This document was prepared in conjunction with work accomplished under Contract No. DE-AC09-96SR18500 with the U. S. Department of Energy.

DISCLAIMER

WSRC-TR-2006-00123
Revision 0

Key Words:
Ion Exchange
Model
Radiolytic Heating
Natural Convection
Plutonium 238
Reillex Resin

Retention:
Permanent

Tracking No. 10560

UNCLASSIFIED
DOES NOT CONTAIN
UNCLASSIFIED CONTROLLED
NUCLEAR INFORMATION

ADC &
Reviewing
Official: James E Laurinat
(James E. Laurinat)

Date: 4-12-06

Revised Thermal Analysis of LANL Ion Exchange Column

April 11, 2006

Patent Status
This internal management report is being transmitted without DOE patent clearance, and no further dissemination or publication shall be made of the report without prior approval of the DOE patent counsel.

Westinghouse Savannah River Company
Aiken, SC 29802

Prepared by the U. S. Department of Energy under Contract DE-AC09-96SR18500

DISCLAIMER

Printed in the United States of America

Prepared For
U.S. Department of Energy

Revised Thermal Analysis of LANL Ion Exchange Column

By J. E. Laurinat

Issued: April 11, 2006

Approvals

Jose E Laurinat 4-11-06

J. E. Laurinat, Author, E&CPT/ATS Date

N. M. Askew, Technical Reviewer, E&CPT/ATS 4/11/06
Date

C. W. Gardner, Responsible Manager, E&CPT/ATS 4/11/06
Date

Alice M. Murray April 11, 2006

A. M. Murray, Responsible Manager, E&CPT/ATS Date

Table of Contents

1.0 Introduction

This document updates a previous calculation of the temperature distributions in a Los Alamos National Laboratory (LANL) ion exchange column.[1] LANL operates two laboratory-scale anion exchange columns, in series, to extract Pu-238 from nitric acid solutions. The Defense Nuclear Facilities Safety Board has requested an updated analysis to calculate maximum temperatures for higher resin loading capacities obtained with a new formulation of the Reillex HPQ anion exchange resin. The increased resin loading capacity will not exceed 118 g plutonium per L of resin bed.[2] Calculations were requested for normal operation of the resin bed at the minimum allowable solution feed rate of 30 mL/min and after an interruption of flow at the end of the feed stage, when one of the columns is fully loaded.

The object of the analysis is to demonstrate that the decay heat from the Pu-238 will not cause resin bed temperatures to increase to a level where the resin significantly degrades. At low temperatures, resin bed temperatures increase primarily due to decay heat. At ~70°C a Low Temperature Exotherm (LTE) resulting from the reaction between 8 – 12 M HNO_3 and the resin has been observed. The LTE has been attributed to an irreversible oxidation of pendant ethyl benzene groups at the termini of the resin polymer chains by nitric acid. The ethyl benzene groups are converted to benzoic acid moities.[3,4] The resin can be treated to permanently remove the LTE by heating a resin suspension in 8M HNO_3 for 30-45 minutes. No degredation of the resin performance is observed after the LTE removal treatment.[2] In fact, heating the resin in boiling (~115-120°C) 12 M HNO_3 for 3 hr displays thermal stability analogous to resin that has been treated to remove the LTE (see Reference 5, Figure 38).

The analysis is based on a previous study of the SRS Frames Waste Recovery (FWR) column, performed in support of the Pu-238 production campaign for NASA's Cassini mission.[6] In that study, temperature transients following an interruption of flow to the column were calculated. The transient calculations were terminated after the maximum resin bed temperature reached the Technical Standard of 60 °C, which was set to prevent significant resin degradation.

The LANL column differs from the FWR column in that it has a significantly smaller radius, 3.73 cm nominal versus approximately 28 cm. It follows that natural convection removes heat much more effectively from the LANL column, so that the column may reach thermal equilibrium. Consequently, the calculations for a flow interruption were extended until an approach to thermal equilibrium was observed.

The LANL ion exchange process also uses a different resin than was used in the FWR column. The LANL column uses Reillex HPQ™ resin, which is more resistant to attack by nitric acid than the Ionac 641™ resin used in the FWR column. Heat generation from the resin oxidation reaction with nitric acid is neglected in this analysis since LANL will be treating the resin to remove the LTE prior to loading the resin in the columns.

Calculations were performed using a finite difference computer code, which incorporates models for absorption and elution of plutonium and for forced and natural convection within the resin bed. Calculations for normal column operation during loading were performed using an initial temperature and a feed temperature equal to the ambient air temperature. The model for the normal flow calculations did not include natural convection within the resin bed. The no flow calculations were started with the temperature and concentration profiles at the end of the loading stage, when there would be a maximum amount of plutonium either adsorbed on the resin or in the feed solution in the column.

2.0 Summary of Results

Maximum resin bed temperatures were calculated for saturation loading of the entire bed. The calculated maximum temperatures are 40.2 °C at the minimum allowable loading flow rate of 30 mL/min and 105.5 °C for a postulated flow stoppage at the end of the load cycle. The maximum temperature for the normal flow case was reached 41 min after the start of loading, when the column became fully loaded, and the maximum temperature for a flow interruption was approached approximately 2 to 3 hr after flow was stopped. The maximum calculated bed temperature for a flow stoppage exceeds 70 °C, the temperature where the LTE of Reillex™ HPQ resin commences[3,4] and is below 110 °C, the approximate temperature at which the dry, nitric acid-treated resin displays a High Temperature Exotherm.[3,4,5]

3.0 Input Conditions for Heat Transfer Analysis

Table 1 lists the input parameters for the heat transfer analysis. As this table indicates, input conditions include the column dimensions, the resin bed physical characteristics, the feed flow rate, the feed concentration, the amount of plutonium fed to the column, and the provision for cooling the column, exposure to ambient air.

4.0 Models for Ion Exchange Chemistry

In the ion exchange process, plutonium is selectively adsorbed onto the resin from concentrated nitric acid solution during loading and washing and is eluted from the resin with dilute nitric acid. In the model, the adsorption and elution processes occur irreversibly during normal operation. During a flow interruption, however, the model assumes that plutonium adsorbs onto the resin when the acid solution between the resin beads is saturated with plutonium and is eluted when the acid solution is not saturated. Saturation conditions are calculated using measured equilibrium relationships for nitrated resins.

The following reaction occurs on the resin surfaces,[7] which are assumed to be evenly distributed throughout the resin beads.

$$R_2Pu(NO_3)_6(solid) \Leftrightarrow PuNO_3^{3+} + 3NO_3^- + 2RNO_3(solid)$$

where R represents a chemically active site on the resin substrate.

Equilibrium for this reversible reaction is governed by the solubility product[7]

$$K_1 = [PuNO_3^{3+}][NO_3^-]^3 \quad (1)$$

where K_1 is an equilibrium constant that depends on the resin type and the plutonium loading on the resin. The model uses a value of 0.26, which was measured for fully loaded Dowex-1 X-4 resin.[7] Although the equilibrium constant for the Dowex 1 X-4 resin may not equal that of the Reillex resin, the effect on the calculations for this study should not be significant, because the resin is assumed to be fully loaded with plutonium and to be in contact with a solution with uniform nitric acid and plutonium concentrations.

In solution, the plutonium ions complex with nitrate ions according to:[7]

$$Pu^{4+} + NO_3^- \Leftrightarrow PuNO_3^{3+}$$

The equilibrium constant for this reaction is given by[7]

$$K_2 = \frac{[PuNO_3^{3+}]}{[Pu^{4+}][NO_3^-]} = 2.9 \pm 0.6 \quad (2)$$

These equilibrium relations can be combined with the following stoichiometric equations to calculate the plutonium and nitrate concentrations in a saturated solution.

$$[NO_3]_{total} = [NO_3^-] + [PuNO_3^{3+}] \quad (3)$$

and

$$[Pu]_{total} = [Pu^{4+}] + [PuNO_3^{3+}] \quad (4)$$

For natural convection during a flow stoppage, the model solves these equations iteratively at each calculation node to calculate the saturation plutonium concentration based on the actual nitrate concentration and the saturation nitrate concentration based on the actual plutonium concentration. The calculated saturation values are then used to compute a rate of adsorption if the solution is supersaturated or desorption if the solution is unsaturated.

5.0 Models for Adsorption and Elution

The model assumes that the rates of adsorption and elution are controlled by diffusion of plutonium to and from the resin bead surfaces. Because the plutonium ionic species react rapidly at the resin surface, adsorption takes place at the outer surfaces of the resin beads before the plutonium can penetrate to the interior of the beads. Consequently, solid-phase, or particle, diffusion controls the rate of adsorption.[7] The particle diffusion model is analogous to a molecular diffusion model, with a particle diffusivity and the bulk adsorbed concentration replacing the ordinary diffusivity and the solution concentration.

A shrinking core diffusion approximation describes desorption of plutonium from the resin beads.[7] For elution during normal operation, the rate of desorption is modified by a factor that accounts for mixing of the dilute elution acid with the concentrated acid used to load the resin.[6] In addition, a pore diffusion model is used to calculate the rate of interchange of plutonium and nitrate between the interparticle and intraparticle, or pore, solutions. The pore diffusion model takes the same form as the particle diffusion model for plutonium adsorption, with a pore diffusivity replacing the particle diffusivity. This interchange rate is calculated for normal operation but not for flow stoppages. To simplify the calculations during a flow stoppage, the model assumes that the acid is well mixed both in the spaces between resin beads and in the resin bead pores. The concentrations of plutonium between the beads and in the bead pores are combined on a volume average basis at the start of the flow stoppage. Thereafter, both solutions are treated as if they are located outside the resin beads.

Reference 6 describes details of the adsorption and elution models, so the adsorption and elution rate equations are not listed in this report. It must be noted, however, that there is an error in the equations for the rate of plutonium desorption, G_a (Equations 33, A-34, A-36, and A-37 in Reference 6). Equation A-34 should read

$$1 - \frac{\bar{y}}{x_{max}} = \frac{\sqrt{c - \frac{x}{x_{max}}}}{\sqrt{\frac{x}{x_{max}}} + \sqrt{c - \frac{x}{x_{max}}}} \qquad (5)$$

With this change, Equations 33 and A-36 become

$$G_a\left(=-\frac{df}{dt}\right)=\frac{-cf_0\left(\frac{\sqrt{c-\frac{x}{x_{max}}}}{\sqrt{\frac{x}{x_{max}}}+\sqrt{c-\frac{x}{x_{max}}}}\right)}{\tau_{pore}\left(1-\frac{4}{5}\sqrt[3]{\frac{f_0}{f}}-\frac{1}{5}\left(\frac{f_0}{f}\right)^2\right)} \qquad (6)$$

Equation A-37 no longer is valid. Instead, as expected, the concentration gradient factor, $1-\frac{\bar{y}}{x_{max}}$, goes to zero as the saturation level of the acid, c, approaches that of the plutonium in solution, $\frac{x}{x_{max}}$.

Particle and pore diffusion coefficients control the rates of particle diffusion during loading and pore diffusion during elution, respectively. In the calculations for the FWR column, diffusivity values were selected to fit loading profiles and discharge transients for test elutions.[6] The particle diffusivity was set to 2 x 10^{-9} cm^2/sec, and the pore diffusivity was set to 1 x 10^{-6} cm^2/sec.

LANL has stated that their Reillex resin loads much more rapidly than the typical resins used at SRS. The rapid loading of the Reillex resin implies that both the particle and pore diffusivities are much greater than the values used for the FWR column analysis. A bounding value for both diffusivities is the effective pore diffusivity for plutonium (IV) nitrate in aqueous solutions. This bounding value assumes that the controlling resistance for both adsorption and elution is pore diffusion and that the resin bead pores are sufficiently large so that steric hindrance is not a factor. The formula for the pore diffusivity is

$$D_{pore}=\frac{\varepsilon_i}{1-\varepsilon_o}\frac{D_{mol}}{\tau} \qquad (7)$$

where the term $\frac{\varepsilon_i}{1-\varepsilon_o}$ represents a volumetric correction factor relating the pore volume to the total volume of the resin beads and τ is the tortuosity. In Reference 6, the pore volume fraction ε_i was estimated to be 0.132 and the interparticle volume fraction ε_o was estimated to be 0.326. A tortuosity factor of 2, which is typical for diffusion through either pores or unconsolidated solids,[8] is used. The molecular diffusivity for plutonium ions is approximately 1.25 x 10^{-5} cm^2/sec (see Equation 28, Section 7.0). These values give an effective diffusivity of 1.2 x 10^{-6} cm^2/sec, which agrees with the pore diffusivity of 1 x 10^{-6} cm^2/sec for the FWR column analysis. Preliminary calculations demonstrated that the use of this diffusivity for both the particle and pore diffusion coefficients resulted

in most of the plutonium being loaded onto the resin at the concentrations and flow rates specified by LANL.

6.0 Models for Bulk Mass and Heat Transfer

The previous section describes the model for mass transfer within the resin beads. Mass and heat transfer in the bulk solution between the resin beads are modeled using standard equations for convection and dispersion. Because flow is restricted to the interparticle solution, the convective mass transfer equations for plutonium and nitrate apply to the interparticle concentrations. The total concentrations are calculated by summing the concentrations in the interparticle solution, in the intraparticle, or pore, solution, and adsorbed on the particles:

$$C_{Pu} = x_{Pu} + y_{Pu} + f \tag{8}$$

$$C_{NO_3} = x_{NO_3} + y_{NO_3} \tag{9}$$

The convective mass transfer equations take the form

$$\frac{\partial x_{Pu}}{\partial t} + v_{zs}\frac{\partial x_{Pu}}{\partial z} + v_{rs}\frac{\partial x_{Pu}}{\partial r} = \left(D_{mol} + \alpha_z + D_{mix}\right)\frac{\partial^2 x_{Pu}}{\partial z^2} + \frac{\alpha_r}{r}\frac{\partial}{\partial r}\left(r\frac{\partial x_{Pu}}{\partial r}\right) - \frac{dy_{Pu}}{dt} - \frac{df}{dt} \tag{10}$$

and

$$\frac{\partial x_{NO_3}}{\partial t} + v_{zs}\frac{\partial x_{NO_3}}{\partial z} + v_{rs}\frac{\partial x_{NO_3}}{\partial r} = \left(D_{mol} + \alpha_z + D_{mix}\right)\frac{\partial^2 x_{NO_3}}{\partial z^2} + \frac{\alpha_r}{r}\frac{\partial}{\partial r}\left(r\frac{\partial x_{NO_3}}{\partial r}\right) - \frac{dy_{NO_3}}{dt} \tag{11}$$

The rates of change of the pore concentrations, $\frac{dy_{Pu}}{dt}$ and $\frac{dy_{NO_3}}{dt}$, are modeled using a pore diffusion model, and the rate of change of the adsorbed plutonium, $\frac{df}{dt}$, is modeled using a particle diffusion model. The pore and particle diffusion models are described in the previous section and in more detail in Reference 6.

The convective heat transfer equation includes a term for the decay heat, Q_{decay}. This equation is

$$\rho_{bed} c_{p,bed} \frac{\partial T}{\partial t} + \rho_f c_{p,f} v_{zs} \frac{\partial T}{\partial z} + \rho_f c_{p,f} v_{rs} \frac{\partial T}{\partial r}$$
$$= \left(k_{bed} + \rho_f c_{p,f} \alpha_z + \rho_f c_{p,f} D_{mix}\right) \frac{\partial^2 T}{\partial z^2} + \left(k_{bed} + \rho_f c_{p,f} \alpha_r\right) \frac{1}{r} \frac{\partial}{\partial r}\left(r \frac{\partial T}{\partial r}\right) + Q_{decay} C_{Pu} \quad (12)$$

The axial dispersion coefficients in the mass and heat transfer equations is given by[9]

$$\alpha_z = 2 d_p v_{zs} \quad (13)$$

The radial dispersion coefficient is about one-fifth as large.[10] Thus,

$$\alpha_r = 0.4 d_p v_{zs} \quad (14)$$

For normal operating conditions (loading, washing, and elution), the superficial axial velocity, v_{zs}, is specified, and the superficial radial velocity, v_{rs}, is assumed to be zero. For flow stoppages, these velocities result from natural convection. Natural convection velocities are ignored during loading washing, and elution. At high feed flow rates, the omission of natural convection does not have a significant effect. At lower feed flow rates, this omission is conservative in that it results in higher resin bed temperatures than would be predicted if natural convection were included.

Velocity components for natural convection flow are calculated by combining the Ergun equation with axial and radial component momentum equations, as suggested by Stewart and Dona.[11] The axial velocity for natural convection is modeled using a form of the Ergun equation for laminar flow, which is[12,13]

$$v_{zs} = \frac{d_p^2 \varepsilon_o^4}{200(1-\varepsilon_o)^2 \mu_L}\left(-\frac{\Delta P}{\Delta z}\right) \quad (15)$$

It may be noted that in Equation 15, $v_{zs} \propto \frac{\varepsilon_o^4}{(1-\varepsilon_o)^2}$, while in the Ergun equation, $v \propto \frac{\varepsilon_o^3}{(1-\varepsilon_o)^2}$. The extra void fraction multiplier is added because the buoyant force applies only to the fraction of the total volume occupied by the bulk solution and not to the resin beads.

The Ergun equation was originally derived for unidirectional flow. To model the two-dimensional flow in the resin bed, separate equations are used for the axial and radial velocity components. These equations take the form

$$v_{zs} = \frac{d_p^2 \varepsilon_o^4}{200(1-\varepsilon_o)^2 \mu}\left(-\frac{\partial P}{\partial z} + (\rho - \overline{\rho})g\right) \tag{16}$$

and

$$v_{rs} = \frac{d_p^2 \varepsilon_o^4}{200(1-\varepsilon_o)^2 \mu}\left(-\frac{\partial P}{\partial r}\right) \tag{17}$$

The buoyancy term in the axial velocity equation contains a reference density, $\overline{\rho}$. At equilibrium, the reference density must be set so that there is no net flow upward or downward at any bed level. This implies that the reference density must equal the average fluid density at a given bed level. In other words,

$$\overline{\rho} = \frac{2}{R^2}\int_{r=0}^{r=R} r\rho dr \tag{18}$$

The pressure gradient terms must be eliminated from Equations 17 and 18 to solve for the velocity components. This is accomplished by taking cross derivatives. The resulting equation is

$$\frac{\partial v_{rs}}{\partial z} - \frac{\partial v_{zs}}{\partial r} + \frac{d_p^2 \varepsilon_o^4 g}{200(1-\varepsilon_o)^2 \mu}\frac{\partial \rho}{\partial r} = 0 \tag{19}$$

The radial velocity is calculated by combining the continuity relation,

$$\frac{1}{r}\frac{\partial r v_{rs}}{\partial r} + \frac{\partial v_{zs}}{\partial z} = 0 \tag{20}$$

with Equation 19.

The vorticity equation is derived by eliminating the pressure gradient terms from the axial and radial component equations of motion.

To simplify the solution of these equations, the axial and radial velocities are expressed in terms of a stream function, defined by

$$v_{zs} = -\frac{1}{r}\frac{\partial r\psi}{\partial r} \tag{21}$$

and

$$v_{rs} = \frac{\partial \psi}{\partial z} \tag{22}$$

In terms of this stream function, the vorticity equation becomes:

$$\frac{\partial^2 \psi}{\partial z^2} + \frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial r\psi}{\partial r}\right) + \frac{d_p^2 \varepsilon_o^4 g}{200(1-\varepsilon_o)^2 \mu}\frac{\partial \rho}{\partial r} = 0 \tag{23}$$

This equation is solved with the boundary conditions

$$\psi = 0 \text{ at } r = 0 \text{ and } r = \frac{d}{2} \tag{24}$$

and

$$\frac{\partial^2 \psi}{\partial z^2} = 0 \text{ at } z = 0 \text{ and } z = L \tag{25}$$

The first set of conditions specifies that there is no flow across the column wall or the column centerline. The second set of conditions states that the radial velocity gradient is zero at the top and bottom of the resin bed.

The natural convection velocities defined by Equations 16, 17, and 18 do not account for convection due to cross-sectional average axial density gradients. At the start of a flow stoppage, the solution in the resin bed is unstably stratified, with the higher plutonium concentrations and therefore the higher densities at the top of the resin bed. Preliminary calculations with these velocities showed that the solution remained unstably stratified even as equilibrium approached. Actually, significant unstable axial density variations would cause heavier fluid from above to mix with lighter fluid from below in regularly spaced regions called Bénard cells. A detailed model of Bénard cell convection was not attempted. Instead, mixing due to axial density gradients is calculated using an analogy to bubble columns. A dispersion model is used to describe mixing in bubble columns. The dispersion coefficient for bubble columns is defined in terms of the circulation velocity induced by the bubble motion, v_c, and the column diameter:[14]

$$D_{mix} = 0.24 d v_c \tag{26}$$

The same correlation is used to calculate the dispersion due to buoyancy-induced mixing in the resin bed. The circulation velocity is set equal to half that given by the Ergun equation and the pressure gradient based on the difference between the average densities for different levels:

$$v_c = \frac{d_p^2 \varepsilon_o^4 \left(\overline{\rho}\big|_z - \overline{\rho}\big|_{z+\Delta z} \right) g}{400 (1 - \varepsilon_o)^2 \mu} \tag{27}$$

The one-half factor appears because mixing occurs in both directions and can therefore occupy only half of the cross-sectional flow area in either direction. The use of cross-sectional average densities accounts for the axial pressure gradients neglected by Equation 16. That equation already incorporates local density variations in the radial direction, so local density gradients are not included.

In the model, the dispersion coefficient is calculated using a weighted distribution of circulation velocities for density differences at different levels. The distribution is based on the mixing height for the dispersion coefficient, which is 0.8 times the column diameter.[14] If it is assumed that the mixing intensity decreases exponentially with axial distance, these values for the mixing height and velocity give the following expression for the local mixing velocity:

$$v_{eff} = v_c \exp\left(-2 \frac{|z - z_0|}{0.8d} \right) \tag{28}$$

where z_0 is the axial location at which the dispersion coefficient is evaluated. It may be confirmed that an integration of this equation with respect to axial displacement gives the mixing height.

7.0 Physical Property Correlations

Correlations for physical properties, including densities, heat capacities, thermal conductivities, and viscosities, are listed in Reference 6. An additional correlation for molecular diffusivity has been added for this analysis.

The only change from the correlations in Reference 1 is to the density correlation. The terms that account for density changes due to the presence of plutonium and nitrate in solution have been changed. No density data for plutonium nitrate solutions could be found, so the new density correlation uses a correlation based on apparent molar densities. This correlation takes the form[15]

$$\rho = \rho_{H_2O} + \frac{\left(M_{Pu(NO_3)_4} - \phi_{Pu(NO_3)_4} \rho_{H_2O} \right)}{1000} [Pu(NO_3)_4] + \frac{\left(M_{HNO_3} - \phi_{HNO_3} \rho_{H_2O} \right)}{1000} [HNO_3] \tag{29}$$

The terms $\phi_{Pu(NO_3)_4}$ and ϕ_{HNO_3} represent partial molar volume for deviations from ideal solution densities. The partial molar volume for HNO_3 is not calculated because the

effect of nitric acid concentration on the density has been measured. The partial molar volume for $Pu(NO_3)_4$ is given by the sum of the ionic partial molar densities:[15]

$$\phi_{Pu(NO_3)_4} = \phi_{Pu^{4+}} + \phi_{NO_3^-} \tag{30}$$

The partial molar density for NO_3^-, in turn, has been determined to be[16]

$$\phi_{NO_3^-} = 29.5 \tag{31}$$

The partial molar density for Pu^{4+} is given by the formula[15]

$$\phi_{Pu^{4+}} = 16 + 4.9r_x^3 - 20n_+ \tag{32}$$

where r_x, the ionic crystal radius for Pu^{4+}, is 0.96 Angstroms.[17]

Substitution of terms for the partial molar volume of $Pu(NO_3)_4$ and of a measured density coefficient for HNO_3[18] results in the following approximate equation for the overall density.

$$\rho = \rho_{H_2O} + 0.4277[Pu(NO_3)_4] + 0.031[HNO_3] \tag{33}$$

The HNO_3 concentration in this equation refers to the excess nitrate that is not ionically coupled with plutonium.

