlation(s)  
 nfrict number of axial friction factor correlations specified by user input  
 ishear user specified input flag for fluid shear stress model  
 nshear number of pairs of gaps that are connected by fluid-fluid shear in the lateral direction  
 kn(mg,2) array storing index numbers of gaps connected to a given gap for fluid-fluid shear  
 ck(mg,4) array storing shear stress terms in each gap at a given axial level for fluid-fluid shear in the lateral momentum equation  
 endd1 variable defining ending address for common block drag

**common /bgrid/**

dbeg2(1) variable defining beginning address for common block *bgrid*  
 factor(lit,mq,mx) array storing relative axial locations for loss coefficients to be applied to a specified set of channels in a given assembly  
 ilce(lit,mq) array storing identification number of ending channel in a location set of loss coefficients in a given assembly  
 lcass(iu) logical variable; set to .true. if an assembly has form drag loss coefficients defined by user input  
 lcff(mk) array storing index numbers of loss coefficient forcing functions to be applied to loss coefficients in channels of a given assembly  
 lcast(mk) array storing index numbers of assembly types with loss coefficients  
 ifact(mq,mx) array storing axial level locations of axial form drag loss coefficients for a given channel  
 nlevel(lit,mq) number of axial locations in a given location set of loss coefficients for a given assembly

cd(lit,mq,mx) array storing axial form drag loss coefficients at a given axial level in a given channel of a given assembly type

xcd(lit,mq,mx) array storing loss coefficients to be applied in a specified set of channels in a given assembly at a specified axial location

ifb(mc) array used to flag a blocked axial level in a channel at a given axial level

nblock number of axial locations where flow blockages are specified by user input

flo(iu) array storing total assembly flow when the network model is used

allf summation of all assembly flows when the network model is used

nbl(lit) number of sets of channels in a given assembly having loss coefficients

flog(mj) array storing total flow in all assemblies of a given group for the network model

pwini(iu) wetted perimeter associated with the inlet loss in a given assembly when using the network model

pwout(iu) wetted perimeter associated with the outlet loss in a given assembly when using the network model

rain(iu) inlet loss parameter to be applied to a given assembly when using the network model; units:  $(\text{ft-lbm})^{-1}$

raout(iu) outlet loss parameter to be applied to a given assembly when using the network model; units:  $(\text{ft-lbm})^{-1}$

rgin(mj) inlet loss parameter to be applied to a given assembly group when using the network model; units:  $(\text{ft-lbm})^{-1}$

rgout(mj) outlet loss parameter to be applied to a given assembly group when using the network model; units:  $(\text{ft-lbm})^{-1}$

netgrp(iu) assembly grouping number in the network model for a given assembly

noutff(iu) array storing identification number of loss coefficient forcing function versus Reynolds number to be used at the assembly outlet in the network model

dpt(iu) array storing pressure drop in each assembly; units:  $\text{lbf/ft}^2$

rmuinlt fluid viscosity at the inlet enthalpy of channel number 1; used to calculate the Reynolds number of the inlet flow in the network model

nogrp number of assembly groups in the network model

pdrop variable storing pressure drop boundary condition for network model; units: lbf/ft<sup>2</sup>

netwk flag for optional network pressure drop model

fpri(iu) array storing the previous iteration value of the total pressure drop in each assembly; units: lbf/ft<sup>2</sup>

dpfri(iu) array storing the previous iteration value of the friction pressure gradient in each assembly, for use in the network model

lblok(mk,2) array storing index numbers of axial levels of blockages, and channels that have blockages

iblokt total number of local blockages

rtin loss parameter on the total flow rate summed over all assemblies, for the network model; units: (ft-lbm)<sup>-1</sup>

hgin(mj) gravitational head length associated with inlet loss in a given assembly group inlet in the network model; units: input in in.

hgout(mj) gravitational head length associated with outlet loss in a given assembly group inlet in the network model; units: input in in.

hain(iu) gravitational head length associated with the assembly inlet loss in a given group in the network model; units: input in in.