The molecular diffusivity is computed using the Nernst-Haskell equation, which takes the form[19]

$$D_{mol} = \frac{R_g T}{F^2} \frac{\frac{1}{n_+} + \frac{1}{n_-}}{\frac{1}{\lambda_+^0} + \frac{1}{\lambda_-^0}} \tag{34}$$

The cation and anion valences, n_+ and n_-, are 3 and 1, respectively. The value for λ_-^0 for nitrate is 71.4 $A/cm^2(V/cm)(g\text{-}equiv/cm^3)$. For plutonium the value of λ_+^0 for lanthanum, which is also a trivalent ion, 69.5 $A/cm^2(V/cm)(g\text{-}equiv/cm^3)$, was used.

8.0 Initial and Boundary Conditions

Solution of the mass and heat transfer equations requires initial conditions for concentrations, temperatures, and velocities and boundary conditions for velocity and

mass and heat transfer rates at the column walls and at the top and bottom of the column. At the start of loading, the plutonium concentrations in the resin bed are set at zero, and the temperature is set equal to the ambient air temperature of 25 °C. The starting concentrations and temperatures for flow stoppage analyses are taken from the end of the loading cycle. This is deemed to be conservative in that it results in the maximum amount of plutonium in the resin bed.

Boundary conditions at the ends of the column are fulfilled by having the flow mix with a head of feed solution at the top of the resin bed and discharge into a heel of solution at the bottom of the bed. The rates of heat and mass transfer in the liquid volumes above and below the resin bed are assumed to be much greater than the transfer rates within the bed. In the calculation, this assumption is implemented by setting temperatures and concentrations at the top and bottom bed surfaces equal to those in the head and heel volumes. The rates of heat and mass transfer into the head and the heel are then computed using one-sided gradients that include the temperatures and concentrations at the top two calculation nodes within the resin bed.

The calculation includes a continuity check to ensure that the area-averaged superficial head and heel velocities are equal to the feed velocity. At the side walls of the column, the radial superficial velocity is set equal to zero. There is no restriction on the axial velocity. In cases for which it is assumed that the column is cooled by ambient air, heat losses out the sides, top, and bottom of the column are calculated using relations for thermal radiation and natural and forced convection. When a water cooling jacket is specified, the surface temperature of the resin bed is set equal to the cooling water temperature, and it is assumed that the top and bottom are cooled by ambient air. The following section describes the heat transfer correlations for convection and radiation.

9.0 Heat Transfer Correlations

Heat transfer from the surface of the column is calculated by adding the thermal radiation heat flux to the maximum of the heat fluxes for natural convection and forced convection:

$$q_{tot} = q_{rad} + \max(q_{nc}, q_{fc}) \tag{35}$$

The thermal radiation heat flux is calculated using the equation:

$$q_{rad} = \sigma\varepsilon\left(T_{surf}^4 - T_{amb}^4\right) \tag{36}$$

A value of 0.6 is used for the surface emissivity ε. This emissivity is characteristic for steel surfaces and is a conservatively low estimate for glass surfaces.[20]

The natural convection heat flux is calculated from correlations for vertical surfaces and for horizontal surfaces facing upward. These correlations take the form

$$q_{nc} = h_{nc}\left(T_{surf} - T_{amb}\right) \tag{37}$$

The heat transfer coefficient is defined in terms of a Nusselt number, Nu:

$$h_{nc} = Nu\frac{k}{L} \tag{38}$$

For the side walls of the column, the Nusselt number is given by the correlation[21]

$$Nu = 0.59Ra_L^{0.25} \tag{39}$$

where Ra_L is the Rayleigh number based on the length of the column, defined by

$$Ra_L = \frac{g\rho_{amb}\left(\rho_{amb} - \rho_{surf}\right)L^3}{\mu^2}\frac{c_p\mu}{k} \tag{40}$$

For the top and bottom column surfaces, the Nusselt numbers are given by correlations for heat transfer from a warm surface facing upward and a warm surface facing downward, respectively:[21]

$$Nu = 0.54Ra_D^{0.25} \tag{41}$$

and

$$Nu = 0.27Ra_D^{0.25} \tag{42}$$

where Ra_D is the Rayleigh number based on the column diameter, defined by

$$Ra_D = \frac{g\rho_{amb}\left(\rho_{amb} - \rho_{surf}\right)d^3}{\mu^2}\frac{c_p\mu}{k} \tag{43}$$

The natural convection heat transfer coefficient is compared to the heat transfer coefficient for forced convection, with an assumed air velocity of 30 cm/s, or 1 ft/s. If the natural convection coefficient is smaller than the forced convection coefficient, the overall convective coefficient is set equal to that for forced convection. The correlation for forced convection takes the form:

$$h_{fc} = Nu_{fc}\frac{k}{d} \tag{44}$$

The Nusselt number for forced convection, Nu_{fc}, is defined in terms of a Reynolds number, Re_{fc}:[22]

$$Nu_{fc} = 0.683 Re_{fc}^{0.466} \quad (45)$$

where

$$Re_{fc} = \frac{d v_{fc} \rho_{amb}}{\mu} \quad (46)$$

10.0 Calculation Method

The finite difference heat and mass transfer calculations were performed on a Unix computer using Fortran codes. Separate codes were written to calculate transients during normal operation of the column (loading, washing, and elution) and following a stoppage of flow. Each code generates an output of maximum resin bed temperatures as a function of time as well as temperature, concentration, velocity, and modified stream function profiles at specified times. The output files containing temperature and concentration profiles from the transient calculation for normal column operation are used to initialize the conditions for the flow stoppage calculations. The modified stream function profiles were used to make contour plots that show the natural convection flow streamlines. These plots were used to check the validity of the natural convection velocity calculations. The modified stream function, $r\psi$, is calculated from Equations 23 through 25.

The calculations for normal conditions were performed using a donor cell algorithm that was semi-implicit in the dispersion terms and explicit in all other terms. An explicit donor cell algorithm was used to perform the calculations following a flow stoppage.

Code listings and sample input and output files appear in the Appendix.

11.0 Results of Calculations

Prior to calculating maximum temperature transients for the test conditions specified by LANL, preliminary analyses were performed to determine an optimum time step size and cell size for the finite difference method. Results were found to be insensitive to the time step size, provided it was small enough to guarantee numerical stability. Time step sizes of 0.02 s for operation at the specified feed flow rate of 30 mL/min, and 1.0 s for natural convection during flow stoppages were used.

Cell sizes were decreased until approximate convergence of temperature profiles and flow streamlines was obtained. It was determined that a grid of 48 axial cells by 24 radial cells gave an adequate approximation. To demonstrate that the discretization error for

these calculations is small, results also were calculated using a grid of 96 axial cells by 48 radial cells.

Figure 1 depicts the variation of the maximum temperature with elapsed time during loading to the resin bed capacity and after the postulated flow stoppage. The results in this figure demonstrate that the column is in thermal equilibrium at the end of the calculation (600 min.) It may be seen that the maximum bed temperature plateaus at the time loading is complete, after about 41 minutes. After the flow stoppage, the maximum temperature reaches 100 ºC in about 115 additional min.

The maximum temperature during loading was calculated to be 40.3 °C for the 48 by 24 node grid and 40.2 °C for the 96 by 48 node grid. After the flow stoppage, the equilibrium temperature was 105.4 °C using the 48 by 24 node grid and 105.5 °C using the 96 by 48 node grid. The comparison of the results for the two different node sizes demonstrates that the finite difference discretization error is negligible. The calculated maximum temperature has an uncertainty of approximately ±1 ºC due to uncertainties in the radiolytic heating rate for Pu-238 and other small uncertainties associated with thermal property uncertainties and finite element discretization.

Figure 2 shows the temperature profile in the resin bed at the end of the load cycle. As expected, for the fully loaded bed, the temperature reaches its maximum at the downstream end of the bed, which is the bottom. Figures 3 and 4 show the equilibrium temperature and flow profiles for the flow stoppage. The temperature profile in Figure 3 exhibits a peak at a point slight above the center of the resin bed. The displacement of the maximum temperature upward from the resin bed center results from the upward direction of the natural circulation flow at the bed center. (As Figure 4 indicates, the flow follows the profile of a thermosiphon, with upward velocities in the center, downward velocities along the cooled sides, radially inward flow along the bottom of the resin bed, and radially outward flow along the top of the bed.)

It may be noted that axial velocity is zero at the top and bottom of the resin bed. The zero axial velocity condition is a consequence of making an assumption that the liquid in the volumes above and below the bed are well-mixed, so that there is no density gradient driving flow into and out of the bed. The no axial flow boundary condition is conservative in that it artificially restricts heat transfer to the liquid volumes above and below the bed.

The two recirculation cells depicted in Figure 4 are driven by mixing with the liquid layers above and below the bed. At equilibrium, the solution within the resin bed is well-mixed. Consequently, the recirculation is primarily due to thermal density gradients. Warmer solution flows upward in the center of the resin bed, and cooler solution flow downward along the column walls. Similar recirculation patterns have been calculated for two-dimensional rectangular fluid containers with uniform volumetric heat sources.[23]

Because the column's length is much greater than its diameter, the calculated recirculation cells have a high axial/radial aspect ratio. The high aspect ratio of the

recirculation cells suggests that there might be more than one series of cells stacked one above another. The model does not test for the hydrodynamic stability of the recirculation pattern, so it does not account for this possibility. Nevertheless, the model should give a reasonably accurate estimate of heat transfer. The reasoning is that the magnitude of both the solution density gradients and the recirculation velocities, which are proportional to those gradients, would remain about the same. In fact, measured heat transfer rates for natural convection in volumes filled with porous glass exceed theoretical heat transfer rates at high aspect ratios.[24] These differences diminish as the convection velocities and hence the heat transfer rates increase. The porous glass heat transfer measurements seem to indicate that the assumption of a single vertical layer of recirculation cells may be conservative in that it might underestimate the actual rate of heat transfer.

12.0 Conclusions

Maximum resin bed temperatures were calculated for saturation loading of the entire bed. The calculated maximum temperatures are 40.2 °C at the minimum allowable loading flow rate of 30 mL/min and 105.5 °C for a postulated flow stoppage at the end of the load cycle. The maximum temperature for the normal flow case was reached 41 min after the start of loading, when the column became fully loaded, and the maximum temperature for a flow interruption was approached approximately 2 to 3 hr after flow was stopped. The maximum calculated bed temperature for a flow stoppage exceeds 70 °C, the temperature where the LTE of Reillex™ HPQ resin commences[3,4] and below 110 °C, the approximate temperature at which the dry, nitric acid-treated resin displays a High Temperature Exotherm.[3,4,5] It is important to note that Reillex™ HPQ resin shows no sign of decomposition after 3 hr of boiling in nitric acid and that the resin must be dry before undergoing the thermal excursion.

13.0 References

1. J. E. Laurinat and M. E. Pansoy-Hjelvik, “Thermal Analysis of LANL Ion Exchange Column (U),” WSRC-TR-99-0027, Rev. 0, May 23, 1999.

2. E. A. Kyser III, “Plutonium Loading onto Reillex HPQ Anion Exchange Resin,” WSRC-TR-2000-00372, Rev. 0, January 3, 2001.

3. J. E. Laurinat, W J. Crooks III, and E. A. Kyser III, “Impact of the Low-Temperature Reactivity of Reillex™ on Actinide Processing,” Solvent Extr. Ion Exch., 20(4&5), 525-542, 2002.

4. W. J. Crooks III, E. A. Kyser III, and S. R. Walter, “Low Temperature Exothermic Reaction of Reillex™ HPQ in Nitric Acid,” Solvent Extr. Ion Exch., 20(4&5), 543-559, 2002.

5. W. J. Crooks III, E. A. Kyser III, and S. R. Walter, "Qualification of Reillex™ HPQ Resin for Use in SRS Processes," WSRC-TR-99-00317, March 10, 2000.

6. J. E. Laurinat, "Self-Heating of Pu-238 Anion Exchange Column (U)," WSRC-TR-94-0166, Rev. 0, March 23, 1994.

7. J. L. Ryan and E. J. Wheelwright, "The Recovery, Purification, and Concentration of Plutonium by Anion Exchange in Nitric Acid (Declassified)," General Electric-Hanford Atomic Products Operation Report HW-55893, January 2, 1959.

8. C. N. Satterfield, Mass Transfer in Heterogeneous Catalysis, Colonial Press (1975), 35-36.

9. R. H. Perry and D. W. Green, eds., Perry's Chemical Engineers' Handbook, 6th ed., McGraw Hill, New York (1984), 16-27.

10. E. A. Ebach and R. R. White, "Mixing of Fluids Flowing through Beds of Packed Solids," A.I.Ch.E. J., 4(2), 161-169.

11. W. E. Stewart, Jr., and C. L. G. Dona, "Free Convection in a Heat Generating Porous Medium in a Finite Vertical Cylinder," J. Heat Transfer, 110(2), 1988, 517-520.

12. R. H. Perry and D. W. Green, eds., Perry's Chemical Engineers' Handbook, 6th ed., McGraw Hill, New York (1984), 5-54.

13. M. Leva, Fluidization, McGraw-Hill, New York (1959), 47.

14. J. B. Joshi, "Axial Mixing in Multiphase Contactors – A Unified Correlation," Trans. I. Chem. E., 58, 1980, 155-165.

15. A. M. Couture and K. J. Laidler, "The Partial Molar Volumes of Ions in Aqueous Solution: 1. Dependence on Charge and Radius," Can. J. Chem., 34, 1956, 1209-1216.

16. A. L. Horvath, Handbook of Aqueous Electrolyte Solutions: Physical Properties, Estimation, and Corelation Methods, Ellis Horwood Ltd., Chichester, England (1985), 131-133.

17. J. J. Katz, G. T. Seaborg, and L. R. Morss, The Chemistry of the Actinide Elements, 2nd ed., Volume 2, Chapman and Hall, London (1986), 1448.

18. C. M. Slansky, "Appendix I: Physical Properties of the Uranyl Nitrate, Aluminum Nitrate, Nitric Acid and Water System," in F. R. Bruce, J. M. Fletcher, and H. H. Hyman, Progress in Nuclear Energy, Series III: Process Chemistry, Vol. 2, Pergamon Press (New York (1958).

19. R. C. Reid, J. M. Prausnitz, and T. K. Sherwood, The Properties of Gases and Liquids, McGraw-Hill, New York (1977), 591.

20. R. H. Perry and D. W. Green, eds., Perry's Chemical Engineers' Handbook, 6th ed., McGraw Hill, New York (1984), 10-51.

21. R. H. Perry and D. W. Green, eds., Perry's Chemical Engineers' Handbook, 6th ed., McGraw Hill, New York (1984), 10-13.

22. F. Kreith and W. Z. Black, Basic Heat Transfer, Harper and Row, New York (1980), 251.

23. A. A. Emara and F. A. Kulacki, "A Numerical Investigation of Thermal Convection in a Heat-Generating Fluid Layer," J. Heat Transfer, 102(8), 1980, 531-537.

24. N. Seki, S. Fukusako, and H.Inaba, "Heat Transfer in a Cavity Packed with Fibrous Glass," J. Heat Transfer, 100(11), 1978, 748-750.

14.0 Nomenclature

c = relative saturation of nitric acid in the solution between the resin beads, dimensionless
c_p = heat capacity of air, cal/g/K
$c_{p,bed}$ = heat capacity of the resin bed, cal/g/K
$c_{p,f}$ = heat capacity of the feed solution, cal/g/K
C_{NO_3} = total bulk nitrate concentration, g/cm^3
C_{Pu} = total bulk plutonium concentration, g/cm^3
d = column diameter, cm
d_p = resin bead diameter, cm
D_{mix} = axial dispersion coefficient for buoyancy-induced or bubble-induced mixing, cm^2/sec
D_{mol} = molecular diffusivity of plutonium nitrate in aqueous solution, cm^2/sec
D_{pore} = pore diffusivity for diffusion of plutonium nitrate in resin beads, cm^2/sec
f = amount of plutonium adsorbed on resin, g/cm^3
f_0 = amount of plutonium adsorbed at beginning of elution, g/cm^3
F = Faraday's constant, 96,500 K/gmol
g = gravitational acceleration, 980 cm/sec^2
G_a = mass transfer rate for adsorption of plutonium, g/cm^3/sec
h_{nc} = natural convection heat transfer coefficient, cal/cm^2/sec/K
k = thermal conductivity of air, cal/cm/sec/K
k_{bed} = thermal conductivity of the resin bed, cal/cm/sec/K
L = column length, cm
M_i = molecular weight of ith compound, g/gmol
n_+ = cation valence for Nernst-Haskell diffusivity equation
n_- = anion valence for Nernst-Haskell diffusivity equation
Nu = Nusselt number
P = pressure, 10^{-1} Pascal
q_{nc} = natural convection heat flux, cal/cm^2/sec
q_{rad} = radiation heat flux, cal/cm^2/sec
q_{tot} = total heat flux, cal/cm^2/sec
Q_{decay} = volumetric rate of heat generation due to alpha decay, cal/cm^3/sec

r = radial distance from center of resin bed, cm

r_x = ionic radius, Angstroms

R = column radius, cm

= gas law constant, 8.314 J/gmole/K

Ra_D = the Rayleigh number based on the column diameter

Ra_L = the Rayleigh number based on the length of the column

T = temperature, K

T_{amb} = ambient temperature, K

T_{surf} = surface temperature, K

v_c = mixing velocity for interchange between adjacent solution layers, also, circulation velocity for bubble mixing, cm/sec

v_{eff} = effective velocity for mixing between solution layers at different levels within the resin bed, cm/sec

= circulation velocity of ambient air, cm/sec

v_{rs} = radial component of the superficial velocity of solution in the resin bed, cm/sec

v_{zs} = axial component of the superficial velocity of solution in the resin bed, cm/sec

x = plutonium concentration in bulk solution between resin beads, g/cm^3

x_{max} = maximum plutonium concentration in bulk solution between resin beads, g/cm^3

x_{NO_3} = nitrate concentration in the bulk solution between resin beads, g/cm^3

x_{Pu} = plutonium concentration in the bulk solution between resin beads, g/cm^3

$\bar{y}$ = time-averaged plutonium concentration in the resin bead pores at the bead surface, used in the calculation of elution rates through the bead, g/cm^3

y_{NO_3} = nitrate concentration in the solution in the resin bead pores, g/cm^3

y_{Pu} = plutonium concentration in the solution in the resin bead pores, g/cm^3

z = axial distance from bed entrance, cm

α_r = radial dispersion coefficient for mixing by the feed flow, cm^2/sec

α_z = axial dispersion coefficient for mixing by the feed flow, cm^2/sec

ε = surface emissivity, assumed to be 0.6 (for machined steel)[4]

ε_i = fraction of bulk volume occupied by liquid inside resin beads

ε_o = fraction of bulk volume occupied by liquid in spaces between resin beads

λ_+^0 = limiting cationic conductance for Nernst-Haskell diffusivity equation, A/cm^2 (V/cm)(g-equiv/cm^3)

λ_-^0 = limiting anionic conductance for Nernst-Haskell diffusivity equation, A/cm^2 (V/cm)(g-equiv/cm^3)

μ = viscosity of air, g/cm/sec

μ_L = viscosity of resin bed solution, g/cm/sec

ρ = solution mixture density, g/cm^3

$\overline{\rho}$ = average solution density at a given axial level, g/cm^3

ρ_{amb} = ambient air density, g/cm^3

ρ_{H_2O} = water density, g/cm^3

ρ_{surf} = air density at column surface, g/cm^3

σ = Boltzmann's constant, 1.356 x 10^{-12} cal/cm^2/sec/K^4

τ = tortuosity factor for pore diffusion

τ_{pore} = charcteristic time for pore diffusion, sec

ϕ_i = excess partial molar volume for ith anion, cation, or salt, cm^3/gmol

ψ = stream function, cm^2/sec

Table 1. Input Parameters for LANL Anion Exchange Column Heat Transfer Calculation

Input Parameter	Value
Initial temperature	25 °C
Heat transfer	natural convection and thermal radiation to air at 25 °C
Resin bed depth	34.1 cm
Resin bed diameter	3.73 cm
Resin bed volume	1.5 L
Height of liquid layer above resin bed	2.54 cm
Height of liquid layer below resin bed	2.54 cm
Resin bead diameter	0.04 cm (average for 30-60 mesh particles)
Pore diffusivity	1.2×10^{-6} cm^2/s
Particle diffusivity	1.2×10^{-6} cm^2/s
Interparticle void fraction	0.33
Volume fraction in resin bead pores before Pu absorption	0.132
Volume fraction in resin bead pores after Pu absorption	0.160
Pu feed concentration	44.1 g/L total Pu, 50 g/L PuO_2
Maximum bulk Pu absorption concentration	118 g/L total Pu
Plutonium enrichment	89% Pu-238
Radiolysis heat generation rate for Pu-238	0.13384 cal/gm Pu-238/s
Number of axial nodes	48
Number of radial nodes	24
Time increment for loading calculation	0.02 s
Time increment for natural convection calculation	1.0 s
Load time	41.0 min
Volume fraction HNO_3 in feed	0.274
Superficial loading velocity	0.0458 cm/s, equals 30 mL/min
Column Radius	3.81 cm nominal 3.73 cm measured by displacement
Length of Resin Bed	34.1 cm
Height of Feed Solution above Bed	2.54 cm
Depth of Discharge Solution below Bed	2.54 cm
Resin Bead Diameter	0.04 cm

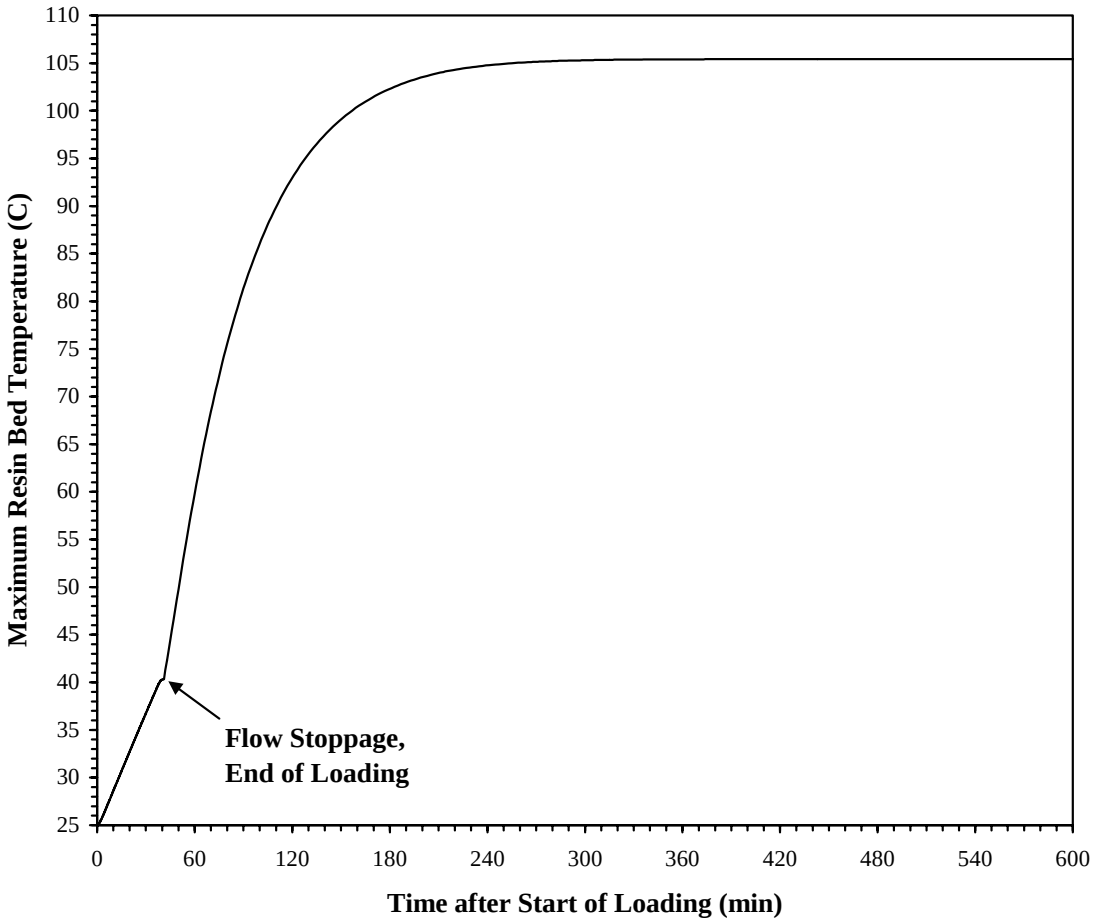

Figure 1. Maximum Resin Bed Temperatures during Saturation Loading with 50 g/L Pu as PuO_2, followed by a Flow Stoppage

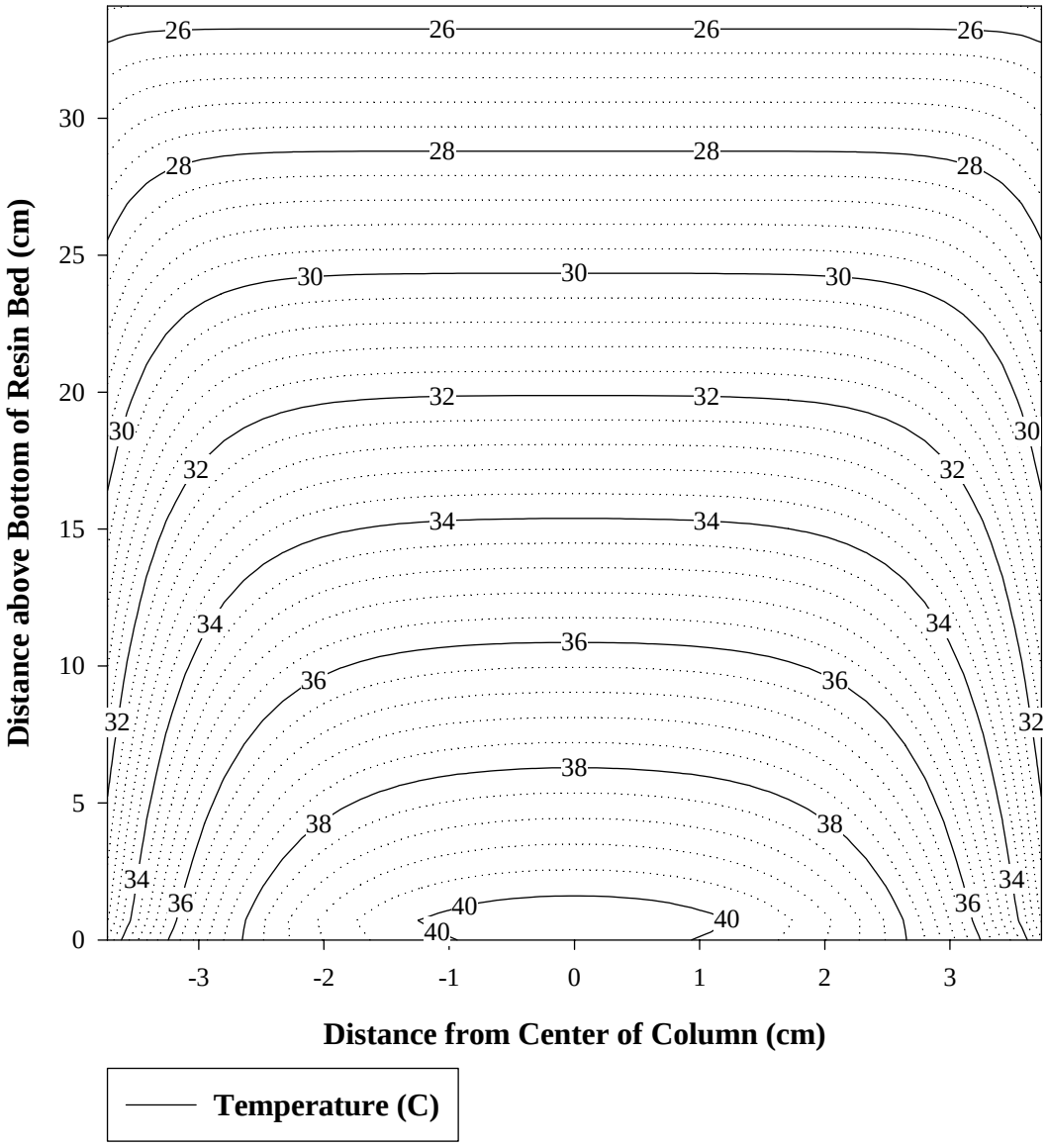

Figure 2. Temperature Profile for a Saturated Resin Bed, Loaded with 50 g/L Pu as PuO_2 at a Flow Rate of 30 mL/min

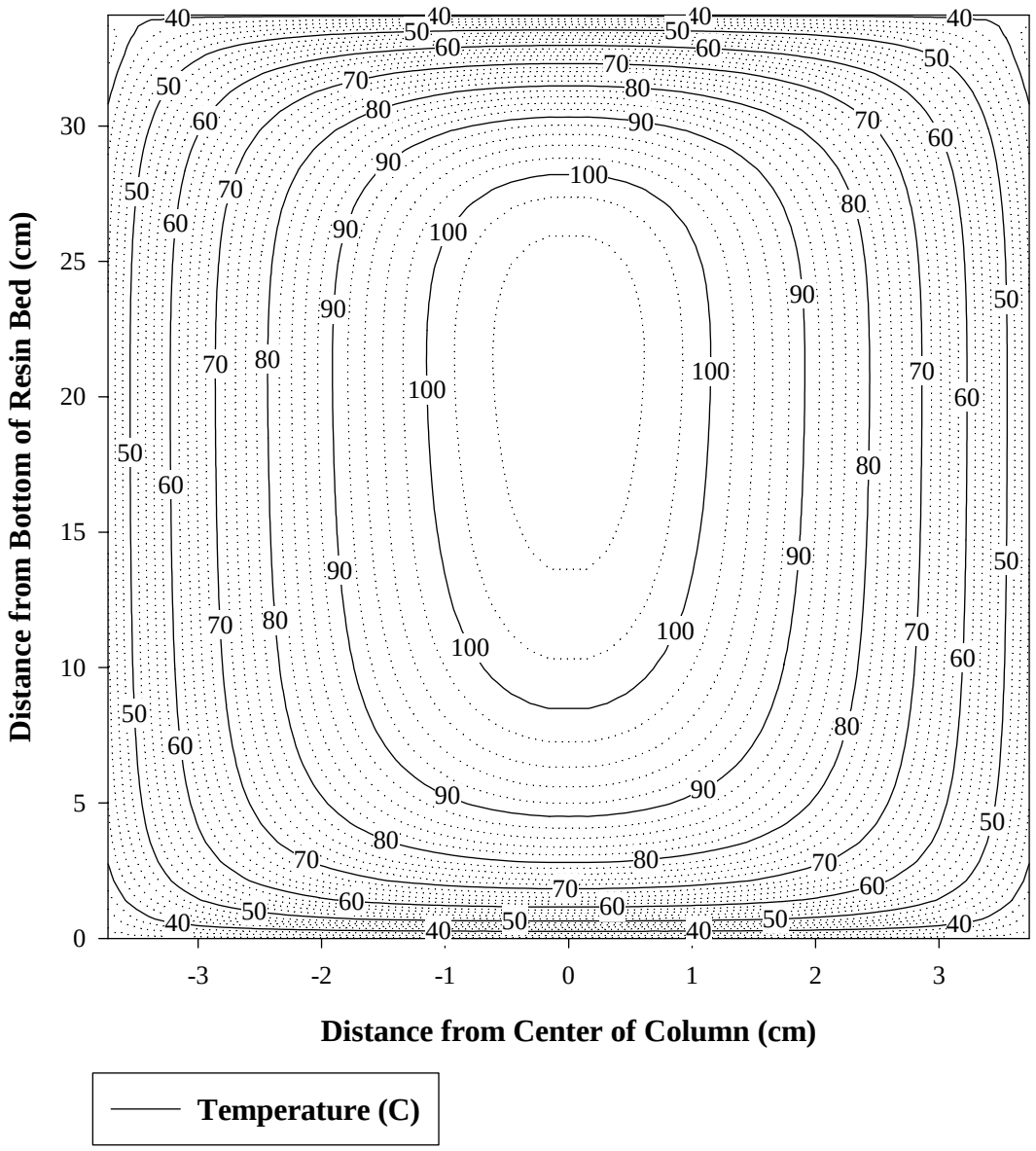

Figure 3. Equilibrium Temperature Profile for a Saturated Resin Bed during a Flow Stoppage

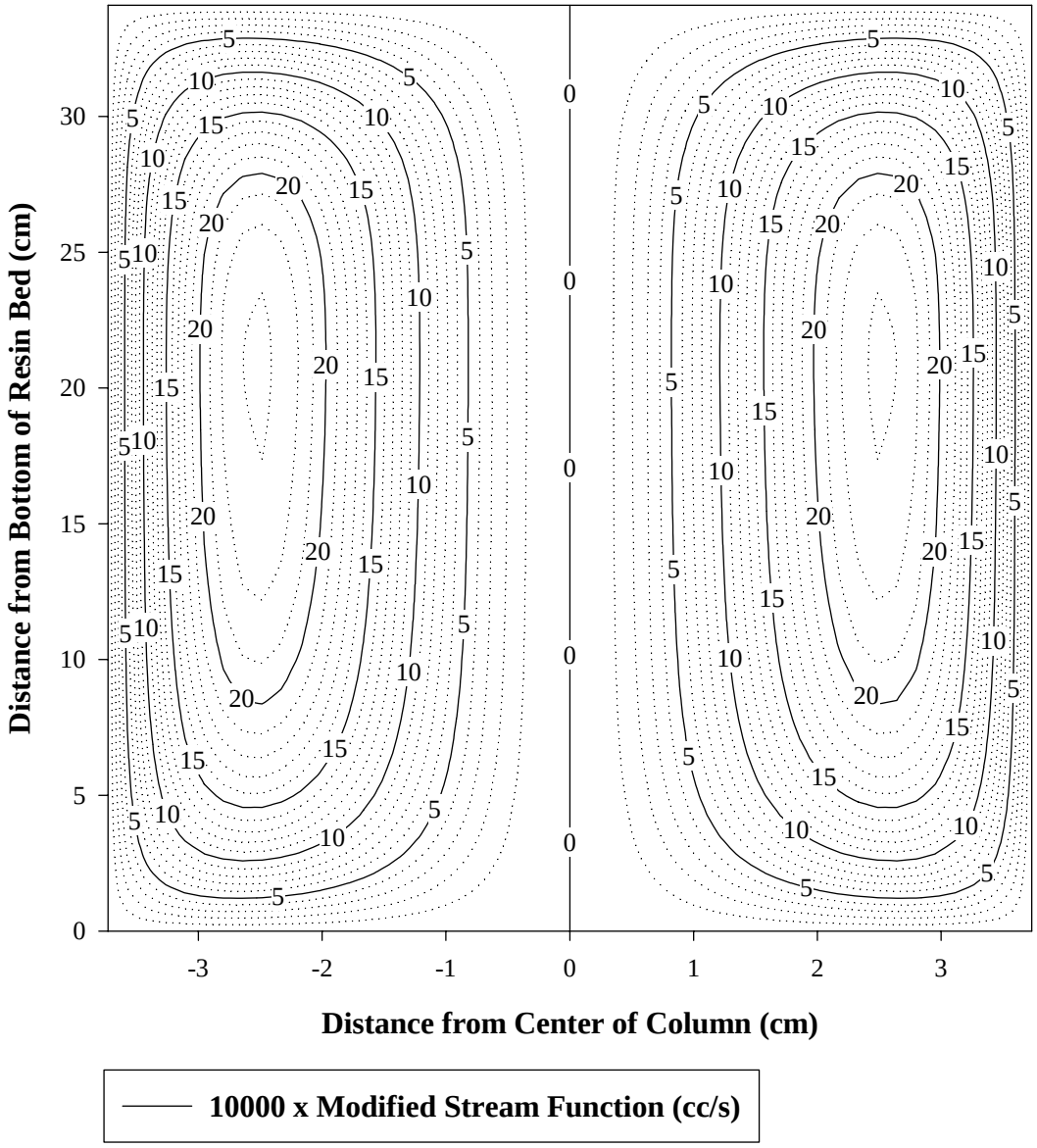

Figure 4. Equilibrium Natural Convection Flow Profile for a Saturated Resin Bed during a Flow Stoppage

Appendix: Source Code, Sample Input, and Sample Output Listings

This appendix contains the source code, sample input, and sample output listings for the LANL ion exchange column heat transfer analysis. Two sets of listings appear, one for calculations during normal operation, which includes loading, washing, and elution, and another assuming that the feed flow to the column has stopped. The calculations performed by these codes are described in the text. In addition, both codes include comments that give definitions of variables and outline the code structure. Sample input and output files also are listed. The output files that contain the maximum temperature transients appear; other output files are not shown.

The source code for normal operation is named FEDFLO.FOR, and the source code for a flow stoppage is called NATCON.FOR. Both codes were executed on the Unix computer srtc-comp2.