ghead(mc) pressure drop due to gravity head in a given channel, for network model calculations

dpcor(mc) array storing the total momentum pressure drop in each channel at a given axial level

walls(iu) array storing the current iteration value of the friction pressure gradient in each assembly, for use in the network model

ilcs(lit,mq) array storing the index of the starting channel in a location set of loss coefficients in a given assembly

nramp number of iterations over which loss coefficients are to be ramped into the solution

ninff(iu) array storing identification number of loss coefficient forcing function versus Reynolds number to be used for the inlet loss of a given assembly for the network model

phead assembly average pressure drop, to be used in calculations for the network model

haout(iu) gravitational head length associated with the assembly outlet loss in a given group in the network model; units: input in inches

ibloka(mc) array storing index numbers of assemblies containing specified blocked channels



iblokc(mc)           array storing index numbers of blocked channels in a  
                       given assembly  
 endd2                variable defining ending address for common block  
                       **bgrid**

**COMDECK TIME**

**common /timez/**

tbeg1(1)            variable defining beginning address for common block  
                       **timez**  
 dt                   time step; units: seconds  
 iprnte              flag to activate option for output of assembly average  
                       and channel exit values  
 iprnta              flag to activate option for output of channel exit  
                       values  
 ttime               total transient time specified by user input; units:  
                       seconds  
 etime               elapsed simulation time; units: seconds  
 clock(8)            array storing cpu time intervals spent in different  
                       parts of the solution during code execution; units:  
                       seconds  
 maxt                maximum cpu time allowed for code execution; units:  
                       seconds  
 ndt                  number of time steps  
 ndtp1               number of specified time steps, plus 1  
 itstep              number of entries in variable time step table  
 endt1               variable defining ending address for common block  
                       **timez**

**common /outout/**

tbeg2(1)            variable defining beginning address for common block  
                       **outout**  
 output(12)          array for storing calculated output variables  
 print(14)           logical variable; set to .true. if input options  
                       specify that processed input is to be written to the  
                       output file  
 tdummy(10)          array to store rod values in degrees Celsius for  
                       output  
 tprint              printout interval for determining the time steps  
                       written to the output file in a transient calculation;  
                       units: seconds

trant	time interval between successive output time steps is a transient calculation
naxp	number of axial heat generation profile tables
nout	flag for all printout, or selected printout controlled by flags ncout, nrout, ngout, and nwout
npchan	number of channels to be printed
npgap	number of gaps to be printed
i2	variable defining logical unit number for reading input
i3	variable defining logical unit number for output file
ipost	logical unit number for output to the post processing file written in subroutine POST
jhot	axial level treated as the bottom of the channel region; level 1 for pressure boundary condition or if the lower plenum model is used, level 2 for flow boundary condition
jpres	option for input group GRAF; (NOTE: developmental feature; not validated, and not documented in Cycle 2)
jrods	option for input group GRAF; (NOTE: developmental feature; not validated in Cycle 2)
jslab	option for input group GRAF; (NOTE; developmental feature; not validated in Cycle 2)
npnode	flag for fuel temperature node printout
nprod	number of rods to be printed
printn(10)	array storing index numbers of fuel rod nodes for which temperatures will be printed in the output file
nskip	interval to skip in time step printout; default is 1, and means no skipped time steps
nskipx	interval to skip in axial node printout; default is 1, and means no skipped nodes
nq	number of time values in each transient forcing function table on average heat generation rate
ng	number of pairs of values in transient forcing function on inlet flow
ig	flag for inlet flow distribution option
nh	number of pairs of values in transient forcing function on inlet enthalpy or temperature
ihh	flag for units of enthalpy or temperature on channel inlet conditions