Source Code for Thermal Analysis during Normal Operation

```
c This program calculates the temperature transient during elution of an ion
c exchange column.  It includes the effects of thermal conduction to the column
c walls and forced convection heat and mass transfer.  The effect of plutonium
c and nitric acid on the bulk solution density is included.  The program solves
c finite difference energy balance equations with forward time differencing,
c donor cell differencing for convection terms, and centered space differencing
c for conduction terms.  A list of variable definitions follows.
c
c aa = one-dimensional matrix coefficient for heat transfer equation
c aa1 = two-dimensional matrix coefficient for heat transfer equation
c alpha = thermal diffusivity, cm2/s
c an1 = fraction of void volume that lies between resin beads
c area = cross-sectional area of resin bed, cm2
c bb = one-dimensional matrix coefficient for heat transfer equation
c bb1 = two-dimensional matrix coefficient for heat transfer equation
c br = nondimensional radial dispersion coefficient for heat transfer equation
c br1 = nondimensional radial dispersion coefficient for heat transfer equation,
c       due to area change with radius
c bz = nondimensional axial dispersion coefficient for heat transfer equation
c c = volume fraction acid in solution between resin beads
c caps = heat capacity of liquid, cal/g/C
c capsb = heat capacity of liquid in volume below resin bed, cal/g/C
c capst = heat capacity of liquid in volume above resin bed, cal/g/C
c cbot = volume fraction acid in liquid volume below resin bed
c cc = one-dimensional matrix coefficient for heat transfer equation
c cc1 = two-dimensional matrix coefficient for heat transfer equation
c cfeed = plutonium concentration in feed solution, g/cc
c cf30 = volume fraction occupied by HNO3 in entering feed solution
c cno3 = nitric acid concentration, gmole/lit
c cno3b = nitric acid concentration in liquid volume below resin bed, gmole/lit
c cno3t = nitric acid concentration in liquid volume above resin bed, gmole/lit
c cold = volume fraction acid in solution between resin beads at previous time
c        step
c cp = heat capacity, cal/g/C
c cpair = heat capacity of ambient air, cal/g/C
c cond = thermal conductivity
c c1feed = plutonium concentration in liquid volume above resin bed, g/cc
c den1 = density of pure HNO3, g/cc
c den1b = density of pure HNO3 in liquid volume below resin bed, g/cc
```

```
c den1t = density of pure HNO3 in liquid volume above resin bed, g/cc
c den2 = density of pure H2O, g/cc
c den2b = density of pure H2O in liquid volume below resin bed, g/cc
c den2t = density of pure H2O in liquid volume above resin bed, g/cc
c dens = liquid density, g/cc
c densb = liquid density in volume below resin bed, g/cc
c denst = liquid density in volume above resin bed, g/cc
c dfpor = pore diffusivity, cm2/s
c dfprt = particle diffusivity, cm2/s
c df = change in adsorbed plutonium concentration during feeding and elution,
c      g/cc
c dg = change in plutonium concentration in pore solution due to elution, g/cc
c dpart = resin bead diameter, cm
c dr = radial discretization step size, cm
c dt = time step, s
c dz = axial discretization step size, cm
c dy2 = change in plutonium concentration in solution between resin beads
c       during feeding and elution, g/cc
c emiss = surface emissivity for column
c f = plutonium loading on resin, g/cc
c fcx = weighting factor for elution of plutonium by acid dilution
c ffcx = weighting factor for elution of plutonium by acid dilution
c ffeed = maximum plutonium resin bed concentration, g/cc
c fold = plutonium loading on resin at previous time step, g/cc
c fx = total bulk plutonium concentration, g/cc
c grav = gravitational acceleration, cm/s2
c hconv = coefficient for natural convection heat transfer to sides of column,
c         also overall convective heat transfer coefficient, cal/cm2/s/C
c hconv2 = coefficient for natural convection heat transfer to top and bottom
c          of column, cal/cm2/s/C
c hconv3 = coefficient for forced convection heat transfer to column, assuming
c          an ambient air velocity of 30 cm/s (1 ft/s)
c hrad = radiation heat transfer coefficient, cal/cm2/s/C
c htc = heat transfer coefficient, cal/cm2/s/C
c htc2b = heat transfer coefficient for the liquid volume below the resin bed,
c         cal/cm2/s/C
c htc2t = heat transfer coefficient for the liquid volume above the resin bed,
c         cal/cm2/s/C
c iht = indicator for heat transfer to ambient air (iht=1) or to water-cooled
c       jacket (iht=2) (A heat transfer coefficient is calculated for heat
c       transfer to ambient air, and a surface temperature is specified for
c       cooling by a water jacket.)
c ind2 = indicator for stage of process (1 = loading, 2 = washing, 3 = elution)
c iss = indicator that switches between implicit radial, explicit axial (iss=0)
c       and explicit radial, implicit axial (iss=1) finite differencing
c ldown = depth of liquid in column below resin bed, cm
c lup = height of liquid in column above resin bed, cm
c length = column length, cm
c m = number of axial discretization nodes
c n = number of radial discretization nodes
c nu = Nusselt number for natural convection heat transfer to ambient air from
c      sides of column
c nu2 = Nusselt number for natural convection heat transfer to ambient air from
c       top and bottom of column
c nu3 = Nusselt number for forced convection heat transfer to ambient air,
c       assuming air velocity of 30 cm/s (1 ft/s)
c phi = updated temperature, K
c poszn = distance from top of resin bed, cm
c pr = Prandtl number for ambient air
c pua = total amount of plutonium adsorbed on the resin beads, g
c puf = total amount of plutonium in the column, g
c put = amount of plutonium fed to the column, g
c radius = column radius, cm
```

```
c rayl = Rayleigh number for natural convection heat transfer to ambient air
c        from sides of column
c rayl2 = Rayleigh number for natural convection heat transfer to ambient air
c         from top and bottom of column
c re = Reynolds number for forced convection heat transfer to column, assuming
c      an ambient air velocity of 30 cm/s (1 ft/s)
c rhcp = product of density and heat capacity for bulk resin bed, cal/cc/C
c rhcp2 = product of density and heat capacity for solution between
c         resin beads, cal/cc/C
c rhcp2b = product of density and heat capacity for liquid volume below resin
c          bed, cal/cc/C
c rhcp2t = product of density and heat capacity for liquid volume above resin
c          bed, cal/cc/C
c rho = bulk density, g/cc
c rhoair = density of air at resin column surface, g/cc
c rhoamb = density of ambient air, g/cc
c rr = right-hand side term of one-dimensional matrix heat transfer equation, K
c rr1 = right-hand side term of two-dimensional matrix heat transfer equation,
K
c rra = radial position of calculation node, cm
c rrn = radial position, cm
c rro = radial position, cm
c rrr = radial position, cm
c r1 = radial position, cm
c r2 = radial position, cm
c qdot = heat production rate due to plutonium loading, cal/g Pu-238/s
c sigma = Boltzmann's constant, cal/cm2/s/K4
c stem1 = temperature in the liquid volume above the resin bed, K
c stem1a = change in temperature in the liquid volume above the resin bed, K
c stem1c = temperature in the liquid volume above the resin bed, C
c stem2 = temperature in the liquid volume below the resin bed, K
c stem2a = change in temperature in the liquid volume below the resin bed, K
c stem2c = temperature in the liquid volume below the resin bed, C
c stp = total number of calculation time steps
c sumf = total loading of plutonium on resin, gm
c svz2 = integrated volumetric flow rate out column, cc/s
c tamb = ambient temperature, K
c tau = time constant for loading plutonium on resin, s
c tau2 = time constant for eluting plutonium from resin, s
c tcn1 = thermal conductivity for nitric acid, cal/cm/s/C
c tcn2 = thermal conductivity for water, cal/cm/s/C
c tcnm = thermal conductivity for bulk resin bed, cal/cm/s/C
c telut = time for elution, s
c temb = product of volumetric flow rate out of resin bed and temperature at
c        bottom of resin bed, K-cc/s
c temch = maximum temperature in column, C
c temc = temperature, C
c temp = temperature, K
c time = elapsed time after stage of process started, m
c time2 = elapsed time after feeding started, s
c tinte = time interval for printing maximum resin bed temperatures, s
c tinte2 = time for printing bed temperature profiles, s
c tload = time for loading, s
c trpt = time to print out temperature matrix, s
c twash = time for washing, s
c tzero = initial temperature, K
c tzero = temperature of solution entering column, K
c uu = vector containing solution to one-dimensional heat transfer equation, K
c vbot = volume of liquid below resin bed, cc
c vel2 = velocity of ambient air, cm/s
c vfi = volume fraction in resin bead pores, prior to adsorption of plutonium
c vfi2 = volume fraction in resin bead pores, after adsorption of plutonium
c vfi3 = volume fraction in resin bead pores, adjusted for actual amount of
```

```
c        plutonium adsorbed
c vfo = volume fraction occupied by flowing solution
c vis = liquid dynamic viscosity, g/cm/s
c visair = dynamic viscosity of ambient air, g/cm/s
c vmid = total volume of resin bed, cc
c vr = superficial radial velocity, cm/s
c vtop = volume of liquid above resin bed, cc
c vz = superficial axial velocity, cm/s
c x = fractional loading of plutonium on resin
c xg = elution rate constant, 1/s
c xkair = thermal conductivity of ambient air, cal/cm/s/C
c xx1 = mass fraction nitric acid in the solution between resin beads
c xx1b = mass fraction nitric acid in liquid volume below resin bed
c xx1t = mass fraction nitric acid in liquid volume above resin bed
c y = bulk plutonium concentration in pore solution, g/cc
c yold = bulk plutonium concentration in pore solution at previous step, g/cc
c y2 = bulk plutonium concentration in solution between resin beads, g/cc
c y2bot = plutonium concentration in liquid volume below resin bed, g/cc
c y2feed = plutonium concentration in solution entering resin bed, g/cc
c y2max = maximum bulk plutonium concentration in solution between resin beads
c         during elution, g/cc
c y2mx2 = maximum bulk plutonium concentration in solution between resin beads
c         during elapsed time during elution, g/cc
c y2old = bulk plutonium concentration in solution between resin beads at
c         previous time step, g/cc
c x = fractional loading of plutonium on resin
c xarea = cross-sectional area of resin bed, cm2
c xold = fractional loading of plutonium on resin at previous time step
c zden = area-average liquid density at a given level, g/cc
c zvz = area-average superficial axial velocity at a given level, cm/s
      implicit double precision (a-z)
      integer i,j,m,mm1,n,nn1,q,r,s,stp,iskip,is1,iss,jlev,
     1ind2
      character*10 ch1
      dimension temc(200,100),temp(200,100),phi(200,100)
      dimension dens(200,100),zden(200),htc(200)
      dimension vis(200,100),y2mx2(200)
      dimension area(100),den1(200,100),den2(200,100)
      dimension rhcp(200,100),rhcp2(200,100),caps(200,100)
      dimension bz(200,100),br(200,100),br1(200,100),tcnm(200,100)
      dimension aa(200),bb(200),cc(200)
      dimension uu(200),rr(200),rra(100)
      dimension aa1(200,100),bb1(200,100),cc1(200,100),rr1(200,100)
      character*1 tab
      dimension c(200),cold(200),f(200),fold(200),x(200),xold(200)
      dimension y(200),yold(200),y2(200),y2old(200),fx(200)
      common /elut1/c,cold,f,fold,x,xold,y,yold,y2,y2old,cfeed,
     1dt,tau,tau2,ffeed,an1,dpart,vz,dz,xg,vfi2,vfo,
     2y2feed,y2max,y2mx2,c1feed
      tab=char(9)
c Input initial resin bed temperature in °C.
      read (15,*) tzero
c Input ambient temperature in °C.
      read (15,*) tamb
c Input indicator for air heat transfer (iht=1) or water jacket heat transfer
c (iht=2).
      read (15,*) iht
c Input resin bed depth in centimeters.
      read (15,*) length
c Input height above resin bed in centimeters.
      read (15,*) lup
c Input depth below resin bed in centimeters.
      read (15,*) ldown
```

```
c Input column radius in centimeters.
      read (15,*) radius
c Input particle diameter in centimeters.
      read (15,*) dpart
c Input pore diffusivity in cm2/sec.
      read (15,*) dfpor
c Input particle diffusivity in cm2/sec.
      read (15,*) dfprt
c Input void fraction.
      read (15,*) vfo
c Input Pu feed concentration, gm/cc.
      read (15,*) cfeed
c Input maximum Pu resin bed concentration, gm/cc.
      read (15,*) ffeed
c Input maximum Pu concentration during elution, gm/cc.
      read (15,*) y2max
c Input radiolysis heat in cal/g Pu-238/sec.
      read (15,*) qdot
      pi=4.*atan(1.)
      rho=1.23
      cp=0.76
      tzero=tzero+273.0
      tamb=tamb+273.0
      grav=980.
      vfi=0.132
      vfi2=0.160
      an1=1./(vfo+vfi)
      pi=4.*atan(1.)
      tau=dpart**2/(4.*pi**2)/dfprt
      tau2=dpart**2/(24.*0.75*dfpor)
      xg=4.*pi**2*dfpor/dpart**2*(1.-vfo)
c Input # steps in axial, radial directions.
      read (15,*) m,n
c Input # of data points to skip between reading each concentration.
      read (15,*) iskip
      dz=length/m
      dr=radius/n
      vtop=pi*radius**2*lup
      vbot=pi*radius**2*ldown
      vmid=pi*radius**2*length
      m=m+1
      n=n+1
      mm1=m-1
      nn1=n-1
      area(1)=pi*(0.5/nn1*radius)**2
      area(n)=pi*(1.-((nn1-0.5)/nn1)**2)*radius**2
      rra(1)=0.
      i=2
      do while (i.le.nn1)
      r1=(i-1.5)/nn1*radius
      r2=(i-0.5)/nn1*radius
      area(i)=pi*(r2**2-r1**2)
      rra(i)=rra(i-1)+dr
      i=i+1
      enddo
      rra(n)=rra(nn1)+dr
      areat=pi*radius**2
      i=1
      sumf=0.
      dvol=pi*radius**2*dz
      do while (i.le.mm1)
      sumf=sumf+0.5*(f(i)+f(i+1))
      i=i+1
```

```
      enddo
      sumf=sumf*dvol
c Input time increment in seconds.
      read (15,*) dt
      dto=dt
c Input number of time steps.
      read (15,*) stp
c Input loading time in sec.
      read (15,*) tload
c Input washing time in sec.
      read (15,*) twash
c Input elution time in sec.
      read (15,*) telut
c Input time to print out temperature matrix in sec.
      read (15,*) trpt
c Input time interval for printing output.
      read (15,*) tinte
c Input time interval for printing maximum temperatures.
      read (15,*) tinte2
c Calculate material properties.
      stem1=tzero
      stem2=tzero
      cbot=0.
      y2bot=0.
      y2feed=0.
      c1feed=0.
      stem1c=stem1-273.
      stem2c=stem2-273.
      den1t=1.6603-1.9894e-3*stem1c
      den1b=1.6603-1.9894e-3*stem2c
      den2t=0.99683-1.3010e-4*stem1c-2.4358e-6*stem1c**2
      den2b=0.99683-1.3010e-4*stem2c-2.4358e-6*stem2c**2
      xx1t=den1t*cf30*(1.-c1feed)/(den1t*cf30*(1.-c1feed)+den2t
     1*(1.-cf30*(1.-c1feed)))
      xx1b=den1b*cf30*(1.-cbot)/(den1b*cf30*(1.-c1feed)+den2b
     1*(1.-cf30*(1.-cbot)))
      cno3t=(1.-c1feed)*cf30*den1t*1000./63.-4.*y2feed*1000./238.
      cno3b=(1.-cbot)*cf30*den1b*1000./63.-4.*y2bot*1000./238.
      if (cno3t.lt.0.) cno3t=0.
      if (cno3b.lt.0.) cno3b=0.
      denst=den2t+0.4277*y2feed*1000./238./vfo+0.031*cno3t
      densb=den2b+0.4277*y2bot*1000./238./vfo+0.031*cno3b
      capst=1.0104-(1.419-0.3*(stem1c-20.)/80.)*xx1t
     1+(2.005-0.6*(stem1c-20.)/80.)*xx1t**2
     2-(1.147-0.3*(stem1c-20.)/80.)*xx1t**3
      capsb=1.0104-(1.419-0.3*(stem2c-20.)/80.)*xx1b
     1+(2.005-0.6*(stem2c-20.)/80.)*xx1b**2
     2-(1.147-0.3*(stem2c-20.)/80.)*xx1b**3
      rhcp2t=denst*capst
      rhcp2b=densb*capsb
      i=1
      do while (i.le.n)
      j=1
      do while (j.le.m)
      cold(j)=0.
      c(j)=0.
      temc(j,i)=tzero-273.
      den1(j,i)=1.6603-1.9894e-3*temc(j,i)
      den2(j,i)=0.99683-1.3010e-4*temc(j,i)-2.4358e-6*temc(j,i)**2
      xx1=den1(j,i)*cf30*(1.-c(j))/(den1(j,i)*cf30*(1.-c(j))+den2(j,i)
     1*(1.-cf30*(1.-c(j))))
      cno3=(1.-c(j))*cf30*den1(j,i)*1000./63.-4.*y2(j)*1000.
     1/238./vfo
```

```
      if (cno3.lt.0.) cno3=0.
      dens(j,i)=den2(j,i)+0.4277*y2(j)*1000./238./vfo+0.031*cno3
      tcn1=6.1388e-4+1.3951e-6*temc(j,i)
      tcn2=1.3518e-3+2.7903e-6*temc(j,i)
      tcnm(j,i)=tcn1*xx1+tcn2*(1.-xx1)
      tcnm(j,i)=0.5*(5.e-4+tcnm(j,i))
      caps(j,i)=1.0104-(1.419-0.3*(temc(j,i)-20.)/80.)*xx1
     1+(2.005-0.6*(temc(j,i)-20.)/80.)*xx1**2
     2-(1.147-0.3*(temc(j,i)-20.)/80.)*xx1**3
      rhcp(j,i)=0.5*1.25*0.52+0.5*dens(j,i)*caps(j,i)
      rhcp2(j,i)=dens(j,i)*caps(j,i)
      alpha=tcnm(j,i)/rhcp(j,i)
      bz(j,i)=(alpha+2.*dpart*vz*rhcp2(j,i)/rhcp(j,i))*dt/dz**2
      j=j+1
      enddo
      i=i+1
      enddo
      j=1
      do while (j.le.m)
      alpha=tcnm(j,1)/rhcp(j,1)
      br(j,1)=(alpha+0.4*dpart*vz*rhcp2(j,1)/rhcp(j,1))*dt*4./dr**2
      br1(j,1)=0.
      j=j+1
      enddo
      i=2
      rrr=dr
      rro=0.
      do while (i.le.n)
      j=1
      do while (j.le.m)
      alpha=tcnm(j,i)/rhcp(j,i)
      rrn=rrr+dr
      br(j,i)=(alpha+0.4*dpart*vz*rhcp2(j,i)/rhcp(j,i))*dt/dr**2
      br1(j,i)=(alpha+0.4*dpart*vz*rhcp2(j,i)/rhcp(j,i))*dt
     1/2./dr/rra(i)
      j=j+1
      enddo
      rro=rrr
      rrr=rrr+dr
      i=i+1
      enddo
c Input initial volume fraction nitric acid.
      read (15,*) cf30
c Input velocity in cm/sec.
      read (15,*) vz
c Write input variables to output.
      write (11,100) length,radius,m,n
      write (11,121) dpart,vfo
      write (11,101) dt,htc,htc2
      write (11,110) sumf
  100 format ('L (CM) =',f7.3,2x,'R (CM) =',f7.3,2x,'AXIAL NODES =',
     1i3,2x,'RADIAL NODES =',i3)
  101 format ('DT (S) =',f8.4,2x,'HSID (CAL/CM2/S/°C) =',f9.6,
     12x,'HTOP (CAL/CM2/S/°C) =',f9.6)
  110 format ('PU CHARGE (GM) =',f11.4)
      write (12,*) 'TIME (MIN)',tab,'TMAX (°C)',tab,'VTOP (CC)',
     1tab,'VSGMAX (CM/S)'
  121 format ('PART DIAM (CM) =',f8.4,2x,'VOID FRACTION =',f7.4)
c Set all temperatures equal to uniform initial value.  Initialize plutonium
c and nitric acid concentrations.
      q=1
      do while (q.le.m)
      r=1
```

```
      do while (r.le.n)
      temp(q,r)=tzero
      tem1=tzero-273.
      den1(q,r)=1.6603-1.9894e-3*temc(q,r)
      den2(q,r)=0.99683-1.3010e-4*temc(q,r)-2.4358e-6*temc(q,r)**2
      r=r+1
      enddo
      fx(q)=0.
      fold(q)=0.
      yold(q)=0.
      y2old(q)=0.
      y2mx2(q)=0.
      cold(q)=0.
      q=q+1
      enddo
c Perform transient calculations.
      s=1
      time2=0.
      iss=1
      do while (time2.le.tload+twash+telut+1.)
      if (time2.le.tload) ind2=1
      if (time2.gt.tload.and.time2.le.tload+twash) ind2=2
      if (time2.gt.tload+twash) ind2=3
      if (iss.eq.0) then
c Switch between implicit radial, explicit axial and explicit radial, implicit
c axial calculations.
      iss=1
      else
      iss=0
      endif
c Calculate heat transfer coefficients
      sigma=1.356e-12
      emiss=0.6
      q=1
      do while (q.le.m)
      hrad=emiss*sigma*(temp(q,n)**3+temp(q,n)**2*tamb
     1+temp(q,n)*tamb**2+tamb**3)
      cpair=3.5*1.9872*29.
      xkair=0.0262/100*0.2389
      visair=0.00018
      pr=cpair*visair/xkair
      rhoamb=29./82.057/tamb
      rhoair=29./82.057/temp(q,n)
      rayl=980.*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
     1/visair**2*pr
      if (rayl.gt.0.) then
      nu=0.59*rayl**0.25
      endif
      if (nu.lt.1.) nu=1.
      hconv=nu*xkair/(length+lup+ldown)
      vel2=30.0
      re=2.*radius*vel2*rhoamb/visair
      nu3=0.683*re**0.466*pr**(1./3.)
      hconv3=xkair*nu3/2./radius
      if (hconv.lt.hconv3) hconv=hconv3
      htc(q)=hconv+hrad
      q=q+1
      enddo
      hrad=emiss*sigma*(stem1**3+stem1**2*tamb+stem1*tamb**2+tamb**3)
      rhoair=29./82.057/stem1
      rayl=980.*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
     1/visair**2*pr
      if (rayl.gt.0.) then
```

```
       nu=0.59*rayl**0.25
       endif
       if (nu.lt.1.) nu=1.
       hconv=nu*xkair/(length+lup+ldown)
       rayl2=980.*(2.*radius)**3*rhoamb*(rhoamb-rhoair)/visair**2*pr
       if (rayl2.gt.0.) then
       nu2=0.54*rayl2**0.25
       endif
       if (nu2.lt.1.) nu2=1.
       hconv2=nu2*xkair/2./radius
       hconv=(2.*radius*lup*hconv+radius**2*hconv2)/(2.*radius*lup
      1+radius**2)
       vel2=30.0
       re=2.*radius*vel2*rhoamb/visair
       nu3=0.683*re**0.466*pr**(1./3.)
       hconv3=xkair*nu3/2./radius
       if (hconv.lt.hconv3) hconv=hconv3
       htc2t=hrad+hconv
       hrad=emiss*sigma*(stem2**3+stem2**2*tamb+stem2*tamb**2+tamb**3)
       rhoair=29./82.057/stem2
       rayl=980.*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
      1/visair**2*pr
       if (rayl.gt.0.) then
       nu=0.59*rayl**0.25
       endif
       if (nu.lt.1.) nu=1.
       hconv=nu*xkair/(length+lup+ldown)
       rayl2=980.*(2.*radius)**3*rhoamb*(rhoamb-rhoair)/visair**2*pr
       if (rayl2.gt.0.) then
       nu2=0.27*rayl2**0.25
       endif
       if (nu2.lt.1.) nu2=1.
       hconv2=nu2*xkair/2./radius
       hconv=(2.*radius*ldown*hconv+radius**2*hconv2)/(2.*radius*ldown
      1+radius**2)
       vel2=30.0
       re=2.*radius*vel2*rhoamb/visair
       nu3=0.683*re**0.466*pr**(1./3.)
       hconv3=xkair*nu3/2./radius
       if (hconv.lt.hconv3) hconv=hconv3
       htc2b=hrad+hconv
c Calculate temperature changes in the spaces above and below the resin bed.
       stem1a=y2feed*0.89*qdot*dt/rhcp2t
       stem2a=y2bot*0.89*qdot*dt/rhcp2b
       i=1
       do while (i.le.n)
       stem1a=stem1a+bz(1,i)*dz/vtop*area(i)*(temp(2,i)-temp(1,i))
       stem2a=stem2a+bz(m,i)*dz/vbot*area(i)*(temp(mm1,i)-temp(m,i))
       stem2a=stem2a-vz*dt/vbot*area(i)*(stem2-temp(m,i))
       i=i+1
       enddo
       stem1a=stem1a-vz*dt*areat/vtop*(stem1-tzero)
       stem1a=stem1a+htc2t*dt/rhcp(1,1)/(vtop/(areat+2.*pi*rra(n)*lup))
      1*(tamb-stem1)
       stem2a=stem2a+htc2b*dt/rhcp(1,1)/(vbot/
      1(areat+2.*pi*rra(n)*ldown))*(tamb-stem2)
c Calculate concentration changes in the space above the resin bed.
       y2feed=y2feed+2.*dpart*vz*dt/dz/vtop*areat/vfo
      1*(y2old(2)-y2old(1))
       if (ind2.eq.1) then
       y2feed=y2feed+vz*areat/vtop*dt*(cfeed-y2feed)
       else
       y2feed=y2feed-vz*areat/vtop*dt*y2feed
```

```
      endif
      if (ind2.eq.3) then
      c1feed=c1feed+2.*dpart*vz*dt/dz/vtop*areat*(cold(2)-cold(1))
      c1feed=c1feed+vz*areat/vtop*dt*(1.-c1feed)
      endif
c Calculate temperature and concentration changes at top of column.
      phi(1,1)=temp(1,1)+fx(1)*0.89*qdot*dt/rhcp(1,1)
      if (iss.eq.0) then
      rr1(1,1)=phi(1,1)+bz(1,1)*(stem1-temp(1,1))
     1+bz(2,1)*(temp(2,1)-temp(1,1))
      aa1(1,1)=0.
      bb1(1,1)=1.+br(1,1)
      cc1(1,1)=-br(1,1)
      else
      rr1(1,1)=phi(1,1)+br(1,1)*(temp(1,2)-temp(1,1))+bz(1,1)*stem1
      aa1(1,1)=0.
      bb1(1,1)=1.+bz(1,1)+bz(2,1)
      cc1(1,1)=-bz(2,1)
      endif
      rr1(1,1)=rr1(1,1)-vz*dt/dz*(temp(1,1)-stem1)
      if (ind2.eq.1.or.ind2.eq.2) then
      call feeder (1,m,ind2)
      else
      call elut (1,m)
      endif
      phi(1,n)=temp(1,n)+fx(1)*0.89*qdot*dt/rhcp(1,n)
      if (iss.eq.0) then
      rr1(1,n)=phi(1,n)+bz(1,n)*(stem1-temp(1,n))
     1+bz(2,n)*(temp(2,n)-temp(1,n))+htc(1)*dt/rhcp(1,n)
     2*(1./dr+1./2./rra(n))*tamb
      aa1(1,n)=-br(1,nn1)+br1(1,n)
      bb1(1,n)=1.+br(1,nn1)-br1(1,n)+htc(1)*dt/rhcp(1,n)
     1*(1./dr+1./2./rra(n))
      cc1(1,n)=0.
      else
      rr1(1,n)=phi(1,n)-br(1,nn1)*(temp(1,n)-temp(1,nn1))
     1+br1(1,n)*(temp(1,n)-temp(1,nn1))+htc(1)*dt/rhcp(1,n)
     2*(1./dr+1./2./rra(n))*(tamb-temp(1,n))+bz(1,n)*stem1
      aa1(1,n)=0.
      bb1(1,n)=1.+bz(1,n)+bz(2,n)
      cc1(1,n)=-bz(2,n)
      endif
      rr1(1,n)=rr1(1,n)-vz*dt/dz*(temp(1,n)-stem1)
      r=2
      do while (r.le.nn1)
      phi(1,r)=temp(1,r)+fx(1)*0.89*qdot*dt/rhcp(1,r)
      if (iss.eq.0) then
      rr1(1,r)=phi(1,r)+bz(1,r)*(stem1-temp(1,r))
     1+bz(2,r)*(temp(2,r)-temp(1,r))
      aa1(1,r)=-br(1,r-1)+br1(1,r)
      bb1(1,r)=1.+br(1,r)+br(1,r-1)
      cc1(1,r)=-br(1,r)-br1(1,r)
      else
      rr1(1,r)=phi(1,r)-br(1,r-1)*(temp(1,r)-temp(1,r-1))
     1+br(1,r)*(temp(1,r+1)-temp(1,r-1))+br1(1,r)
     2*(temp(1,r)-temp(1,r-1))+bz(1,r)*stem1
      aa1(1,r)=0.
      bb1(1,r)=1.+bz(1,r)+bz(2,r)
      cc1(1,r)=-bz(2,r)
      endif
      rr1(1,r)=rr1(1,r)-vz*dt/dz*(temp(1,r)-stem1)
      r=r+1
      enddo
```

```
c Calculate temperature and concentration changes in middle of column.
      q=2
      do while (q.le.mm1)
      phi(q,1)=temp(q,1)+fx(q)*0.89*qdot*dt/rhcp(q,1)
      if (iss.eq.0) then
      rr1(q,1)=phi(q,1)+bz(q+1,1)*(temp(q+1,1)-temp(q,1))
     1+bz(q,1)*(temp(q-1,1)-temp(q,1))
      aa1(q,1)=0.
      bb1(q,1)=1.+br(q,1)
      cc1(q,1)=-br(q,1)
      else
      rr1(q,1)=phi(q,1)+br(q,1)*(temp(q,2)-temp(q,1))
      aa1(q,1)=-bz(q,1)
      bb1(q,1)=1.+bz(q,1)+bz(q+1,1)
      cc1(q,1)=-bz(q+1,1)
      endif
      rr1(q,1)=rr1(q,1)-vz*dt/dz*(temp(q,1)-temp(q-1,1))
      phi(q,n)=temp(q,n)+fx(q)*0.89*qdot*dt/rhcp(q,n)
      if (iss.eq.0) then
      rr1(q,n)=phi(q,n)+bz(q+1,n)*(temp(q+1,n)-temp(q,n))
     1+bz(q,n)*(temp(q-1,n)-temp(q,n))+htc(q)*dt/rhcp(q,n)
     2*(1./dr+1./2./rra(n))*tamb
      aa1(q,n)=-br(q,nn1)+br1(q,n)
      bb1(q,n)=1.+br(q,nn1)-br1(q,n)+htc(q)*dt/rhcp(q,n)
     1*(1./dr+1./2./rra(n))
      cc1(q,n)=0.
      else
      rr1(q,n)=phi(q,n)-br(q,nn1)*(temp(q,n)-temp(q,nn1))
     1+br1(q,n)*(temp(q,n)-temp(q,nn1))+htc(q)*dt/rhcp(q,n)
     2*(1./dr+1./2./rra(n))*(tamb-temp(q,n))
      aa1(q,n)=-bz(q,n)
      bb1(q,n)=1.+bz(q,n)+bz(q+1,n)
      cc1(q,n)=-bz(q+1,n)
      endif
      rr1(q,n)=rr1(q,n)-vz*dt/dz*(temp(q,n)-temp(q-1,n))
      q=q+1
      enddo
      q=2
      do while (q.le.mm1)
      r=2
      do while (r.le.nn1)
      phi(q,r)=temp(q,r)+fx(q)*0.89*qdot*dt/rhcp(q,r)
      if (iss.eq.0) then
      rr1(q,r)=phi(q,r)+bz(q+1,r)*(temp(q+1,r)-temp(q,r))
     1+bz(q,r)*(temp(q-1,r)-temp(q,r))
      aa1(q,r)=-br(q,r-1)+br1(q,r)
      bb1(q,r)=1.+br(q,r)+br(q,r-1)
      cc1(q,r)=-br(q,r)-br1(q,r)
      else
      rr1(q,r)=phi(q,r)-br(q,r-1)*(temp(q,r)-temp(q,r-1))
     1+br(q,r)*(temp(q,r+1)-temp(q,r))+br1(q,r)
     2*(temp(q,r+1)-temp(q,r-1))
      aa1(q,r)=-bz(q,r)
      bb1(q,r)=1.+bz(q,r)+bz(q+1,r)
      cc1(q,r)=-bz(q+1,r)
      endif
      rr1(q,r)=rr1(q,r)-vz*dt/dz*(temp(q,r)-temp(q-1,r))
      r=r+1
      enddo
      if (ind2.eq.1.or.ind2.eq.2) then
      call feeder (q,m,ind2)
      else
      call elut (q,m)
```

```
      endif
      q=q+1
      enddo
c Calculate temperature and concentration changes at bottom of column.
      phi(m,1)=temp(m,1)+fx(m)*0.89*qdot*dt/rhcp(m,1)
      if (iss.eq.0) then
      rr1(m,1)=phi(m,1)+bz(m,1)*(stem2-temp(m,1))
     1+bz(m,1)*(temp(mm1,1)-temp(m,1))
      aa1(m,1)=0.
      bb1(m,1)=1.+br(m,1)
      cc1(m,1)=-br(m,1)
      else
      rr1(m,1)=phi(m,1)+br(m,1)*(temp(m,2)-temp(m,1))+bz(m,1)*stem2
      aa1(m,1)=-bz(m,1)
      bb1(m,1)=1.+2.*bz(m,1)
      cc1(m,1)=0.
      endif
      rr1(m,1)=rr1(m,1)-vz*dt/dz*(temp(m,1)-temp(mm1,1))
      phi(m,n)=temp(m,n)+fx(m)*0.89*qdot*dt/rhcp(m,n)
      if (iss.eq.0) then
      rr1(m,n)=phi(m,n)+bz(m,n)*(stem2-temp(m,n))
     1+bz(m,n)*(temp(mm1,n)-temp(m,n))+htc(m)*dt/rhcp(m,n)
     2*(1./dr+1./2./rra(n))*tamb
      aa1(m,n)=-br(m,nn1)+br1(m,n)
      bb1(m,n)=1.+br(m,nn1)-br1(m,n)+htc(m)*dt/rhcp(m,n)
     1*(1./dr+1./2./rra(n))
      cc1(m,n)=0.
      else
      rr1(m,n)=phi(m,n)-br(m,nn1)*(temp(m,n)-temp(m,nn1))
     1+br1(m,n)*(temp(m,n)-temp(m,nn1))+htc(m)*dt/rhcp(m,n)
     2*(1./dr+1./2./rra(n))*(tamb-temp(m,n))+bz(m,n)*stem2
      aa1(m,n)=-bz(m,n)
      bb1(m,n)=1.+2.*bz(m,n)
      cc1(m,n)=0.
      endif
      rr1(m,n)=rr1(m,n)-vz*dt/dz*(temp(m,n)-temp(mm1,n))
      r=2
      do while (r.le.nn1)
      phi(m,r)=temp(m,r)+fx(m)*0.89*qdot*dt/rhcp(m,r)
      if (iss.eq.0) then
      rr1(m,r)=phi(m,r)+bz(m,r)*(stem2-temp(m,r))
     1+bz(m,r)*(temp(mm1,r)-temp(m,r))
      aa1(m,r)=-br(m,r-1)+br1(m,r)
      bb1(m,r)=1.+br(m,r)+br(m,r-1)
      cc1(m,r)=-br(m,r)-br1(m,r)
      else
      rr1(m,r)=phi(m,r)-br(m,r-1)*(temp(m,r)-temp(m,r-1))
     1+br(m,r)*(temp(m,r+1)-temp(m,r))
     2+br1(m,r)*(temp(m,r+1)-temp(m,r-1))+bz(m,r)*stem2
      aa1(m,r)=-bz(m,r)
      bb1(m,r)=1.+2.*bz(m,r)
      cc1(m,r)=0.
      endif
      rr1(m,r)=rr1(m,r)-vz*dt/dz*(temp(m,r)-temp(mm1,r))
      r=r+1
      enddo
      if (ind2.eq.1.or.ind2.eq.2) then
      call feeder (m,m,ind2)
      else
      call elut (m,m)
      endif
      if (iss.eq.0) then
      j=1
```

```
      do while (j.le.m)
      i=1
      do while (i.le.n)
      aa(i)=aa1(j,i)
      bb(i)=bb1(j,i)
      cc(i)=cc1(j,i)
      rr(i)=rr1(j,i)
      i=i+1
      enddo
      call tridag (aa,bb,cc,rr,uu,n)
      i=1
      do while (i.le.n)
      phi(j,i)=uu(i)
      i=i+1
      enddo
      j=j+1
      enddo
      else
      i=1
      do while (i.le.n)
      j=1
      do while (j.le.m)
      aa(j)=aa1(j,i)
      bb(j)=bb1(j,i)
      cc(j)=cc1(j,i)
      rr(j)=rr1(j,i)
      j=j+1
      enddo
      call tridag (aa,bb,cc,rr,uu,m)
      j=1
      do while (j.le.m)
      phi(j,i)=uu(j)
      j=j+1
      enddo
      i=i+1
      enddo
      endif
c Calculate concentration changes in the liquid volume below the resin bed.
      y2bot=y2bot+2.*dpart*vz*dt/dz/vbot*areat/vfo
     1*(y2old(mm1)-y2old(m))
      y2bot=y2bot+vz*areat/vbot*(y2old(m)/vfo-y2bot)
      cbot=cbot+vz*areat/vbot*(cold(m)-cbot)
c Update temperatures and concentrations.  Calculate densities and viscosities.
      stem1=stem1+stem1a
      stem2=stem2+stem2a
      j=1
      do while (j.le.m)
      yold(j)=y(j)
      y2old(j)=y2(j)
      xold(j)=x(j)
      cold(j)=c(j)
      if (ind2.eq.1.or.ind2.eq.2) fold(j)=f(j)
      if (ind2.eq.1.or.ind2.eq.2) fx(j)=f(j)+y2(j)
      if (ind2.eq.3) fx(j)=f(j)+y2(j)+y(j)
      j=j+1
      enddo
      stem1c=stem1-273.
      stem2c=stem2-273.
      den1t=1.6603-1.9894e-3*stem1c
      den1b=1.6603-1.9894e-3*stem2c
      den2t=0.99683-1.3010e-4*stem1c-2.4358e-6*stem1c**2
      den2b=0.99683-1.3010e-4*stem2c-2.4358e-6*stem2c**2
      xx1t=den1t*cf30*(1.-c1feed)/(den1t*cf30*(1.-c1feed)+den2t
```

```
1*(1.-cf30*(1.-c1feed)))
 xx1b=den1b*cf30*(1.-cbot)/(den1b*cf30*(1.-c1feed)+den2b
1*(1.-cf30*(1.-cbot)))
 cno3t=(1.-c1feed)*cf30*den1t*1000./63.-4.*y2feed*1000./238.
 cno3b=(1.-cbot)*cf30*den1b*1000./63.-4.*y2bot*1000./238.
 if (cno3t.lt.0.) cno3t=0.
 if (cno3b.lt.0.) cno3b=0.
 denst=den2t+0.4277*y2feed*1000./238./vfo+0.031*cno3t
 densb=den2b+0.4277*y2bot*1000./238./vfo+0.031*cno3b
 capst=1.0104-(1.419-0.3*(stem1c-20.)/80.)*xx1t
1+(2.005-0.6*(stem1c-20.)/80.)*xx1t**2
2-(1.147-0.3*(stem1c-20.)/80.)*xx1t**3
 capsb=1.0104-(1.419-0.3*(stem2c-20.)/80.)*xx1b
1+(2.005-0.6*(stem2c-20.)/80.)*xx1b**2
2-(1.147-0.3*(stem2c-20.)/80.)*xx1b**3
 rhcp2t=denst*capst
 rhcp2b=densb*capsb
 i=1
 do while (i.le.n)
 j=1
 do while (j.le.m)
 temp(j,i)=phi(j,i)
 if (iht.eq.2.and.i.eq.n) temp(j,i)=tamb
 temc(j,i)=phi(j,i)-273.0
 if (iht.eq.2.and.i.eq.n) temc(j,i)=tamb-273.0
 den1(j,i)=1.6603-1.9894e-3*temc(j,i)
 den2(j,i)=0.99683-1.3010e-4*temc(j,i)-2.4358e-6*temc(j,i)**2
 xx1=den1(j,i)*cf30*(1.-c(j))/(den1(j,i)*cf30*(1.-c(j))
1+den2(j,i)*(1.-cf30*(1.-c(j))))
 cno3=(1.-c(j))*cf30*den1(j,i)*1000./63.-4.*y2(j)*1000.
1/238./vfo
 if (cno3.lt.0.) cno3=0.
 dens(j,i)=den2(j,i)+0.4277*y2(j)*1000./238./vfo+0.031*cno3
 tcn1=6.1388e-4+1.3951e-6*temc(j,i)
 tcn2=1.3518e-3+2.7903e-6*temc(j,i)
 tcnm(j,i)=tcn1*xx1+tcn2*(1.-xx1)
 tcnm(j,i)=0.5*(tcnm(j,i)+5.e-4)
 caps(j,i)=1.0104-(1.419-0.3*(temc(j,i)-20.)/80.)*xx1
1+(2.005-0.6*(temc(j,i)-20.)/80.)*xx1**2
2-(1.147-0.3*(temc(j,i)-20.)/80.)*xx1**3
 rhcp(j,i)=0.5*1.25*0.52+0.5*dens(j,i)*caps(j,i)
 rhcp2(j,i)=dens(j,i)*caps(j,i)
 alpha=tcnm(j,i)/rhcp(j,i)
 bz(j,i)=(alpha+2.*dpart*vz*rhcp2(j,i)/rhcp(j,i))*dt/dz**2
 vis(j,i)=0.01002*10.**((1.3272*(20.-temc(j,i))-0.001053*
1(temc(j,i)-20.)**2)/(temc(j,i)+105.))
 j=j+1
 enddo
 i=i+1
 enddo
 j=1
 do while (j.le.m)
 alpha=tcnm(j,1)/rhcp(j,1)
 br(j,1)=(alpha+0.4*dpart*vz*rhcp2(j,1)/rhcp(j,1))*dt*4./dr**2
 j=j+1
 enddo
 i=2
 rrr=dr
 rro=0.
 do while (i.le.nn1)
 j=1
 do while (j.le.m)
 alpha=tcnm(j,i)/rhcp(j,i)
```

```
      rrn=rrr+dr
      br(j,i)=(alpha+0.4*dpart*vz*rhcp2(j,i)/rhcp(j,i))*dt
     1*4.*(rrn+rrr)/dr/((rrn+rrr)**2-(rrr+rro)**2)
      j=j+1
      enddo
      rro=rrr
      rrr=rrr+dr
      i=i+1
      enddo
c Calculate flow-weighted average effluent temperatures and concentrations at
c top and bottom of column.  These are used as inlet temperatures and
c concentrations for the radial locations where flow enters the column.
      temb=0.
      svz2=0.
      i=1
      do while (i.le.n)
      temb=temb+area(i)*temp(m,i)*vz
      svz2=svz2+area(i)*vz
      i=i+1
      enddo
      stem2=stem2+dt/vbot*(temb-svz2*stem2)
c Calculate maximum plutonium concentrations at each axial location.
      j=1
      do while (j.le.m)
      if (y2mx2(j).lt.y2(j)) y2mx2(j)=y2(j)
      j=j+1
      enddo
c Integrate Pu concentration profile and compare with amount loaded.
      pi=4.*atan(1.)
      xarea=pi*radius**2
      put=xarea*cfeed*vz*tload
      pua=0.
      puf=0.
      j=1
      do while (j.le.mm1)
      pua=pua+0.5*(f(j)+f(j+1))*xarea*dz
      puf=puf+0.5*(fx(j)+fx(j+1))*xarea*dz
      j=j+1
      enddo
      puf=puf+vtop*y2feed+vbot*y2bot
c Write transient temperatures and velocities to output files.
      time2=time2+dt
      if (time2.lt.tload) time=time2/60.
      if (time2.ge.tload.and.time2.lt.tload+twash)
     1time=(time2-tload)/60.
      if (time2.ge.tload+twash)
     1time=(time2-tload-twash)/60.
      if (abs(nint(time/tinte)-time/tinte).lt.0.0001
     1.or.abs((time2-tload)/tinte).lt.0.0001
     2.or.abs((time2-tload-twash)/tinte).lt.0.0001
     3.or.abs((time2-tload-twash-telut)/tinte).lt.0.0001) then
c Write headers for output files.
      write (11,*) 'TIME (MIN)',tab,'POS (CM)',tab,'LOAD',tab,
     1'TCEN (°C)',tab,'TMID (°C)',tab,'TOUT (°C)'
      write (16,*) 'TIME (MIN)',tab,'LENGTH (CM)',tab,'ELU (VF)',tab,
     1'ADS (G/CC)',tab,'LIQ (G/CC)',tab,'POR (G/CC)',tab,'TOT (G/CC)'
      j=1
      stem1m=stem1-273.
      write (11,102) time,tab,-1.,tab,0.,tab,stem1m,tab,stem1m,
     1tab,stem1m
      do while (j.le.m)
      poszn=length*(j-1)/(m-1)
      write (11,102) time,tab,poszn,tab,f(j),tab,temc(j,1),tab,
```

```
     1temc(j,5),tab,temc(j,n)
  102 format (f9.4,a1,f10.5,a1,f7.5,3(a1,f9.4))
      write (16,400) time,tab,poszn,tab,c(j),tab,f(j),tab,y2(j),
     1tab,y(j),tab,fx(j)
  400 format (f9.4,a1,f10.5,5(a1,f9.5))
      j=j+1
      enddo
      write (16,400) time,tab,-1.,tab,c1feed,tab,0.,tab,y2feed,tab,
     10.,tab,0.
      write (16,400) time,tab,-2.,tab,cbot,tab,0.,tab,y2bot,tab,0.,
     1tab,0.
      write (16,*) 'PU FED (GM)',tab,'PU ADS (GM)',tab,'PU TOT (GM)',
     1tab,'FRAC ADS',tab,'FRAC TOT'
      write (16,401) put,tab,pua,tab,puf,tab,pua/put,tab,puf/put
  401 format (f9.4,2(a1,f9.4),2(a1,f9.5))
      stem2m=stem2-273.
      write (11,102) time,tab,-2.,tab,0.,tab,stem2m,tab,stem2m,tab,
     1stem2m
      endif
c Find highest temperature.
      temch=0.
      i=1
      do while (i.le.n)
      j=1
      do while (j.le.m)
      if (temch.lt.temc(j,i)) then
      temch=temc(j,i)
      endif
      j=j+1
      enddo
      i=i+1
      enddo
      if (abs(nint(time/tinte2)-time/tinte2).lt.0.0001)
     1write (12,301) time,tab,temch,tab,vtop,tab,vgim
  301 format (f11.4,a1,f10.5,a1,f11.5,a1,f11.5)
      if (abs(time2-trpt).lt.0.5*dt) then
      write (19,*) stem1c
      write (19,*) stem2c
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (19,*) temc(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (19,*) temc(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
      endif
      s=s+1
      enddo
      stop
      end


c This is a subroutine that uses LU decomposition to solve a set of equations
c that is tridiagonal.  This subroutine was copies from Press, Teukolsky,
c Vetterling, and Flannery, Numerical Recipes in FORTRAN:  The Art of
Scientific
```

```
c Computing, 2nd ed., Cambridge University Press (New York), 1986, p. 43.
      subroutine tridag (a,b,c,r,u,n)
      implicit double precision (a-h,o-z)
      dimension a(n),b(n),c(n),r(n),u(n)
      dimension gam(400)
      bet=b(1)
      u(1)=r(1)/bet
      jjj=2
      do while (jjj.le.n)
      gam(jjj)=c(jjj-1)/bet
      bet=b(jjj)-a(jjj)*gam(jjj)
      u(jjj)=(r(jjj)-a(jjj)*u(jjj-1))/bet
      jjj=jjj+1
      enddo
      jjj=n-1
      do while(jjj.ge.1)
      u(jjj)=u(jjj)-gam(jjj+1)*u(jjj+1)
      jjj=jjj-1
      enddo
      return
      end

      subroutine feeder (i,m,ind2)
c Compute changes in the plutonium concentrations during feeding or washing.
      implicit double precision (a-z)
      integer i,m,ind2
      dimension c(200),cold(200),f(200),fold(200),x(200),xold(200),
     1y(200),yold(200),y2(200),y2old(200),y2mx2(200)
      common /elut1/c,cold,f,fold,x,xold,y,yold,y2,y2old,cfeed,
     1dt,tau,tau2,ffeed,an1,dpart,vz,dz,xg,vfi2,vfo,
     2y2feed,y2max,y2mx2,c1feed
      if (i.eq.1.and.ind2.eq.1) dy2=vz/vfo*dt/dz*(vfo*y2feed-y2old(1))
     1+2.*dpart*vz*dt/dz**2*(y2old(2)-y2old(1))
      if (i.eq.1.and.ind2.eq.2) dy2=-(2.*dpart*vz*dt/dz**2
     1+vz/vfo*dt/dz)*y2old(1)+2.*dpart*vz*dt/dz**2*y2old(2)
      if (i.gt.1.and.i.lt.m)
     1dy2=(2.*dpart*vz*dt/dz**2+vz/vfo*dt/dz)*y2old(i-1)
     2-(4.*dpart*vz*dt/dz**2+vz/vfo*dt/dz)*y2old(i)
     3+2.*dpart*vz*dt/dz**2*y2old(i+1)
      if (i.eq.m) dy2=(2.*dpart*vz*dt/dz**2+vz/vfo*dt/dz)
     1*y2old(m-1)-(2.*dpart*vz*dt/dz**2+vz/vfo*dt/dz)*y2old(m)
      if (y2(i)+dy2.lt.0.) dy2=-y2(i)
      xy2=y2old(i)/y2feed
      if (xold(i).lt.1.) then
      if (xold(i).gt.1.d-3)
     1x(i)=xold(i)+dt/tau*xy2*xold(i)**2/(log((1.+xold(i))/
     2(1.-xold(i)))-2.*xold(i))
      if (xold(i).le.1.d-3.and.xold(i).gt.1.d-8)
     1x(i)=xold(i)+dt/tau*xy2*1.5/xold(i)
      if (x(i).gt.1.) x(i)=1.
      if (x(i).lt.xold(i).and.i.gt.1) x(i)=xold(i)
      if (y2mx2(i).gt.1.d-3) then
      df=ffeed*(x(i)-xold(i))*y2old(i)/y2mx2(i)
      else
      df=ffeed*(x(i)-xold(i))
      endif
      if (y2old(i)+dy2.ge.y2mx2(i)) then
      if (df.gt.dy2) then
      df=dy2
      dy2=0.
      else
      dy2=dy2-df
      endif
```

```
      if (xold(i).le.1.d-8) then
      df=dy2
      dy2=0.
      endif
      else
      if (y2(i)+dy2-df.lt.0.) then
      df=y2(i)+dy2
      dy2=-y2(i)
      else
      dy2=dy2-df
      endif
      endif
      endif
      if (ind2.eq.1.and.xold(i).ge.1.) df=0.
      if (ind2.eq.2.and.xold(i).ge.1.) then
      df=0.
      if (y2(i)+dy2.lt.0.) dy2=-y2(i)
      endif
      f(i)=f(i)+df
      y2(i)=y2(i)+dy2
      x(i)=xold(i)+df/ffeed
      return
      end

      subroutine elut (i,m)
c Compute changes in nitric acid and plutonium concentrations for elution.
      implicit double precision (a-z)
      integer i,m
      dimension c(200),cold(200),f(200),fold(200),x(200),xold(200),
     1y(200),yold(200),y2(200),y2old(200),y2mx2(200)
      common /elut1/c,cold,f,fold,x,xold,y,yold,y2,y2old,cfeed,
     1dt,tau,tau2,ffeed,an1,dpart,vz,dz,xg,vfi2,vfo,
     2y2feed,y2max,y2mx2,c1feed
      if (i.eq.1) c(1)=cold(1)+an1*(vz*dt/dz*(c1feed-cold(1))
     1+2.*dpart*vz*dt/dz**2*(cold(2)-cold(1)))
      if (i.gt.1.and.i.lt.m) c(i)=cold(i)+an1*
     1((2.*dpart*vz*dt/dz**2+vz*dt/dz)*cold(i-1)-(4.*dpart*vz*dt/dz**2
     2+vz*dt/dz)*cold(i)+2.*dpart*vz*dt/dz**2*cold(i+1))
      if (i.eq.m) c(m)=cold(m)+an1*(vz*dt/dz
     1*(cold(m-1)-cold(m))+2.*dpart*dt/dz**2*(cold(m-1)-cold(m)))
      if (fold(i).gt.0.) then
      if (y2old(i).gt.1.d-8.and.cold(i).gt.1.d-8) then
      if (y2old(i)/y2max.lt.cold(i)) then
      ffcx=(cold(i)-y2old(i)/y2max)**0.5
      fcx=ffcx/(ffcx+(y2old(i)/y2max)**0.5)
      else
      fcx=1.-cold(i)
      endif
      else
      fcx=1.
      endif
      if (f(i).gt.0..and.y2old(i).lt.y2max) then
      df=-1./(1.-0.8*(fold(i)/f(i))**(1./3.)-0.2*(fold(i)/f(i))**2
     1-1.d-8)*cold(i)*dt/tau2*fold(i)*fcx
      if (df.gt.(3.*cold(i)*dt/tau2*fcx)**0.5*fold(i))
     1then
      df=(3.*cold(i)*dt/tau2*fcx)**0.5*fold(i)
      endif
      else
      df=0.
      endif
      if (f(i)-df.lt.0.) then
      df=f(i)
```

```
 f(i)=0.
 else
 f(i)=f(i)-df
 endif
 else
 df=0.
 endif
 if (y2old(i).lt.y2max) then
 if (fold(i).gt.0.) then
 vfi3=vfi2*(1.-f(i)/fold(i))**0.5
 else
 vfi3=vfi2
 endif
 if (vfi3.lt.1.d-8) vfi3=1.d-8
 if (yold(i).gt.1.d-8) then
 if (y2old(i).gt.0.) then
 y2max1=(y2max*y2old(i))**0.5
 else
 y2max1=0.
 endif
 if (abs(y2max1/vfo-yold(i)/vfi3).gt.1.d-8.and.
1abs(y2max1).gt.1.d-8) then
 dg=xg*dt*(yold(i)/vfi3)**2*((y2max1/vfo)**2
1-(yold(i)/vfi3)**2)/(2.*yold(i)/vfi3*(y2max1/vfo)**2
2-y2max1/vfo*((y2max1/vfo)**2-(yold(i)/vfi3)**2)
3*log((y2max1/vfo+yold(i)/vfi3)/(abs(y2max1/vfo-
4yold(i)/vfi3))))
 else
 dg=xg*dt*(y2max1/vfo-yold(i)/vfi3)
 endif
 else
 if (y2old(i).gt.0.) then
 dg=(1.5*y2max*y2old(i)*xg*dt)**0.5
 else
 dg=0.
 endif
 endif
 if (yold(i)+dg.lt.0.) then
 dg=-yold(i)
 y(i)=0.
 else
 y(i)=yold(i)+dg
 endif
 else
 dg=0.
 endif
 if (i.eq.1) y2(i)=y2old(i)-vz/vfo*dt/dz
1*(y2old(i)-vfo*y2feed)+2.*dpart*vz*dt/dz**2
2*(y2old(i+1)-y2old(i))+df-dg
 if (i.gt.1.and.i.lt.m)
1y2(i)=y2old(i)+(2.*dpart*vz*dt/dz**2
2+vz/vfo*dt/dz)*y2old(i-1)-(4.*dpart*vz*dt/dz**2+2.*vz/vfo*dt/dz)
3*y2old(i)+2.*dpart*vz*dt/dz**2*y2old(i+1)+df-dg
 if (i.eq.m) y2(m)=y2old(m)-vz/vfo*dt/dz
1*(y2old(m)-y2old(m-1))+2.*dpart*vz*dt/dz**2
2*(y2old(m-1)-y2old(m))+df-dg
 if (y2(i).lt.0.) then
 y(i)=y(i)+y2(i)
 y2(i)=0.
 endif
 return
 end
```