np	number of pairs of values in transient forcing function on system pressure
im(ms)	array to store ii indices of gaps for output of input data describing channel gap connections
jm(ms)	array to store jj indices of gaps for output of input data describing channel gap connections
imp(10)	array to store ii indices of gaps for a line of output of crossflow results
jmp(10)	array to store ii indices of gaps for a line of output of crossflow results
i8	variable storing logical unit number for restart file (tape8)
kase	index number of a given case
printc(mc)	array storing index numbers of channels to be printed
printg(mg)	array storing index numbers of gaps to be printed
printr(mr)	array storing index numbers of rods to be printed
ipreb	flag to print rebalancing summary with solution iteration summary
ncout	flag for restricted channel output
nrout	flag for restricted rod output
ngout	flag for restricted gap output
nwout	flag for restricted slab node output
npwall	flag for slab node printout
printw(mw)	array storing index numbers of slab nodes to be printed
postr(mr)	array for post-processing logic; (NOTE: developmental feature; not validated in Cycle 2)
postw(mw)	array for post-processing logic; (NOTE: developmental feature; not validated in Cycle 2)
posth(mc)	array for post-processing logic; (NOTE: developmental feature; not validated in Cycle 2)
postp(mc)	array for post-processing logic; (NOTE: developmental feature; not validated in Cycle 2)
conv(2)	array storing factors to convert units on temperature from Fahrenheit to Celsius in the output

iotemp flag for option for °F or °C units on output temperatures  
endt2 variable defining ending address for common block  
outout

common /chrvar/

h1 character variable; left parenthesis, '('  
h2 character variable; period, '.'  
h3 character variable; right parenthesis, ')'  
h4 character variable; ' w('   
h5 character variable; ')wp('   
h6 character variable; 'w'  
h7 character variable; 'x'  
h8 character variable; 't('

common /tables/

tbeg3(1) variable defining beginning address of common block  
tables  
yp(lmm) array storing transient time in the system pressure  
forcing function table; units: seconds  
fp(lmm) array storing time-dependent forcing factor normalized  
to initial value for the system pressure forcing  
function table  
yg(lmm) array storing transient time in the inlet flow or  
pressure drop forcing function tables; units: seconds  
fg(lmm,liw) array storing time-dependent forcing factors  
normalized to initial value for the inlet flow or  
pressure drop forcing function tables  
fh(lmm) array storing time-dependent forcing factor normalized  
to initial value for the inlet enthalpy or temperature  
forcing function table  
yq(lmm) array storing transient time in the heat generation  
rate forcing function tables; units: seconds  
fq(lmm,liw) array storing time-dependent forcing factor normalized  
to initial value for the heat generation rate forcing  
function tables  
yt(mp) array storing transient time in the time-step size  
table for variable time-step size; units: seconds  
yh(lmm) array storing transient time in the inlet enthalpy or  
temperature forcing function table; units: seconds

fhx(lmm)            array storing time-dependent forcing factor normalized to initial value for the exit enthalpy forcing function table

yhx(lmm)            array storing transient time in the exit enthalpy forcing function table; units: seconds

ft(mp)              array storing variable time step sizes for table of variable time step size

y(mp,mv)            array storing relative axial location of elements in axial heat generation rate profile tables

axial(mp,mv)        array storing relative heat generation rate profiles, normalized to the average heat generation rate

naflx(iu)           index number of axial heat flux profile table used in a given assembly

tmncvl(iu)          array storing input value of factor to adjust lateral momentum control volume length for the calculation of turbulent crossflow exchange in all gaps of a given assembly

recl(lmm)           array storing the Reynolds numbers for the loss coefficient forcing function profiles

fflc(lmm,liw)       array storing the loss coefficient factors for the loss coefficient versus Reynolds number forcing function profiles

nax(mv)             number of pairs of entries in a given axial heat generation rate profile table

nlcfp                maximum number of points in input tables of loss coefficient versus Reynolds number, flow or pressure drop versus time, heat flux versus time, system pressure versus time, and inlet or exit enthalpy versus time

nolc                 number of assembly types for which loss coefficients are to be specified by user input

nlcff                number of forcing functions of loss coefficient versus Reynolds number

nhx                 number of pairs of values in transient forcing function on exit enthalpy or temperature

endt3                variable defining ending address of common block tables

**common /bound/**

tbeg4(1)            variable defining beginning address for common block bound

hout                exit enthalpy (Btu/lbm) or temperature (°F)

njump                input flag for restart calculation

ntries maximum number of outer iterations in solution of conservation equations  
 hin(2) average inlet enthalpy or temperature for channels with each specified fluid; units: Btu/lbm or °F  
 gin average inlet mass flux; units: input as Mlbm/h-ft<sup>2</sup>  
 tin average inlet temperature; units: °F  
 dps pressure drop; defined by input for uniform specified pressure drop boundary condition option  
 itdp flag for flow or pressure drop boundary condition option to be applied to all assemblies  
 itdpa(iu) flag for flow or pressure drop boundary condition option in a given assembly  
 ngprfl number of transient forcing functions for inlet mass flux or pressure drop tables  
 nqprfl number of heat generation rate transient forcing function tables  
 npfvt(iu) index number of pressure drop of flow forcing function used by a given assembly  
 exith enthalpy at the channel exit, for transport terms for reverse flow at the exit; units: Btu/lbm  
 exitmu viscosity of the fluid in the plenum node, when the plenum model is used; units: lbm/ft-s  
 exitt temperature of the fluid in the plenum node, when the plenum model is used; units: °F  
 negf(iu) array storing flag indicating the presence of negative axial flow in some channel(s) of a given assembly  
 negflo flag set when negative flow is encountered in a channel  
 ftotal total inlet flow rate; units: (lbm/sec)  
 endt4 variable defining ending address for common block bound

**common /index/**

tbeg5(1) variable defining beginning address for common block index  
 nhhigh(mc) array storing index numbers of channels that interact with a given channel for the higher order interaction terms in the solution of the fluid and rod energy equations solution

nlow(mc)	array storing index numbers of channels that interact with a given channel for the lower order interaction terms in the solution of the fluid and rod energy equations
ncount(mr)	array storing the number of channels that interact with each rod for the interaction terms in the solution of the fluid and rod energy equations
ihigh	current value of nhigh(i) for a given channel i, in the solution of the fluid and rod energy equations
ilow	current value of nlow(i) for a given channel i, in the solution of the fluid and rod energy equations
ilowpl	current value of ilow, plus 1
npoint	current value of ihigh
ihalt	maximum number of iterations in the solution of the fluid and rod energy equations by the method of successive over-relaxation; defined as 100 in subroutine SETIN
endt5	variable defining ending address for common block index
 <b>common /convrg/</b>	
tbeg6(1)	variable defining beginning address for common block convrg
qoh	maximum error in the fluid energy equation
accely	acceleration factor for iterative Gauss-Siedel solution of momentum equations
werry	convergence criterion for continuity solution in inner loop of main interaction loop
accelf	damping factor for iterative axial flow solution
ferror	axial flow convergence criterion for axial momentum equation
qerror	convergence criterion for total energy conservation
dampng	not used
accelh	acceleration factor for successive over-relaxation solution of the rod and fluid energy equations; set equal to accelw, which is specified by input
accrho	damping factor on change in fluid density
accelw	acceleration factor for the solid node energy equation solution
herror	convergence criterion for fluid energy equation

gor maximum error in the solution of the rod energy equation

endt6 variable defining ending address for common block *convrg*

COMDECK FUELS

common /fuel/

flbeg1(1) variable defining beginning address for common block *fuel*

kfuel(mt) thermal conductivity of fuel for a given fuel geometry type; units: input in Btu/h-ft-°F, converted to Btu/s-ft-F

kclad(mt) thermal conductivity of cladding for a given fuel geometry type; units: input in Btu/h-ft-°F, converted to Btu/s-ft-°F

rfuel(mt) density of fuel for a given fuel geometry type; units: lbm/ft<sup>3</sup>

rclad(mt) density of cladding for a given fuel geometry type; units: lbm/ft<sup>3</sup>

cfuel(mt) specific heat of fuel for a given fuel geometry type; units: Btu/lbm-°F

cclad(mt) specific heat of cladding for a given fuel geometry type; units: Btu/lbm-°F

tclad(mt) cladding thickness for a given fuel geometry type; units: input in in., converted to ft

hgap(mt) gap conductance coefficient of the fuel-cladding gap for a given fuel geometry type; units: Btu/h-ft<sup>2</sup>-°F

powr(ij) array storing current value of forcing function multiplicative factor for each power profile table

ntheta number of circumferential nodes in fuel rods

nrthta total number of rod nodes; nrod\*ntheta

nqax flag for temperature dependent fuel material properties

nc flag for fuel model option; if nc>4, number of finite difference nodes in the fuel rod, including one node in the cladding