Sample Input Listing for Thermal Analysis during Normal Operation

```
25.
25.
1
34.1
2.54
2.54
3.73
0.04
0.0000012
0.0000012
0.33
0.0441
0.118
0.0263
0.13384
96,48
0
0.02
316800
12000.
149.
187.
12000.
0.25
0.05
0.274
0.0458
```

Sample Output Listing for Thermal Analysis during Normal Operation

TIME (MIN)	TMAX (°C)	VTOP (CC)	VSGMAX (CM/S)
.0500	25.00013	111.02001	.00000
.1000	25.00091	111.02001	.00000
.1500	25.00266	111.02001	.00000
.2000	25.00550	111.02001	.00000
.2500	25.00946	111.02001	.00000
.3000	25.01448	111.02001	.00000
.3500	25.02048	111.02001	.00000
.4000	25.02734	111.02001	.00000
.4500	25.03496	111.02001	.00000
.5000	25.04322	111.02001	.00000
.5500	25.05201	111.02001	.00000
.6000	25.06124	111.02001	.00000
.6500	25.07081	111.02001	.00000
.7000	25.08065	111.02001	.00000
.7500	25.09068	111.02001	.00000
.8000	25.10085	111.02001	.00000
.8500	25.11110	111.02001	.00000
.9000	25.12139	111.02001	.00000
.9500	25.13168	111.02001	.00000
1.0000	25.14449	111.02001	.00000
1.0500	25.15837	111.02001	.00000
1.1000	25.17238	111.02001	.00000
1.1500	25.18647	111.02001	.00000
1.2000	25.20060	111.02001	.00000
1.2500	25.21470	111.02001	.00000
1.3000	25.22874	111.02001	.00000
1.3500	25.24269	111.02001	.00000

1.4000	25.25650	111.02001	.00000
1.4500	25.27016	111.02001	.00000
1.5000	25.28363	111.02001	.00000
1.5500	25.29911	111.02001	.00000
1.6000	25.31623	111.02001	.00000
1.6500	25.33325	111.02001	.00000
1.7000	25.35015	111.02001	.00000
1.7500	25.36688	111.02001	.00000
1.8000	25.38341	111.02001	.00000
1.8500	25.39972	111.02001	.00000
1.9000	25.41579	111.02001	.00000
1.9500	25.43159	111.02001	.00000
2.0000	25.44709	111.02001	.00000
2.0500	25.46345	111.02001	.00000
2.1000	25.48271	111.02001	.00000
2.1500	25.50180	111.02001	.00000
2.2000	25.52068	111.02001	.00000
2.2500	25.53927	111.02001	.00000
2.3000	25.55753	111.02001	.00000
2.3500	25.57540	111.02001	.00000
2.4000	25.59287	111.02001	.00000
2.4500	25.60991	111.02001	.00000
2.5000	25.62650	111.02001	.00000
2.5500	25.64633	111.02001	.00000
2.6000	25.66681	111.02001	.00000
2.6500	25.68695	111.02001	.00000
2.7000	25.70674	111.02001	.00000
2.7500	25.72616	111.02001	.00000
2.8000	25.74513	111.02001	.00000
2.8500	25.76362	111.02001	.00000
2.9000	25.78159	111.02001	.00000
2.9500	25.79902	111.02001	.00000
3.0000	25.81996	111.02001	.00000
3.0500	25.84137	111.02001	.00000
3.1000	25.86230	111.02001	.00000
3.1500	25.88277	111.02001	.00000
3.2000	25.90276	111.02001	.00000
3.2500	25.92224	111.02001	.00000
3.3000	25.94115	111.02001	.00000
3.3500	25.95947	111.02001	.00000
3.4000	25.97843	111.02001	.00000
3.4500	26.00076	111.02001	.00000
3.5000	26.02254	111.02001	.00000
3.5500	26.04375	111.02001	.00000
3.6000	26.06440	111.02001	.00000
3.6500	26.08449	111.02001	.00000
3.7000	26.10398	111.02001	.00000
3.7500	26.12284	111.02001	.00000
3.8000	26.14103	111.02001	.00000
3.8500	26.16370	111.02001	.00000
3.9000	26.18601	111.02001	.00000
3.9500	26.20769	111.02001	.00000
4.0000	26.22874	111.02001	.00000
4.0500	26.24917	111.02001	.00000
4.1000	26.26896	111.02001	.00000
4.1500	26.28810	111.02001	.00000
4.2000	26.30653	111.02001	.00000
4.2500	26.32896	111.02001	.00000
4.3000	26.35155	111.02001	.00000
4.3500	26.37347	111.02001	.00000
4.4000	26.39471	111.02001	.00000
4.4500	26.41529	111.02001	.00000
4.5000	26.43520	111.02001	.00000

4.5500	26.45442	111.02001	.00000
4.6000	26.47292	111.02001	.00000
4.6500	26.49560	111.02001	.00000
4.7000	26.51828	111.02001	.00000
4.7500	26.54028	111.02001	.00000
4.8000	26.56156	111.02001	.00000
4.8500	26.58214	111.02001	.00000
4.9000	26.60204	111.02001	.00000
4.9500	26.62122	111.02001	.00000
5.0000	26.63967	111.02001	.00000
5.0500	26.66293	111.02001	.00000
5.1000	26.68559	111.02001	.00000
5.1500	26.70754	111.02001	.00000
5.2000	26.72875	111.02001	.00000
5.2500	26.74925	111.02001	.00000
5.3000	26.76904	111.02001	.00000
5.3500	26.78811	111.02001	.00000
5.4000	26.80726	111.02001	.00000
5.4500	26.83050	111.02001	.00000
5.5000	26.85305	111.02001	.00000
5.5500	26.87487	111.02001	.00000
5.6000	26.89595	111.02001	.00000
5.6500	26.91629	111.02001	.00000
5.7000	26.93593	111.02001	.00000
5.7500	26.95482	111.02001	.00000
5.8000	26.97491	111.02001	.00000
5.8500	26.99801	111.02001	.00000
5.9000	27.02039	111.02001	.00000
5.9500	27.04204	111.02001	.00000
6.0000	27.06293	111.02001	.00000
6.0500	27.08309	111.02001	.00000
6.1000	27.10253	111.02001	.00000
6.1500	27.12122	111.02001	.00000
6.2000	27.14237	111.02001	.00000
6.2500	27.16528	111.02001	.00000
6.3000	27.18747	111.02001	.00000
6.3500	27.20891	111.02001	.00000
6.4000	27.22960	111.02001	.00000
6.4500	27.24955	111.02001	.00000
6.5000	27.26877	111.02001	.00000
6.5500	27.28724	111.02001	.00000
6.6000	27.30951	111.02001	.00000
6.6500	27.33222	111.02001	.00000
6.7000	27.35419	111.02001	.00000
6.7500	27.37541	111.02001	.00000
6.8000	27.39588	111.02001	.00000
6.8500	27.41561	111.02001	.00000
6.9000	27.43461	111.02001	.00000
6.9500	27.45310	111.02001	.00000
7.0000	27.47630	111.02001	.00000
7.0500	27.49879	111.02001	.00000
7.1000	27.52054	111.02001	.00000
7.1500	27.54153	111.02001	.00000
7.2000	27.56178	111.02001	.00000
7.2500	27.58129	111.02001	.00000
7.3000	27.60007	111.02001	.00000
7.3500	27.61973	111.02001	.00000
7.4000	27.64272	111.02001	.00000
7.4500	27.66499	111.02001	.00000
7.5000	27.68651	111.02001	.00000
7.5500	27.70728	111.02001	.00000
7.6000	27.72730	111.02001	.00000
7.6500	27.74660	111.02001	.00000

7.7000	27.76515	111.02001	.00000
7.7500	27.78600	111.02001	.00000
7.8000	27.80878	111.02001	.00000
7.8500	27.83083	111.02001	.00000
7.9000	27.85213	111.02001	.00000
7.9500	27.87268	111.02001	.00000
8.0000	27.89249	111.02001	.00000
8.0500	27.91156	111.02001	.00000
8.1000	27.92989	111.02001	.00000
8.1500	27.95195	111.02001	.00000
8.2000	27.97451	111.02001	.00000
8.2500	27.99635	111.02001	.00000
8.3000	28.01743	111.02001	.00000
8.3500	28.03776	111.02001	.00000
8.4000	28.05736	111.02001	.00000
8.4500	28.07623	111.02001	.00000
8.5000	28.09453	111.02001	.00000
8.5500	28.11759	111.02001	.00000
8.6000	28.13995	111.02001	.00000
8.6500	28.16157	111.02001	.00000
8.7000	28.18245	111.02001	.00000
8.7500	28.20257	111.02001	.00000
8.8000	28.22196	111.02001	.00000
8.8500	28.24062	111.02001	.00000
8.9000	28.26010	111.02001	.00000
8.9500	28.28297	111.02001	.00000
9.0000	28.30513	111.02001	.00000
9.0500	28.32655	111.02001	.00000
9.1000	28.34721	111.02001	.00000
9.1500	28.36713	111.02001	.00000
9.2000	28.38632	111.02001	.00000
9.2500	28.40478	111.02001	.00000
9.3000	28.42544	111.02001	.00000
9.3500	28.44812	111.02001	.00000
9.4000	28.47007	111.02001	.00000
9.4500	28.49129	111.02001	.00000
9.5000	28.51176	111.02001	.00000
9.5500	28.53148	111.02001	.00000
9.6000	28.55048	111.02001	.00000
9.6500	28.56874	111.02001	.00000
9.7000	28.59057	111.02001	.00000
9.7500	28.61306	111.02001	.00000
9.8000	28.63482	111.02001	.00000
9.8500	28.65584	111.02001	.00000
9.9000	28.67611	111.02001	.00000
9.9500	28.69565	111.02001	.00000
10.0000	28.71446	111.02001	.00000
10.0500	28.73252	111.02001	.00000
10.1000	28.75552	111.02001	.00000
10.1500	28.77782	111.02001	.00000
10.2000	28.79940	111.02001	.00000
10.2500	28.82022	111.02001	.00000
10.3000	28.84030	111.02001	.00000
10.3500	28.85965	111.02001	.00000
10.4000	28.87827	111.02001	.00000
10.4500	28.89749	111.02001	.00000
10.5000	28.92031	111.02001	.00000
10.5500	28.94243	111.02001	.00000
10.6000	28.96382	111.02001	.00000
10.6500	28.98446	111.02001	.00000
10.7000	29.00435	111.02001	.00000
10.7500	29.02352	111.02001	.00000
10.8000	29.04195	111.02001	.00000

10.8500	29.06232	111.02001	.00000
10.9000	29.08497	111.02001	.00000
10.9500	29.10690	111.02001	.00000
11.0000	29.12811	111.02001	.00000
11.0500	29.14856	111.02001	.00000
11.1000	29.16827	111.02001	.00000
11.1500	29.18726	111.02001	.00000
11.2000	29.20551	111.02001	.00000
11.2500	29.22702	111.02001	.00000
11.3000	29.24950	111.02001	.00000
11.3500	29.27126	111.02001	.00000
11.4000	29.29227	111.02001	.00000
11.4500	29.31254	111.02001	.00000
11.5000	29.33207	111.02001	.00000
11.5500	29.35089	111.02001	.00000
11.6000	29.36896	111.02001	.00000
11.6500	29.39161	111.02001	.00000
11.7000	29.41392	111.02001	.00000
11.7500	29.43550	111.02001	.00000
11.8000	29.45633	111.02001	.00000
11.8500	29.47642	111.02001	.00000
11.9000	29.49578	111.02001	.00000
11.9500	29.51442	111.02001	.00000
12.0000	29.53328	111.02001	.00000
12.0500	29.55610	111.02001	.00000
12.1000	29.57824	111.02001	.00000
12.1500	29.59964	111.02001	.00000
12.2000	29.62029	111.02001	.00000
12.2500	29.64020	111.02001	.00000
12.3000	29.65939	111.02001	.00000
12.3500	29.67785	111.02001	.00000
12.4000	29.69784	111.02001	.00000
12.4500	29.72050	111.02001	.00000
12.5000	29.74246	111.02001	.00000
12.5500	29.76369	111.02001	.00000
12.6000	29.78416	111.02001	.00000
12.6500	29.80390	111.02001	.00000
12.7000	29.82292	111.02001	.00000
12.7500	29.84120	111.02001	.00000
12.8000	29.86232	111.02001	.00000
12.8500	29.88481	111.02001	.00000
12.9000	29.90660	111.02001	.00000
12.9500	29.92765	111.02001	.00000
13.0000	29.94795	111.02001	.00000
13.0500	29.96752	111.02001	.00000
13.1000	29.98636	111.02001	.00000
13.1500	30.00447	111.02001	.00000
13.2000	30.02671	111.02001	.00000
13.2500	30.04904	111.02001	.00000
13.3000	30.07066	111.02001	.00000
13.3500	30.09153	111.02001	.00000
13.4000	30.11166	111.02001	.00000
13.4500	30.13105	111.02001	.00000
13.5000	30.14973	111.02001	.00000
13.5500	30.16816	111.02001	.00000
13.6000	30.19103	111.02001	.00000
13.6500	30.21319	111.02001	.00000
13.7000	30.23464	111.02001	.00000
13.7500	30.25533	111.02001	.00000
13.8000	30.27528	111.02001	.00000
13.8500	30.29451	111.02001	.00000
13.9000	30.31302	111.02001	.00000
13.9500	30.33256	111.02001	.00000

14.0000	30.35527	111.02001	.00000
14.0500	30.37727	111.02001	.00000
14.1000	30.39853	111.02001	.00000
14.1500	30.41906	111.02001	.00000
14.2000	30.43884	111.02001	.00000
14.2500	30.45790	111.02001	.00000
14.3000	30.47623	111.02001	.00000
14.3500	30.49689	111.02001	.00000
14.4000	30.51943	111.02001	.00000
14.4500	30.54126	111.02001	.00000
14.5000	30.56236	111.02001	.00000
14.5500	30.58270	111.02001	.00000
14.6000	30.60232	111.02001	.00000
14.6500	30.62121	111.02001	.00000
14.7000	30.63937	111.02001	.00000
14.7500	30.66114	111.02001	.00000
14.8000	30.68352	111.02001	.00000
14.8500	30.70518	111.02001	.00000
14.9000	30.72610	111.02001	.00000
14.9500	30.74628	111.02001	.00000
15.0000	30.76572	111.02001	.00000
15.0500	30.78445	111.02001	.00000
15.1000	30.80243	111.02001	.00000
15.1500	30.82532	111.02001	.00000
15.2000	30.84753	111.02001	.00000
15.2500	30.86902	111.02001	.00000
15.3000	30.88977	111.02001	.00000
15.3500	30.90977	111.02001	.00000
15.4000	30.92905	111.02001	.00000
15.4500	30.94761	111.02001	.00000
15.5000	30.96667	111.02001	.00000
15.5500	30.98942	111.02001	.00000
15.6000	31.01147	111.02001	.00000
15.6500	31.03279	111.02001	.00000
15.7000	31.05336	111.02001	.00000
15.7500	31.07319	111.02001	.00000
15.8000	31.09231	111.02001	.00000
15.8500	31.11069	111.02001	.00000
15.9000	31.13085	111.02001	.00000
15.9500	31.15344	111.02001	.00000
16.0000	31.17532	111.02001	.00000
16.0500	31.19647	111.02001	.00000
16.1000	31.21687	111.02001	.00000
16.1500	31.23654	111.02001	.00000
16.2000	31.25548	111.02001	.00000
16.2500	31.27369	111.02001	.00000
16.3000	31.29496	111.02001	.00000
16.3500	31.31738	111.02001	.00000
16.4000	31.33910	111.02001	.00000
16.4500	31.36008	111.02001	.00000
16.5000	31.38030	111.02001	.00000
16.5500	31.39980	111.02001	.00000
16.6000	31.41858	111.02001	.00000
16.6500	31.43662	111.02001	.00000
16.7000	31.45898	111.02001	.00000
16.7500	31.48125	111.02001	.00000
16.8000	31.50279	111.02001	.00000
16.8500	31.52360	111.02001	.00000
16.9000	31.54365	111.02001	.00000
16.9500	31.56298	111.02001	.00000
17.0000	31.58159	111.02001	.00000
17.0500	31.60012	111.02001	.00000
17.1000	31.62292	111.02001	.00000

17.1500	31.64502	111.02001	.00000
17.2000	31.66640	111.02001	.00000
17.2500	31.68703	111.02001	.00000
17.3000	31.70692	111.02001	.00000
17.3500	31.72608	111.02001	.00000
17.4000	31.74452	111.02001	.00000
17.4500	31.76414	111.02001	.00000
17.5000	31.78678	111.02001	.00000
17.5500	31.80872	111.02001	.00000
17.6000	31.82992	111.02001	.00000
17.6500	31.85038	111.02001	.00000
17.7000	31.87010	111.02001	.00000
17.7500	31.88910	111.02001	.00000
17.8000	31.90737	111.02001	.00000
17.8500	31.92807	111.02001	.00000
17.9000	31.95055	111.02001	.00000
17.9500	31.97232	111.02001	.00000
18.0000	31.99335	111.02001	.00000
18.0500	32.01363	111.02001	.00000
18.1000	32.03319	111.02001	.00000
18.1500	32.05202	111.02001	.00000
18.2000	32.07012	111.02001	.00000
18.2500	32.09190	111.02001	.00000
18.3000	32.11422	111.02001	.00000
18.3500	32.13582	111.02001	.00000
18.4000	32.15668	111.02001	.00000
18.4500	32.17680	111.02001	.00000
18.5000	32.19618	111.02001	.00000
18.5500	32.21485	111.02001	.00000
18.6000	32.23280	111.02001	.00000
18.6500	32.25565	111.02001	.00000
18.7000	32.27780	111.02001	.00000
18.7500	32.29923	111.02001	.00000
18.8000	32.31992	111.02001	.00000
18.8500	32.33987	111.02001	.00000
18.9000	32.35909	111.02001	.00000
18.9500	32.37758	111.02001	.00000
19.0000	32.39660	111.02001	.00000
19.0500	32.41929	111.02001	.00000
19.1000	32.44128	111.02001	.00000
19.1500	32.46255	111.02001	.00000
19.2000	32.48306	111.02001	.00000
19.2500	32.50284	111.02001	.00000
19.3000	32.52189	111.02001	.00000
19.3500	32.54022	111.02001	.00000
19.4000	32.56030	111.02001	.00000
19.4500	32.58284	111.02001	.00000
19.5000	32.60466	111.02001	.00000
19.5500	32.62575	111.02001	.00000
19.6000	32.64610	111.02001	.00000
19.6500	32.66570	111.02001	.00000
19.7000	32.68459	111.02001	.00000
19.7500	32.70275	111.02001	.00000
19.8000	32.72390	111.02001	.00000
19.8500	32.74628	111.02001	.00000
19.9000	32.76793	111.02001	.00000
19.9500	32.78885	111.02001	.00000
20.0000	32.80902	111.02001	.00000
20.0500	32.82847	111.02001	.00000
20.1000	32.84719	111.02001	.00000
20.1500	32.86517	111.02001	.00000
20.2000	32.88740	111.02001	.00000
20.2500	32.90961	111.02001	.00000

20.3000	32.93110	111.02001	.00000
20.3500	32.95184	111.02001	.00000
20.4000	32.97184	111.02001	.00000
20.4500	32.99112	111.02001	.00000
20.5000	33.00968	111.02001	.00000
20.5500	33.02803	111.02001	.00000
20.6000	33.05078	111.02001	.00000
20.6500	33.07283	111.02001	.00000
20.7000	33.09414	111.02001	.00000
20.7500	33.11472	111.02001	.00000
20.8000	33.13455	111.02001	.00000
20.8500	33.15366	111.02001	.00000
20.9000	33.17205	111.02001	.00000
20.9500	33.19146	111.02001	.00000
21.0000	33.21405	111.02001	.00000
21.0500	33.23593	111.02001	.00000
21.1000	33.25707	111.02001	.00000
21.1500	33.27748	111.02001	.00000
21.2000	33.29714	111.02001	.00000
21.2500	33.31609	111.02001	.00000
21.3000	33.33430	111.02001	.00000
21.3500	33.35477	111.02001	.00000
21.4000	33.37719	111.02001	.00000
21.4500	33.39891	111.02001	.00000
21.5000	33.41988	111.02001	.00000
21.5500	33.44011	111.02001	.00000
21.6000	33.45961	111.02001	.00000
21.6500	33.47839	111.02001	.00000
21.7000	33.49643	111.02001	.00000
21.7500	33.51795	111.02001	.00000
21.8000	33.54022	111.02001	.00000
21.8500	33.56176	111.02001	.00000
21.9000	33.58257	111.02001	.00000
21.9500	33.60262	111.02001	.00000
22.0000	33.62196	111.02001	.00000
22.0500	33.64057	111.02001	.00000
22.1000	33.65844	111.02001	.00000
22.1500	33.68101	111.02001	.00000
22.2000	33.70311	111.02001	.00000
22.2500	33.72449	111.02001	.00000
22.3000	33.74512	111.02001	.00000
22.3500	33.76501	111.02001	.00000
22.4000	33.78417	111.02001	.00000
22.4500	33.80262	111.02001	.00000
22.5000	33.82130	111.02001	.00000
22.5500	33.84394	111.02001	.00000
22.6000	33.86587	111.02001	.00000
22.6500	33.88708	111.02001	.00000
22.7000	33.90754	111.02001	.00000
22.7500	33.92726	111.02001	.00000
22.8000	33.94626	111.02001	.00000
22.8500	33.96453	111.02001	.00000
22.9000	33.98425	111.02001	.00000
22.9500	34.00673	111.02001	.00000
23.0000	34.02850	111.02001	.00000
23.0500	34.04953	111.02001	.00000
23.1000	34.06982	111.02001	.00000
23.1500	34.08937	111.02001	.00000
23.2000	34.10821	111.02001	.00000
23.2500	34.12631	111.02001	.00000
23.3000	34.14706	111.02001	.00000
23.3500	34.16938	111.02001	.00000
23.4000	34.19098	111.02001	.00000

23.4500	34.21184	111.02001	.00000
23.5000	34.23196	111.02001	.00000
23.5500	34.25135	111.02001	.00000
23.6000	34.27002	111.02001	.00000
23.6500	34.28795	111.02001	.00000
23.7000	34.30973	111.02001	.00000
23.7500	34.33189	111.02001	.00000
23.8000	34.35332	111.02001	.00000
23.8500	34.37401	111.02001	.00000
23.9000	34.39396	111.02001	.00000
23.9500	34.41318	111.02001	.00000
24.0000	34.43168	111.02001	.00000
24.0500	34.44956	111.02001	.00000
24.1000	34.47225	111.02001	.00000
24.1500	34.49425	111.02001	.00000
24.2000	34.51551	111.02001	.00000
24.2500	34.53603	111.02001	.00000
24.3000	34.55581	111.02001	.00000
24.3500	34.57486	111.02001	.00000
24.4000	34.59320	111.02001	.00000
24.4500	34.61209	111.02001	.00000
24.5000	34.63463	111.02001	.00000
24.5500	34.65645	111.02001	.00000
24.6000	34.67754	111.02001	.00000
24.6500	34.69789	111.02001	.00000
24.7000	34.71750	111.02001	.00000
24.7500	34.73639	111.02001	.00000
24.8000	34.75455	111.02001	.00000
24.8500	34.77447	111.02001	.00000
24.9000	34.79684	111.02001	.00000
24.9500	34.81850	111.02001	.00000
25.0000	34.83942	111.02001	.00000
25.0500	34.85960	111.02001	.00000
25.1000	34.87904	111.02001	.00000
25.1500	34.89777	111.02001	.00000
25.2000	34.91575	111.02001	.00000
25.2500	34.93669	111.02001	.00000
25.3000	34.95890	111.02001	.00000
25.3500	34.98039	111.02001	.00000
25.4000	35.00114	111.02001	.00000
25.4500	35.02114	111.02001	.00000
25.5000	35.04042	111.02001	.00000
25.5500	35.05898	111.02001	.00000
25.6000	35.07679	111.02001	.00000
25.6500	35.09874	111.02001	.00000
25.7000	35.12079	111.02001	.00000
25.7500	35.14211	111.02001	.00000
25.8000	35.16268	111.02001	.00000
25.8500	35.18252	111.02001	.00000
25.9000	35.20163	111.02001	.00000
25.9500	35.22002	111.02001	.00000
26.0000	35.23804	111.02001	.00000
26.0500	35.26063	111.02001	.00000
26.1000	35.28251	111.02001	.00000
26.1500	35.30366	111.02001	.00000
26.2000	35.32406	111.02001	.00000
26.2500	35.34373	111.02001	.00000
26.3000	35.36268	111.02001	.00000
26.3500	35.38090	111.02001	.00000
26.4000	35.39991	111.02001	.00000
26.4500	35.42234	111.02001	.00000
26.5000	35.44405	111.02001	.00000
26.5500	35.46503	111.02001	.00000

26.6000	35.48527	111.02001	.00000
26.6500	35.50477	111.02001	.00000
26.7000	35.52355	111.02001	.00000
26.7500	35.54160	111.02001	.00000
26.8000	35.56161	111.02001	.00000
26.8500	35.58388	111.02001	.00000
26.9000	35.60542	111.02001	.00000
26.9500	35.62623	111.02001	.00000
27.0000	35.64629	111.02001	.00000
27.0500	35.66563	111.02001	.00000
27.1000	35.68424	111.02001	.00000
27.1500	35.70212	111.02001	.00000
27.2000	35.72313	111.02001	.00000
27.2500	35.74523	111.02001	.00000
27.3000	35.76661	111.02001	.00000
27.3500	35.78724	111.02001	.00000
27.4000	35.80714	111.02001	.00000
27.4500	35.82631	111.02001	.00000
27.5000	35.84475	111.02001	.00000
27.5500	35.86245	111.02001	.00000
27.6000	35.88446	111.02001	.00000
27.6500	35.90640	111.02001	.00000
27.7000	35.92761	111.02001	.00000
27.7500	35.94807	111.02001	.00000
27.8000	35.96780	111.02001	.00000
27.8500	35.98680	111.02001	.00000
27.9000	36.00508	111.02001	.00000
27.9500	36.02312	111.02001	.00000
28.0000	36.04561	111.02001	.00000
28.0500	36.06738	111.02001	.00000
28.1000	36.08842	111.02001	.00000
28.1500	36.10871	111.02001	.00000
28.2000	36.12826	111.02001	.00000
28.2500	36.14711	111.02001	.00000
28.3000	36.16521	111.02001	.00000
28.3500	36.18424	111.02001	.00000
28.4000	36.20656	111.02001	.00000
28.4500	36.22816	111.02001	.00000
28.5000	36.24903	111.02001	.00000
28.5500	36.26915	111.02001	.00000
28.6000	36.28854	111.02001	.00000
28.6500	36.30721	111.02001	.00000
28.7000	36.32515	111.02001	.00000
28.7500	36.34515	111.02001	.00000
28.8000	36.36731	111.02001	.00000
28.8500	36.38875	111.02001	.00000
28.9000	36.40944	111.02001	.00000
28.9500	36.42939	111.02001	.00000
29.0000	36.44862	111.02001	.00000
29.0500	36.46713	111.02001	.00000
29.1000	36.48488	111.02001	.00000
29.1500	36.50587	111.02001	.00000
29.2000	36.52787	111.02001	.00000
29.2500	36.54913	111.02001	.00000
29.3000	36.56965	111.02001	.00000
29.3500	36.58943	111.02001	.00000
29.4000	36.60850	111.02001	.00000
29.4500	36.62683	111.02001	.00000
29.5000	36.64441	111.02001	.00000
29.5500	36.66638	111.02001	.00000
29.6000	36.68821	111.02001	.00000
29.6500	36.70931	111.02001	.00000
29.7000	36.72966	111.02001	.00000

29.7500	36.74927	111.02001	.00000
29.8000	36.76817	111.02001	.00000
29.8500	36.78633	111.02001	.00000
29.9000	36.80431	111.02001	.00000
29.9500	36.82669	111.02001	.00000
30.0000	36.84835	111.02001	.00000
30.0500	36.86927	111.02001	.00000
30.1000	36.88945	111.02001	.00000
30.1500	36.90890	111.02001	.00000
30.2000	36.92763	111.02001	.00000
30.2500	36.94562	111.02001	.00000
30.3000	36.96456	111.02001	.00000
30.3500	36.98678	111.02001	.00000
30.4000	37.00827	111.02001	.00000
30.4500	37.02902	111.02001	.00000
30.5000	37.04903	111.02001	.00000
30.5500	37.06831	111.02001	.00000
30.6000	37.08688	111.02001	.00000
30.6500	37.10469	111.02001	.00000
30.7000	37.12460	111.02001	.00000
30.7500	37.14665	111.02001	.00000
30.8000	37.16797	111.02001	.00000
30.8500	37.18855	111.02001	.00000
30.9000	37.20839	111.02001	.00000
30.9500	37.22751	111.02001	.00000
31.0000	37.24591	111.02001	.00000
31.0500	37.26355	111.02001	.00000
31.1000	37.28442	111.02001	.00000
31.1500	37.30630	111.02001	.00000
31.2000	37.32745	111.02001	.00000
31.2500	37.34786	111.02001	.00000
31.3000	37.36753	111.02001	.00000
31.3500	37.38649	111.02001	.00000
31.4000	37.40471	111.02001	.00000
31.4500	37.42217	111.02001	.00000
31.5000	37.44401	111.02001	.00000
31.5500	37.46573	111.02001	.00000
31.6000	37.48671	111.02001	.00000
31.6500	37.50695	111.02001	.00000
31.7000	37.52645	111.02001	.00000
31.7500	37.54524	111.02001	.00000
31.8000	37.56329	111.02001	.00000
31.8500	37.58110	111.02001	.00000
31.9000	37.60337	111.02001	.00000
31.9500	37.62492	111.02001	.00000
32.0000	37.64573	111.02001	.00000
32.0500	37.66580	111.02001	.00000
32.1000	37.68514	111.02001	.00000
32.1500	37.70376	111.02001	.00000
32.2000	37.72164	111.02001	.00000
32.2500	37.74040	111.02001	.00000
32.3000	37.76250	111.02001	.00000
32.3500	37.78389	111.02001	.00000
32.4000	37.80452	111.02001	.00000
32.4500	37.82442	111.02001	.00000
32.5000	37.84360	111.02001	.00000
32.5500	37.86205	111.02001	.00000
32.6000	37.87975	111.02001	.00000
32.6500	37.89946	111.02001	.00000
32.7000	37.92140	111.02001	.00000
32.7500	37.94261	111.02001	.00000
32.8000	37.96308	111.02001	.00000
32.8500	37.98281	111.02001	.00000

32.9000	38.00182	111.02001	.00000
32.9500	38.02010	111.02001	.00000
33.0000	38.03762	111.02001	.00000
33.0500	38.05828	111.02001	.00000
33.1000	38.08005	111.02001	.00000
33.1500	38.10110	111.02001	.00000
33.2000	38.12139	111.02001	.00000
33.2500	38.14095	111.02001	.00000
33.3000	38.15980	111.02001	.00000
33.3500	38.17791	111.02001	.00000
33.4000	38.19526	111.02001	.00000
33.4500	38.21686	111.02001	.00000
33.5000	38.23847	111.02001	.00000
33.5500	38.25934	111.02001	.00000
33.6000	38.27946	111.02001	.00000
33.6500	38.29886	111.02001	.00000
33.7000	38.31753	111.02001	.00000
33.7500	38.33547	111.02001	.00000
33.8000	38.35303	111.02001	.00000
33.8500	38.37519	111.02001	.00000
33.9000	38.39663	111.02001	.00000
33.9500	38.41733	111.02001	.00000
34.0000	38.43728	111.02001	.00000
34.0500	38.45651	111.02001	.00000
34.1000	38.47502	111.02001	.00000
34.1500	38.49279	111.02001	.00000
34.2000	38.51128	111.02001	.00000
34.2500	38.53327	111.02001	.00000
34.3000	38.55454	111.02001	.00000
34.3500	38.57507	111.02001	.00000
34.4000	38.59485	111.02001	.00000
34.4500	38.61392	111.02001	.00000
34.5000	38.63226	111.02001	.00000
34.5500	38.64985	111.02001	.00000
34.6000	38.66927	111.02001	.00000
34.6500	38.69110	111.02001	.00000
34.7000	38.71220	111.02001	.00000
34.7500	38.73256	111.02001	.00000
34.8000	38.75217	111.02001	.00000
34.8500	38.77108	111.02001	.00000
34.9000	38.78925	111.02001	.00000
34.9500	38.80665	111.02001	.00000
35.0000	38.82701	111.02001	.00000
35.0500	38.84867	111.02001	.00000
35.1000	38.86960	111.02001	.00000
35.1500	38.88978	111.02001	.00000
35.2000	38.90924	111.02001	.00000
35.2500	38.92797	111.02001	.00000
35.3000	38.94597	111.02001	.00000
35.3500	38.96320	111.02001	.00000
35.4000	38.98448	111.02001	.00000
35.4500	39.00597	111.02001	.00000
35.5000	39.02673	111.02001	.00000
35.5500	39.04674	111.02001	.00000
35.6000	39.06603	111.02001	.00000
35.6500	39.08460	111.02001	.00000
35.7000	39.10242	111.02001	.00000
35.7500	39.11961	111.02001	.00000
35.8000	39.14166	111.02001	.00000
35.8500	39.16299	111.02001	.00000
35.9000	39.18357	111.02001	.00000
35.9500	39.20341	111.02001	.00000
36.0000	39.22254	111.02001	.00000

36.0500	39.24094	111.02001	.00000
36.1000	39.25859	111.02001	.00000
36.1500	39.27658	111.02001	.00000
36.2000	39.29847	111.02001	.00000
36.2500	39.31962	111.02001	.00000
36.3000	39.34004	111.02001	.00000
36.3500	39.35971	111.02001	.00000
36.4000	39.37867	111.02001	.00000
36.4500	39.39690	111.02001	.00000
36.5000	39.41437	111.02001	.00000
36.5500	39.43283	111.02001	.00000
36.6000	39.45455	111.02001	.00000
36.6500	39.47554	111.02001	.00000
36.7000	39.49578	111.02001	.00000
36.7500	39.51529	111.02001	.00000
36.8000	39.53408	111.02001	.00000
36.8500	39.55214	111.02001	.00000
36.9000	39.56943	111.02001	.00000
36.9500	39.58696	111.02001	.00000
37.0000	39.60851	111.02001	.00000
37.0500	39.62932	111.02001	.00000
37.1000	39.64938	111.02001	.00000
37.1500	39.66873	111.02001	.00000
37.2000	39.68735	111.02001	.00000
37.2500	39.70523	111.02001	.00000
37.3000	39.72233	111.02001	.00000
37.3500	39.73866	111.02001	.00000
37.4000	39.75447	111.02001	.00000
37.4500	39.77510	111.02001	.00000
37.5000	39.79498	111.02001	.00000
37.5500	39.81415	111.02001	.00000
37.6000	39.83259	111.02001	.00000
37.6500	39.85028	111.02001	.00000
37.7000	39.86720	111.02001	.00000
37.7500	39.88332	111.02001	.00000
37.8000	39.89866	111.02001	.00000
37.8500	39.91323	111.02001	.00000
37.9000	39.92706	111.02001	.00000
37.9500	39.94018	111.02001	.00000
38.0000	39.95255	111.02001	.00000
38.0500	39.96417	111.02001	.00000
38.1000	39.97967	111.02001	.00000
38.1500	39.99558	111.02001	.00000
38.2000	40.01069	111.02001	.00000
38.2500	40.02505	111.02001	.00000
38.3000	40.03868	111.02001	.00000
38.3500	40.05159	111.02001	.00000
38.4000	40.06375	111.02001	.00000
38.4500	40.07515	111.02001	.00000
38.5000	40.08581	111.02001	.00000
38.5500	40.09576	111.02001	.00000
38.6000	40.10503	111.02001	.00000
38.6500	40.11365	111.02001	.00000
38.7000	40.12164	111.02001	.00000
38.7500	40.12896	111.02001	.00000
38.8000	40.13565	111.02001	.00000
38.8500	40.14174	111.02001	.00000
38.9000	40.14727	111.02001	.00000
38.9500	40.15230	111.02001	.00000
39.0000	40.15683	111.02001	.00000
39.0500	40.16083	111.02001	.00000
39.1000	40.16431	111.02001	.00000
39.1500	40.16737	111.02001	.00000

```
   39.2000    40.17007     111.02001       .00000
   39.2500    40.17247     111.02001       .00000
   39.3000    40.17461     111.02001       .00000
   39.3500    40.17654     111.02001       .00000
   39.4000    40.17827     111.02001       .00000
   39.4500    40.17981     111.02001       .00000
   39.5000    40.18119     111.02001       .00000
   39.5500    40.18241     111.02001       .00000
   39.6000    40.18349     111.02001       .00000
   39.6500    40.18445     111.02001       .00000
   39.7000    40.18530     111.02001       .00000
   39.7500    40.18606     111.02001       .00000
   39.8000    40.18674     111.02001       .00000
   39.8500    40.18735     111.02001       .00000
   39.9000    40.18790     111.02001       .00000
   39.9500    40.18839     111.02001       .00000
   40.0000    40.18882     111.02001       .00000
   40.0500    40.18921     111.02001       .00000
   40.1000    40.18956     111.02001       .00000
   40.1500    40.18988     111.02001       .00000
   40.2000    40.19017     111.02001       .00000
   40.2500    40.19042     111.02001       .00000
   40.3000    40.19066     111.02001       .00000
   40.3500    40.19087     111.02001       .00000
   40.4000    40.19106     111.02001       .00000
   40.4500    40.19123     111.02001       .00000
   40.5000    40.19139     111.02001       .00000
   40.5500    40.19153     111.02001       .00000
   40.6000    40.19166     111.02001       .00000
   40.6500    40.19178     111.02001       .00000
   40.7000    40.19189     111.02001       .00000
   40.7500    40.19199     111.02001       .00000
   40.8000    40.19208     111.02001       .00000
   40.8500    40.19216     111.02001       .00000
   40.9000    40.19223     111.02001       .00000
   40.9500    40.19230     111.02001       .00000
   41.0000    40.19236     111.02001       .00000
```