nrod number of rods

nrodtpt number of tables of axial fuel type variation

nfuel number of fuel types



recall	flag to indicate iterative solution is finished, either because converged, or specified cpu time has been exceeded
bx(2,mn,mr)	array storing the nonzero elements of the triangular matrix in the solution of the rod energy equation
zx(mn,mr)	array used in the matrix algebra for the solution of the rod energy equation
qtotal	total heat generation; units: Btu/sec
scladm	maximum clad surface temperature; units: °F
scldmxo	previous iteration value of maximum clad surface temperature; units: °F
cladm	maximum cladding surface temperature on a given rod; units: °F
acladm(2)	index number of assembly containing rod with maximum cladding surface temperature
rcladm(2)	index number of rod with maximum cladding surface temperature
zcladm(2)	axial level location of maximum rod cladding surface temperature
lr(mr,6)	array containing index numbers of up to six channels seen by a given rod
dfuel(mt)	fuel pellet diameter for a given fuel geometry type; units: input in in., converted to ft
phi(mr,6)	fraction of rod surface that sees a given channel
radial(mr)	rod radial power factor, normalized to average rod power; (dimensionless)
d(mr)	fuel rod diameter; units: ft
idfuel(mr)	array storing fuel geometry type index for each rod
nhfvt(iu)	index number of heat generation rate forcing function used by a given assembly
sparev(20)	spare variable array; used to store information needed temporarily between subroutines
qvoid(mr)	heat generation rate in the central void of an annular fuel rod of a given fuel geometry type; units: Btu/s-ft <sup>3</sup>
idtyp(mt)	array storing index numbers of fuel geometry types
nzone(mt)	array storing index number of fuel type in different zones of rods with axially varying fuel type
iztyp(mt,my)	index of fuel material type table to be used in a given axial zone

zend(mt,my) relative axial location of the end of a fuel zone for an axially varying fuel type table; (dimensionless)

nclad(mr,6) array storing index numbers of up to six rods in contact with a given rod for conduction heat transfer

nwall(mr,10) array storing index numbers of up to ten rods that are in contact with a given wall node

nrr(mr) array storing index numbers of rods in contact with adjacent rods for conduction heat transfer

nrw(mr) array storing index numbers of wall nodes in contact with rods for conduction heat transfer

rcon contact conductance between rods; units: h-ft-°F/Btu

rrdim geometry factor for thermal connections for rod-to-rod contact; (dimensionless)

wcon contact conductance between rods and wall surfaces; units: h-ft-°F/Btu

wrdim geometry factor for thermal connections for rod-to-wall contact; (dimensionless)

delrad(mt) array storing node thickness of each node for the conduction heat transfer solution in the fuel pellet; units: ft

dodr(mn,mt) array storing the node outside diameter divided by the node thickness for each fuel node of a given fuel type

dsqr(mn,mt) array storing the node outside diameter squared, minus the inside diameter squared for each fuel node of a given fuel type

conr(mr) array storing average fluid conductivity in the channels seen by each rod at a given axial level; units: Btu/sec-ft-°F

endfl1 variable defining ending address for common block fuel

**common /repeat/**

flbeg2(1) variable defining beginning address for common block repeat

drod(mt) rod outside diameter for a given fuel geometry type; units: input in inches, converted to ft

geomf(mt) rod geometry flag for a given fuel geometry type

dfueli(mt) inside diameter of annular fuel rod of a given fuel geometry type; units: input in inches, converted to ft

rcfuel(mt) array storing the time derivative term for the fuel material,  $\rho C_p / \Delta t$ , for each fuel type

rcclad(mt) array storing the time derivative term for the cladding material,  $\rho C_p / \Delta t$ , for each fuel type

r2kf(mt)	array storing the coefficient term, fuel diameter squared divided by four times the fuel thermal conductivity, for each fuel type
phitot(mr)	summation of all phi(n,6) terms for a given rod
ring(mr)	array storing constant terms for matrix algebra in solution of rod energy equation
rang(mr)	array storing constant terms for matrix algebra in solution of rod energy equation
rung(mr)	array storing constant terms for matrix algebra in solution of rod energy equation
hgap1(mt)	array storing the total thermal conductance of the gap and cladding for each fuel type
lri(lir,mc)	array storing the index numbers of all rods that see a given channel
trold(m7,mr,mx)	rod temperature at each node, in each rod, at each axial level; units: °F
rtran	local variable storing current value of transient time derivative in rod energy equation
endf12	variable defining ending address for common block repeat

#### COMDECK WALLS

##### common /walls/

wbeg1(1)	variable defining beginning address for common block walls
kside(iu,m11)	array storing index numbers of slab nodes corresponding to the surfaces in a given radiative heat transfer group
uwal(mh1,mw1)	array storing total conductance between adjacent slab nodes
qwal(mw)	array storing summation of all energy entering or leaving each slab node
kwal(mh,mw)	array storing index numbers of solid nodes adjacent to a given slab node for solid-to-solid thermal connections
icf(nd,mw)	array storing index numbers of solid-to-fluid thermal connection type for a given slab node that is connected to a channel
twsave(mw)	array storing previous iteration value of temperatures of all slab nodes connected to a given slab node, at a given axial level

rdimf(liz) geometry factor from the center of the solid node surface to the side facing a channel for a given solid-to-fluid connection type  
 icon(mh,mw) array storing index numbers of thermal connection types for the connections between specific slab nodes  
 rwal(liz) thermal resistance from the center of the solid node to the side facing a channel for a given solid-to-fluid connection type  
 wid(liz) perimeter of solid node facing a channel for a given solid-to-fluid connection type  
 nsidf(mw) array storing the number of thermal connections to adjacent channels for each slab node  
 nsidw(mw) array storing the number of thermal connections to adjacent slab nodes for each slab node  
 relxk(mw) array storing the current values of the diagonal terms in the coefficient matrix for the slab energy equation in the solution by successive over-relaxation  
 nkss number of solid-to-solid node thermal connections  
 nksf number of solid-to-fluid node thermal connections  
 wallxc(mw) axial cross-sectional area of a given slab node; units: input in in<sup>2</sup>  
 nrpf flag for option to specify different heat rate for each assembly  
 qslab(mw) volumetric heat generation rate in a given slab node; units: Btu/h-ft<sup>3</sup>  
 naxk(mw) array storing index number of axial heat generation profile table for a given slab node  
 endw1 variable defining ending address for common block walls

common /radi8/

wbeg2(1) variable defining beginning address for common block radi8  
 rser(liz) thermal resistance between centers of adjacent nodes connected by a given thermal connection type; units: s-ft-°F/Btu  
 rpar(liz) thermal resistance of the gap between adjacent solid structure nodes connected by a given thermal connection type; units: s-ft-°F/Btu  
 ema(liz) surface emissivity of lower-numbered node in a pair of nodes connected by a given thermal connection type  
 emb(liz) surface emissivity of higher-numbered node in a pair of nodes connected by a given thermal connection type

arad(liz) length of node on the face looking at the gap between adjacent nodes connected by a given thermal connection type; units: input in in.

radsav(2,mw) array storing radiation source term and matrix coefficient in the slab energy equation

sumsav(iu,m11) array storing summation of the radiation view factor terms times the fourth power of the surface temperature for all surfaces in a given assembly

t4(m6) array storing the fourth power of the surface temperature for all rod and slab surfaces

radgap(liz) radiation portion of contact conductance for a given slab-to-slab thermal connection type

emx(lnz) emissivity of a given surface in a user-defined radiation group

brad(lnz,m8) array used in calculation of grey body view factors from black body view factor input, using gaussian elimination in subroutine SETIN

areas(nv,lnz) perimeter of a given surface in a user-defined radiation group; units: input in in<sup>2</sup>

fv1(la,la) array storing grey body view factors for radiation group 1, read from tape10

fv2(lb,lb) array storing grey body view factors for radiation group 2, read from tape10

fv3(llc,llc) array storing grey body view factors for radiation group 3, read from tape10

fv4(ld,ld) array storing grey body view factors for radiation group 4, read from tape10

fv5(le,le) array storing grey body view factors for radiation group 5, read from tape10

fv6(lf,lf) array storing grey body view factors for radiation group 6, read from tape10

rratio(6,lnz) array storing ratio of wall node perimeters to rod perimeter, in a radiation group read from tape10

irad flag for radiation heat transfer in energy solution

msid(iu) array storing the number of slab nodes with surfaces in a given radiative heat transfer group

itypr(iu) array storing radiative heat transfer group type for a given assembly

fwv(nv,lnz,lnz) array storing input values of black body view factors between surfaces in a given user-specified radiation group; (maximum of six)

sumr(mr) sum of all radiation view factors for a given rod

emv(6,lnz) array storing emissivity values for wall nodes in a radiation group read from tape10

emrsav(6) array storing the rod emissivity value for each user-defined radiation group

radc(mr) array storing radiation term, emissivity times temperature cubed, for each rod at a given level

sumrg(lnz,nv) array storing summation of all radiation view factors for each surface, in each radiation group read from tape10

endw2 variable defining ending address for common block radi8

common /walbnd/

wbeg3(1) variable defining beginning address for common block walbnd

cs1(lif) first coefficient for user-defined boundary heat flux correlation

cs2(lif) first coefficient for user-defined boundary heat flux correlation

ems(lif) combined surface emissivities of boundary nodes for user-defined boundary heat flux correlation

ts(iq,lip,mx) temperature array for side axial and radial boundary nodes

nzoneb(lip,iq) array storing number of axial zones in a table of boundary connection types versus axial distance for a given boundary node

nbctyp(lip,iq,mx) array storing the thermal boundary connection types for an axial boundary variation table for a given boundary node

zendb(lip,iq,mx) array storing relative axial locations of entries in an axial boundary variation table for a given boundary node

nzonet(no) array storing number of pairs of entries in a given side boundary temperature profile table

zendt(mx,no) array storing relative axial location of a given element in a given side boundary temperature profile table

bt(mx,no) array storing boundary temperatures in a given side boundary temperature profile table

nbttyp(iq) array storing identification number of boundary temperature profile seen by a given boundary node

nbtemp number of axial boundary temperature profiles

qbsrc(iq)	boundary heat flux applied to a given radial region; units: input as Btu/h-ft <sup>2</sup>
qpsrc(2,2)	plenum radial and axial boundary heat flux; units: input as Btu/h-ft <sup>2</sup>
tamb(iq,mx)	array storing boundary temperature seen by each boundary node at each axial level
spcr(lip,iq)	perimeter of a given boundary region of a boundary node, normalized awsid(iq); dimensionless
qside(mx)	array storing total heat loss from side boundary nodes at each axial level
nhsid(iq)	array storing number of radial boundary regions connecting a given node to a boundary temperature
nwsid	number of solid structure nodes connected to the side boundary
iwsid(iq)	array storing index numbers of solid structures nodes connected to side thermal boundary temperatures
awsid(iq)	perimeter of the side facing the boundary for a given solid structure node connected to a side boundary temperature
nbtyp	number of thermal boundary connection types
endw3	variable defining ending address for common block walbnd
 <b>common /plena/</b>	
wbeg4(1)	variable defining beginning address for common block <i>plena</i>
npr	plenum model flag
iplenr(2)	array storing plenum type flag
awps(lip)	nominal plenum radial heat transfer area; units: input in in <sup>2</sup>
awpax(lip)	nominal plenum axial heat transfer area; units: input in in <sup>2</sup>
npsid(2)	number of radial plenum boundary regions
npax(2)	number of axial plenum boundary regions
tambp(2,2)	plenum radial and axial boundary temperatures; units: °F
nptyp(lip,2,2)	array storing thermal connection type number for axial connections between slab nodes and plenum
spcrp(lip,2,2)	plenum radial region heat transfer area multiplier; normalized to awps(lip):(dimensionless)

ttop(lip,2,2)	array storing fluid and radial boundary node temperatures in plena
qtop(2)	total energy lost through the plena
pexit	system pressure; units: psia
ifpln	flag for fluid properties table to use in plena
npw(2,2)	array storing starting and ending slab node numbers for thermal connections to plena
cpsav(2)	average specific heat of fluid in the plena
hsave(2)	average enthalpy in the plena
exittod(2)	previous iteration value of average fluid temperature in the plena
iunca(iu)	array storing index numbers of assemblies not connected to the plenum model
dxwall	half-node length for axial fluid connection between channels and the plena in the energy equation solution
dxplen(mw2)	length for heat transfer from a given slab node to the center of a plenum radial region; units: input in inches
ipw(mw)	array storing number of solid-to-solid thermal connections for each slab node
ipintp(2,mw2)	array storing index numbers of slab nodes connected to a given plenum radial region
hplen	average fluid enthalpy for energy transport due to reverse flow at the channel exit or inlet when the plenum model is used
endw4	variable defining ending address for common block <i>plena</i>

**common /redstf/**

wbeg5(1)	variable defining beginning address for common block <i>redstf</i>
sldamp	damping factor for slab conduction equation in the energy solution
nslgrp	number of groups of slab nodes for which the conduction solution is damped
bslgrp(6)	index number of the first node in a set of sequentially numbered nodes for which the conduction solution is damped
eslgrp(6)	index number of the last node in a set of sequentially numbered nodes for which the conduction solution is damped



endw5                    variable defining ending address for common block  
redstf

**COMDECK REBAL**

**common /rebal/**

rbeg1(1)                variable defining beginning address for common block  
rebal

qcoef(iu)              array storing rod dependent terms for each assembly in  
solution of the fluid and rod energy equations

coefh(mc)              array storing radiation terms for each channel in an  
assembly, in the solution of the fluid and rod energy  
equations

coefw(iu,lnz)         array storing radiation contribution to source term  
for each surface in each assembly, in the fluid and  
rod energy equation solution

caxld(mx2,3)         array storing terms in the coefficient matrix for the  
fluid energy equation solution in subroutine ENERGY,  
and also for the one-dimensional energy rebalancing in  
subroutine RECIRC

gres(mx2)             array storing net energy in all nodes at each axial  
level

dtax(mx2)             array used in one-dimensional energy rebalancing for  
average temperature at each level

qcon(iu)              array storing summation of radiation terms in all  
channels of each assembly, for the solution of the  
fluid and rod energy equations

endr1                 variable defining ending address for common block  
rebal

## Distribution

<u>No. of Copies</u>		<u>No. of Copies</u>	
<b>OFFSITE</b>			
	Battelle Memorial Institute Technical Library Office of Nuclear Waste Isolation 505 King Avenue Columbus, Ohio 43201		T. L. Sanders Sandia National Laboratory P.O. Box 5800 Albuquerque, NM 87185
	A. Brownstein U.S. Department of Energy 1000 Independence Avenue Washington, D.C. 20585		M. L. Smith Virginia Power Company 5000 Dominion Blvd. Glen Allen, VA 23060
2	Electric Power Research Institute P.O. Box 10412 Palo Alto, CA 94303 Attn: R. W. Lambert R. L. Yang	2	U.S. Nuclear Regulatory Commission Office of Nuclear Materials Safety and Safeguards Mail Stop 6-H-3 Washington, D.C. 20555 Attn: F. C. Sturz M. B. Raddatz
	FLUOR Engineers Advanced Technology Division 333 Michelson Drive Irvine, CA 92730		<b>ONSITE</b>
	J. V. Massey Sierra Nuclear Corporation 5619 Scotts Valley Drive, #240 Scotts Valley, CA 95066	3	<b>DOE Richland Operations Office</b>
			D. C. Langstaff K8-50 D. E. Trader K8-50 J. W. Wiley K8-50
2	Office of Civilian Radioactive Waste Management U.S. Department of Energy Washington, D.C. 20585 Attn: W. J. Danker, RW-33 L. Stewart, RW-421 J. R. Williams, RW-421	1	<b>Westinghouse Hanford Company</b>
	J. Richardson WESTON Technical Support Team 955 L'Enfant Plaza SW Washington, D.C. 20024		D. H. Nyman B5-24
		29	<b>Pacific Northwest Laboratory</b>
			J. M. Creer K8-50 J. M. Cuta K7-15 C. W. Enderlin K7-15 M. A. McKinnon K8-34 T. E. Michener (14) K7-15 D. R. Reactor (2) K7-15 K. P. Recknagle K7-15 Z. I. Zntoniak K7-15 Information Release Office (7) K1-06