Source Code Listing for Thermal Analysis during a Flow Stoppage

```
c This program calculates the temperature transient following a flow
c interruption during elution of an ion exchange column.  It includes the
c effects of thermal conduction to the column walls and natural convection due
c to temperature and concentration gradients in the bulk solution.  The effect
c of plutonium and nitric acid on the bulk solution density is included.  The
c program solves finite difference energy balance equations with forward time
c differencing, donor cell differencing for convection terms, and centered
space
c differencing for conduction terms.  A list of variable definitions follows.
c
c aa = parameter for calculation of saturated plutonium concentration
c ab = parameter for calculation of saturated nitrate concentration
c af1 = area factor for inward radial flow, 1/cm
c af2 = area factor for outward radial flow, 1/cm
c alpha = thermal diffusivity, cm2/s
c an1 = fraction of void volume that lies between resin beads
c area = partial cross-sectional area of resin bed for calculation node, cm2
c areat = total cross-sectional area of resin bed, cm2
c br = nondimensional radial dispersion coefficient for heat transfer equation
c brc = nondimensional radial dispersion coefficient for acid mass transfer
```

```
c       equation
c brc1 = nondimensional radial dispersion coefficient for acid mass transfer
c        equation, due to area change with radius
c bry = nondimensional radial dispersion coefficient for Pu mass transfer
c       equation
c bry1 = nondimensional radial dispersion coefficient for Pu mass transfer
c        equation, due to area change with radius
c br1 = nondimensional radial dispersion coefficient for heat transfer equation,
c       due to area change with radius
c bz = nondimensional axial dispersion coefficient for heat transfer equation
c bzc = nondimensional axial dispersion coefficient for acid mass transfer
c       equation
c bzy = nondimensional axial dispersion coefficient for Pu mass transfer
c       equation
c c = volume fraction acid in solution between resin beads
c cold = volume fraction acid in solution between resin beads at previous time
c        step
c cond = thermal conductivity of resin, cal/cm/s/C
c caps = heat capacity of liquid, cal/g/C
c capsb = heat capacity of liquid in volume below resin bed, cal/g/C
c capst = heat capacity of liquid in volume above resin bed, cal/g/C
c cfeed = plutonium concentration in feed solution, g/cc
c cf30 = volume fraction of feed solution occupied by HNO3
c cmax2 = maximum total nitrate concentration for a saturated plutonium nitrate
c         solution, gmole/lit
c cmx2 = 1 - nitric acid concentration/nitric acid concentration at saturation
c cno3 = nitric acid concentration, gmole/lit
c cno3b = nitric acid concentration in liquid volume below resin bed, gmole/lit
c cno3t = nitric acid concentration in liquid volume above resin bed, gmole/lit
c cno32 = saturation nitrate concentration, calculated as a function of
c         plutonium concentration, gmole/lit
c cp = heat capacity, cal/g/C
c cpu = local plutonium concentration in solution between resin beads,
gmole/lit
c cpu2 = saturation plutonium concentration, calculated as a function of acid
c        concentration, gmole/lit
c cpair = heat capacity of ambient air, cal/g/C
c cond = thermal conductivity
c dca = change in volume fraction acid in solution between resin beads at each
c       calculation node
c den1 = density of pure HNO3, g/cc
c den1b = density of pure HNO3 in liquid volume below resin bed, g/cc
c den1t = density of pure HNO3 in liquid volume above resin bed, g/cc
c den2 = density of pure H2O, g/cc
c den2b = density of pure H2O in liquid volume below resin bed, g/cc
c den2t = density of pure H2O in liquid volume above resin bed, g/cc
c dens = liquid density, g/cc
c densb = liquid density in volume below resin bed, g/cc
c denst = liquid density in volume above resin bed, g/cc
c dfpor = pore diffusivity, cm2/s
c dfprt = particle diffusivity, cm2/s
c df = change in adsorbed plutonium concentration during feeding and elution,
c      g/cc
c diffc = molecular diffusivity of HNO3, cm2/s
c diffcb = molecular diffusivity of HNO3 in volume below resin bed, cm2/s
c diffct = molecular diffusivity of HNO3 in volume above resin bed, cm2/s
c diffy = molecular diffusivity of Pu, cm2/s
c diffyb = molecular diffusivity of Pu in volume below resin bed, cm2/s
c diffyt = molecular diffusivity of Pu in volume above resin bed, cm2/s
c dpart = resin bead diameter, cm
c dr = radial discretization step size, cm
c dt = time step, s
c dz = axial discretization step size, cm
```

```
c dy2 = change in plutonium concentration in solution between resin beads, g/cc
c dy2a = change in plutonium concentration in solution between resin beads at
c        each calculation node, g/cc
c emiss = surface emissivity for column
c f = plutonium loading on resin, g/cc
c fact1 = factor for iterative calculation of modified stream function
c fact2 = factor for iterative calculation of modified stream function
c fact3 = factor for iterative calculation of modified stream function
c fact4 = factor for iterative calculation of modified stream function
c fact5 = factor for iterative calculation of modified stream function
c fcx = weighting factor for elution of plutonium by acid dilution
c ffac = multiplying factor for rate of adsorption of plutonium by resin,
c        added to prevent adsorption rate from exceeding rates of convection
c        and dispersion mass transfer and thereby making calculation
numerically
c        unstable
c ffcx = weighting factor for elution of plutonium by acid dilution
c ffeed = maximum plutonium resin bed concentration, g/cc
c fold = plutonium loading on resin at previous time step, g/cc
c fpu = multiplier used to maintain constant plutonium content in column
c fx = total bulk plutonium concentration, g/cc
c grav = gravitational acceleration, cm/s2
c gzn = update value of modified stream function, cc/s
c gzo = value of stream function for previous iteration, cc/s
c hconv = coefficient for natural convection heat transfer to sides of column,
c         also overall convective heat transfer coefficient, cal/cm2/s/C
c hconv2 = coefficient for natural convection heat transfer to top and bottom
c          of column, cal/cm2/s/C
c hconv2b = overall coefficient for natural convection heat transfer to liquid
c           volume at bottom of column, cal/cm2/s/C
c hconv2t = overall coefficient for natural convection heat transfer to liquid
c           volume at top of column, cal/cm2/s/C
c hconv3 = coefficient for forced convection heat transfer to column, assuming
c          an ambient air velocity of 30 cm/s (1 ft/s)
c hrad = radiation heat transfer coefficient, cal/cm2/s/C
c htc = heat transfer coefficient, cal/cm2/s/C
c htc2b = overall heat transfer coefficient for liquid volume below resin bed,
c         cal/cm2/s/C
c htc2t = overall heat transfer coefficient for liquid volume above resin bed,
c         cal/cm2/s/C
c iht = indicator for heat transfer to ambient air (iht=1) or to water-cooled
c       jacket (iht=2) (A heat transfer coefficient is calculated for heat
c       transfer to ambient air, and a surface temperature is specified for
c       cooling by a water jacket.)
c indk = indicator that tells whether rate of adsorption of plutonium exceeds
c        rate of mass transfer due to convection and dispersion.  If true,
c        indk=1 and adsorption rate is halved.  If false, indk=0.
c ind3 = indicator that tells whether solution between the resin beads is
c       saturated with plutonium (If ind=1, solution is saturated and program
c       is directed to loading rate calculations in subroutine feeder; if ind=0,
c       solution is not saturated and program is directed to elution rate
c       calculations in subroutine elut.)
c ldown = depth of liquid in column below resin bed, cm
c lup = height of liquid in column above resin bed, cm
c length = column length, cm
c m = number of axial discretization nodes
c mlen = effective mixing length for axial dispersion coefficient, ratio of
c        dispersion coefficient to mixing velocity, cm
c mlen2 = mixing cell height for axial dispersion due to Benard cell mixing, cm
c n = number of radial discretization nodes
c nu = Nusselt number for natural convection heat transfer to ambient air from
c      sides of column
c nu2 = Nusselt number for natural convection heat transfer to ambient air from
```

```
c       top and bottom of column
c nu3 = Nusselt number for forced convection heat transfer to ambient air,
c       assuming air velocity of 30 cm/s (1 ft/s)
c phi = updated temperature, K
c poszn = distance from top of resin bed, cm
c pr = Prandtl number for ambient air
c pua = total amount of plutonium adsorbed on the resin beads, g
c puao = initial total amount of plutonium adsorbed on the resin beads, g
c puf = total amount of plutonium in the column, g
c pusn = total amount of plutonium in solution, g
c pusno = initial total amount of plutonium in solution, g
c put = amount of plutonium fed to the column, g
c qdot = heat production rate due to plutonium loading, cal/g Pu-238/s
c radius = column radius, cm
c rayl = Rayleigh number for natural convection heat transfer to ambient air
c        from sides of column
c rayl2 = Rayleigh number for natural convection heat transfer to ambient air
c         from top and bottom of column
c re = Reynolds number for forced convection heat transfer to column, assuming
c      an ambient air velocity of 30 cm/s (1 ft/s)
c rhcp = product of density and heat capacity for bulk resin bed, cal/cc/C
c rhcp2 = product of density and heat capacity for solution between
c         resin beads, cal/cc/C
c rhcp2b = product of density and heat capacity for liquid volume below resin
c          bed, cal/cc/C
c rhcp2t = product of density and heat capacity for liquid volume above resin
c          bed, cal/cc/C
c rho = bulk density, g/cc
c rhoair = density of air at resin column surface, g/cc
c rhoamb = density of ambient air, g/cc
c rrn = updated radial position, used in radial integrations of flow rates and
c       temperatures, cm
c rro = second previous radial position, cm
c rrr = previous radial position, cm
c rr1 = radial position of calculation node, cm
c r1 = radial position, cm
c r2 = radial position, cm
c sc1 = volume fraction acid in solution in liquid volume at top of column
c sc1a = change in volume fraction acid in solution in liquid volume at top of
c        column
c sc2 = volume fraction acid in solution in liquid volume at bottom of column
c sc2a = change in volume fraction acid in solution in liquid volume at bottom
c        of column
c sf = stream function for superficial velocities in resin bed, cm2/s
c sigma = Boltzmann's constant, cal/cm2/s/K4
c stem1 = temperature in the liquid volume above the resin bed, K
c stem1a = change in temperature in the liquid volumn above the resin bed, K
c stem1c = temperature in the liquid volumn above the resin bed, C
c stem2 = temperature in the liquid volume below the resin bed, K
c stem2a = change in temperature in the liquid volume below the resin bed, K
c stem2c = temperature in the liquid volume below the resin bed, C
c stp = total number of calculation time steps
c sumsf = integration parameter for stream function calculation, cc/s
c svz2 = integrated volumetric flow rate out column, cc/s
c sy21 = plutonium concentration in liquid volume at top of column, g/cc
c sy21a = change in plutonium concentration in liquid volume at top of column,
c         g/cc
c sy22 = plutonium concentration in liquid volumn at bottom of column, g/cc
c sy22a = change in plutonium concentration in liquid volume at bottom of
c         column, g/cc
c tamb = ambient temperature, K
c tau = time constant for loading plutonium on resin, s
c tau2 = time constant for eluting plutonium from resin, s
```

```
c tcn1 = thermal conductivity for nitric acid, cal/cm/s/C
c tcn1b = thermal conductivity for nitric acid in volume below resin bed,
c         cal/cm/s/C
c tcn1t = thermal conductivity for nitric acid in volume above resin bed,
c         cal/cm/s/C
c tcn2 = thermal conductivity for water, cal/cm/s/C
c tcn2b = thermal conductivity for water in volume below resin bed, cal/cm/s/C
c tcn2t = thermal conductivity for water in volume above resin bed, cal/cm/s/C
c tcnm = thermal conductivity for bulk resin bed, cal/cm/s/C
c tcnmb = thermal conductivity for liquid in volume below resin bed, cal/cm/s/C
c tcnmt = thermal conductivity for liquid in volume above resin bed, cal/cm/s/C
c telut = time for elution, s
c temb = product of volumetric flow rate out of resin bed and temperature at
c        bottom of resin bed, K-cc/s
c temch = maximum temperature in column, C
c temc = temperature, C
c temp = temperature, K
c time = elapsed time after flow is interrupted, m
c time2 = elapsed time after flow is interrupted, s
c tinte = time interval for printing maximum resin bed temperatures, s
c tinte2 = time for printing bed temperature profiles, s
c tload = time for loading, s
c twash = time for washing, s
c tzero = initial temperature, K
c tzero = temperature of solution entering column, K
c vbot = volume of liquid below resin bed, cc
c vel2 = velocity of ambient air, cm/s
c vfi = volume fraction in resin bead pores, prior to adsorption of plutonium
c vfi2 = volume fraction in resin bead pores, after adsorption of plutonium
c vfi3 = volume fraction in resin bead pores, adjusted for actual amount of
c        plutonium adsorbed
c vfo = volume fraction occupied by flowing solution
c vis = liquid dynamic viscosity, g/cm/s
c visb = dynamic viscosity for liquid volume below resin bed, g/cm/s
c vist = dynamic viscosity for liquid volume above resin bed, g/cm/s
c visair = dynamic viscosity of ambient air, g/cm/s
c vmid = total volume of resin bed, cc
c vr = superficial radial velocity, cm/s
c vtop = volume of liquid above resin bed, cc
c vz = superficial axial velocity, cm/s
c vzsum = integration parameter for integration of axial velocity profile, cc/s
c wf = weighting factor for axial dispersion coefficient, based on distance
c      over which density difference for natural convection velocity is based
c x = fractional loading of plutonium on resin
c xg = elution rate constant, 1/s
c xkair = thermal conductivity of ambient air, cal/cm/s/C
c xold = fractional loading of plutonium on resin at previous time step
c xx1 = mass fraction nitric acid in the solution between resin beads
c xx1b = mass fraction nitric acid in liquid volume below resin bed
c xx1t = mass fraction nitric acid in liquid volume above resin bed
c y = bulk plutonium concentration in pore solution, g/cc
c yold = bulk plutonium concentration in pore solution at previous step, g/cc
c y2 = bulk plutonium concentration in solution between resin beads, g/cc
c y2feed = plutonium concentration in solution entering resin bed, g/cc
c y2max = maximum bulk plutonium concentration in solution between resin beads
c         during elution, g/cc
c y2max2 = maximum plutonium concentration for a saturated plutonium nitrate
c          solution, gmole/lit
c y2mx2 = maximum bulk plutonium concentration in solution between resin beads
c         during elapsed time during elution, g/cc
c y2old = bulk plutonium concentration in solution between resin beads at
c         previous time step, g/cc
c zden = area-average liquid density at a given level, g/cc
```

```
c zvz = area-average superficial axial velocity at a given level, cm/s
      implicit double precision (a-z)
      integer i,j,m,mm1,n,nn1,q,r,s,stp,stp2,iskip,is1,mm2,mm3,jlev,
     1ind,istt,istc,ind3,k,nct,nct2,nct3,nlast,j4,jmax,jmin,mm4,mm5,
     2indk,k2
      character*10 ch1
      dimension temc(200,100),temp(200,100),phi(200,100)
      dimension vr(200,100),vz(200,100),dens(200,100),zden(200)
      dimension vis(200,100),zvz(200)
      dimension area(100),den2(200,100),br1(200,100),brc1(200,100)
      dimension rhcp(200,100),tcnm(200,100)
      dimension bz(200,100),br(200,100),fx(200,100)
      dimension rr1(100),bzc(200,100),brc(200,100)
      dimension bzy(200,100),bry(200,100),bry1(200,100)
      dimension rhcp2(200,100),diffy(200,100),diffc(200,100)
      dimension alpha(200,100),gzo(200,100),gzn(200,100)
      dimension dy2a(200,100),dca(200,100)
      dimension af1(100),af2(100),vzm(200)
      dimension c(200,100),cold(200,100),f(200,100),fold(200,100),
     1x(200,100),xold(200,100),y2(200,100),y2old(200,100),
     2den1(200,100),ind3(200,100),cmx2(200,100),
     3y2max(200,100)
      character*1 tab
      common /elut1/c,cold,f,fold,x,xold,y2,y2old,den1,
     2y2max,y2max2,cmax2,ind3,cmx2,cfeed,dt,tau,tau2,ffeed,an1,
     3dpart,dz,xg,vfi2,vfo,ind
      tab=char(9)
c Input ambient temperature in °C.
      read (15,*) tamb
c Input indicator for air heat transfer (iht=1) or water jacket heat transfer
c (iht=2).
      read (15,*) iht
c Input resin bed depth in centimeters.
      read (15,*) length
c Input height above resin bed in centimeters.
      read (15,*) lup
c Input depth below resin bed in centimeters.
      read (15,*) ldown
c Input column radius in centimeters.
      read (15,*) radius
c Input particle diameter in centimeters.
      read (15,*) dpart
c Input pore diffusivity in cm2/sec.
      read (15,*) dfpor
c Input particle diffusivity in cm2/sec.
      read (15,*) dfprt
c Input void fraction.
      read (15,*) vfo
c Input maximum resin bed concentration.
      read (15,*) ffeed
c Input radiolysis heat in cal/g Pu-238/sec.
      read (15,*) qdot
c Input initial volume fraction nitric acid.
      read (15,*) cf30
      pi=4.*atan(1.)
c Input parameters for mixing model.
      rho=1.23
      cp=0.76
      tzero=tzero+273.0
      tamb=tamb+273.0
      grav=980.
      vfi=0.132
      vfi2=0.160
```

```
      an1=1./(vfo+vfi)
      mlen=0.48*radius
      mlen2=1.6*radius
      tau=dpart**2/(4.*pi**2)/dfprt
      tau2=dpart**2/(24.*0.75*dfpor)
      xg=4.*pi**2*dfpor/dpart**2*(1.-vfo)
      aa=1.
      k=1
      do while (k.le.8)
      aa=aa-(aa**5-2.9**4*0.26*(4.-aa)*(1.-aa)**4)
     1/(4.*aa**4+2.9**4*0.26*((1.-aa)**4+4.*(4.-aa)*(1.-aa)**3))
      k=k+1
      enddo
      y2max2=aa/2.9/(4.-aa)/(1.-aa)
      cmax2=4.*y2max2
c Input # steps in axial, radial directions.
      read (15,*) m,n
c Input # of data points to skip between reading each concentration.
      read (15,*) iskip
c Input starting line for reading concentration profile.
      read (15,*) istc
      m=m+1
      n=n+1
      mm1=m-1
      nn1=n-1
      mm2=m+1
      mm3=m+2
      mm4=m-2
      mm5=m-3
c Read in temperatures.
      read (19,*) stem1c
      read (19,*) stem2c
      stem1=stem1c+273.
      stem2=stem2c+273.
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      read (19,*) temp(j,i)
      temp(j,i)=temp(j,i)+273.
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      read (19,*) temc(j,i)
      temp(j,i)=temc(j,i)+273.
      i=i+1
      enddo
      j=j-1
      enddo
c Read in concentrations.
      istc=istc-1
      k=1
      do while (k.le.istc)
      read (18,*)
      k=k+1
      enddo
      j=1
      do while (j.le.m)
      read (18,*) tim,ln,xc1,xc2,xc3,xc4,xc5
      i=1
      do while (i.le.n)
      c(j,i)=cf30*(1.-xc1)
```

```
      cold(j,i)=c(j,i)
      f(j,i)=xc2
      fold(j,i)=xc2
      y2(j,i)=xc3+xc4
      y2old(j,i)=xc3+xc4
      xold(j,i)=xc2/ffeed
      fx(j,i)=xc5
      i=i+1
      enddo
      j=j+1
      enddo
      read (18,*) tim,ln,xc1,xc2,xc3,xc4,xc5
      sc1=cf30*(1.-xc1)
      sy21=xc3
      read (18,*) tim,ln,xc1,xc2,xc3,xc4,xc5
      sc2=cf30*(1.-xc1)
      sy22=xc3
      dz=length/dfloat(mm1)
      dr=radius/dfloat(nn1)
      vtop=pi*radius**2*lup
      vbot=pi*radius**2*ldown
      vmid=pi*radius**2*length
      area(1)=pi*(0.5/dfloat(nn1)*radius)**2
      area(n)=pi*(1.-((dfloat(nn1)-0.5)/dfloat(nn1))**2)*radius**2
      i=2
      do while (i.le.nn1)
      r1=(dfloat(i)-1.5)/dfloat(nn1)*radius
      r2=(dfloat(i)-0.5)/dfloat(nn1)*radius
      area(i)=pi*(r2**2-r1**2)
      i=i+1
      enddo
      areat=pi*radius**2
c Input starting time increment in seconds.
      read (15,*) dts
c Input later time increment in seconds.
      read (15,*) dtf
c Input number of time steps for starting time increment.
      read (15,*) stp2
c Input total number of time steps.
      read (15,*) stp
c Initialize radial positions.
      rr1(1)=0.
      i=2
      do while (i.le.n)
      rr1(i)=rr1(i-1)+dr
      i=i+1
      enddo
      af1(1)=0.
      af2(1)=0.
      i=2
      do while (i.le.nn1)
      af1(i)=2.*rr1(i)/(rr1(i)**2-rr1(i-1)**2)
      af2(i)=2.*rr1(i)/(rr1(i+1)**2-rr1(i)**2)
      i=i+1
      enddo
c Initialize physical properties.
      stem1c=stem1-273.
      stem2c=stem2-273.
      den1t=1.6603-1.9894e-3*stem1c
      den1b=1.6603-1.9894e-3*stem2c
      den2t=0.99683-1.3010e-4*stem1c-2.4358e-6*stem1c**2
      den2b=0.99683-1.3010e-4*stem2c-2.4358e-6*stem2c**2
      xx1t=den1t*sc1/(den1t*sc1+den2t*(1.-sc1))
```

```
 xx1b=den1b*sc2/(den1b*sc2+den2b*(1.-sc2))
 cno3t=sc1*den1t*1000./63.-4.*sy21*1000./238.
 cno3b=sc2*den1b*1000./63.-4.*sy22*1000./238.
 if (cno3t.lt.0.) cno3t=0.
 if (cno3b.lt.0.) cno3b=0.
 denst=den2t+0.4277*sy21*1000./238.+0.031*cno3t
 densb=den2b+0.4277*sy22*1000./238.+0.031*cno3b
 capst=1.0104-(1.419-0.3*(stem1c-20.)/80.)*xx1t
1+(2.005-0.6*(stem1c-20.)/80.)*xx1t**2
2-(1.147-0.3*(stem1c-20.)/80.)*xx1t**3
 capsb=1.0104-(1.419-0.3*(stem2c-20.)/80.)*xx1b
1+(2.005-0.6*(stem2c-20.)/80.)*xx1b**2
2-(1.147-0.3*(stem2c-20.)/80.)*xx1b**3
 rhcp2t=denst*capst
 rhcp2b=densb*capsb
 tcn1t=6.1388e-4+1.3951e-6*stem1c
 tcn2t=1.3518e-3+2.7903e-6*stem1c
 tcn1b=6.1388e-4+1.3951e-6*stem2c
 tcn2b=1.3518e-3+2.7903e-6*stem2c
 tcnmt=tcn1t*xx1t+tcn2t*(1.-xx1t)
 tcnmb=tcn1b*xx1t+tcn2b*(1.-xx1t)
 vist=0.01002*10.**((1.3272*(20.-stem1c)-0.001053*
1(stem1c-20.)**2)/(stem1c+105.))
 visb=0.01002*10.**((1.3272*(20.-stem2c)-0.001053*
1(stem2c-20.)**2)/(stem2c+105.))
 diffyt=8.314/96500**2*stem1*(1./1.+1./3.)/(1./69.5
1+1./71.4)
 diffct=8.314/96500**2*stem1*(1./1.+1./1.)/(1./349.8
1+1./71.4)
 diffyb=8.314/96500**2*stem2*(1./1.+1./3.)/(1./69.5
1+1./71.4)
 diffcb=8.314/96500**2*stem2*(1./1.+1./1.)/(1./349.8
1+1./71.4)
 i=1
 do while (i.le.n)
 j=1
 do while (j.le.m)
 temc(j,i)=temp(j,i)-273.
 den1(j,i)=1.6603-1.9894e-3*temc(j,i)
 den2(j,i)=0.99683-1.3010e-4*temc(j,i)-2.4358e-6*temc(j,i)**2
 xx1=den1(j,i)*c(j,i)/(den1(j,i)*c(j,i)+den2(j,i)*(1.-c(j,i)))
 cno3=c(j,i)*den1(j,i)*1000./63.-4.*y2(j,i)*1000./238./vfo
 if (cno3.lt.0.) cno3=0.
 dens(j,i)=den2(j,i)+0.4277*y2(j,i)*1000./238./vfo+0.031*cno3
 tcn1=6.1388e-4+1.3951e-6*temc(j,i)
 tcn2=1.3518e-3+2.7903e-6*temc(j,i)
 tcnm(j,i)=tcn1*xx1+tcn2*(1.-xx1)
 tcnm(j,i)=0.5*(5.e-4+tcnm(j,i))
 caps=1.0104-(1.419-0.3*(temc(j,i)-20.)/80.)*xx1
1+(2.005-0.6*(temc(j,i)-20.)/80.)*xx1**2
2-(1.147-0.3*(temc(j,i)-20.)/80.)*xx1**3
 rhcp(j,i)=0.5*1.25*0.52+0.5*dens(j,i)*caps
 rhcp2(j,i)=dens(j,i)*caps
 vis(j,i)=0.01002*10.**((1.3272*(20.-temc(j,i))-0.001053*
1(temc(j,i)-20.)**2)/(temc(j,i)+105.))
 alpha(j,i)=tcnm(j,i)/rhcp(j,i)
 diffy(j,i)=8.314/96500**2*temp(j,i)*(1./1.+1./3.)/(1./69.5
1+1./71.4)
 diffc(j,i)=8.314/96500**2*temp(j,i)*(1./1.+1./1.)/(1./349.8
1+1./71.4)
 bz(j,i)=(alpha(j,i)+2.*dpart*abs(vz(j,i))*rhcp2(j,i)/rhcp(j,i))
1*dt/dz**2
 bzy(j,i)=(diffy(j,i)+2.*dpart*abs(vz(j,i)))*dt/dz**2
```

```
       bzc(j,i)=(diffc(j,i)+2.*dpart*abs(vz(j,i)))*dt/dz**2
       j=j+1
       enddo
       i=i+1
       enddo
       j=1
       do while (j.le.m)
       br(j,1)=(alpha(j,1)+0.4*dpart*abs(vz(j,1))*rhcp2(j,1)/rhcp(j,1))
      1*dt*4./dr**2
       br1(j,1)=0.
       bry(j,1)=(diffy(j,1)+0.4*dpart*abs(vz(j,1)))*dt*4./dr**2
       bry1(j,1)=0.
       brc(j,1)=(diffc(j,1)+0.4*dpart*abs(vz(j,1)))*dt*4./dr**2
       brc1(j,1)=0.
       j=j+1
       enddo
       i=2
       do while (i.le.n)
       j=1
       do while (j.le.m)
       br(j,i)=(alpha(j,i)+0.4*dpart*abs(vz(j,i))*rhcp2(j,i)
      1/rhcp(j,i))*dt/dr**2
       br1(j,i)=(alpha(j,i)+0.4*dpart*abs(vz(j,i))*rhcp2(j,i)
      1/rhcp(j,i))*dt/dr/rr1(i)/2.
       bry(j,i)=(diffy(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr**2
       bry1(j,i)=(diffy(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr/rr1(i)/2.
       brc(j,i)=(diffc(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr**2
       brc1(j,i)=(diffc(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr/rr1(i)/2.
       j=j+1
       enddo
       i=i+1
       enddo
c Write input variables to output.
       write (11,100) length,radius,m,n
       write (11,121) dpart,vfo
       write (11,101) dt,htc,htc2
c      write (11,110) sumf
  100 format ('L (CM) =',f7.3,2x,'R (CM) =',f7.3,2x,'AXIAL NODES =',
      1i3,2x,'RADIAL NODES =',i3)
  101 format ('DT (S) =',f8.4,2x,'HSID (CAL/CM2/S/°C) =',f9.6,
      12x,'HTOP (CAL/CM2/S/°C) =',f9.6)
c  110 format ('PU CHARGE (GM) =',f11.4)
       write (12,*) 'TIME (MIN)',tab,'TMAX (°C)',tab,'PUA (GM)',
      1tab,'PUT (GM)'
  121 format ('PART DIAM (CM) =',f8.4,2x,'VOID FRACTION =',f7.4)
c Find highest temperature.
       temch=0.
       j=1
       do while (j.le.m)
       i=1
       do while (i.le.n)
       if (temch.lt.temc(j,i)) then
       temch=temc(j,i)
       endif
       i=i+1
       enddo
       j=j+1
       enddo
c Calculate total amounts of plutonium and plutonium adsorbed.
       pua=0.
       puf=areat*(lup*sy21+ldown*sy22)
       pusn=areat*(lup*sy21+ldown*sy22)
       i=1
```

```
      do while (i.le.n)
      pua=pua+0.5*dz*area(i)*(f(1,i)+f(m,i))
      puf=puf+0.5*dz*area(i)*(fx(1,i)+fx(m,i))
      pusn=pusn+0.5*dz*area(i)*(y2(1,i)+y2(m,i))
      i=i+1
      enddo
      j=2
      do while (j.le.mm1)
      i=1
      do while (i.le.n)
      pua=pua+dz*f(j,i)*area(i)
      puf=puf+dz*fx(j,i)*area(i)
      pusn=pusn+dz*y2(j,i)*area(i)
      i=i+1
      enddo
      j=j+1
      enddo
      puao=pua
      pusno=pusn
      time=0.
      if (abs(nint(time/2)-time/2).lt.0.001)
     1write (12,301) time,tab,temch,tab,pua,tab,puf
c Set initial velocities equal to zero.
      j=1
      do while (j.le.m)
      vzm(j)=0.
      i=1
      do while (i.le.n)
      vz(j,i)=0.
      vr(j,i)=0.
      i=i+1
      enddo
      j=j+1
      enddo
      i=1
c Write initial concentrations and temperatures.
      write (11,*) 'TIME (MIN)',tab,'POS (CM)',tab,'LOAD',tab,
     1'TCEN (°C)',tab,'TMID (°C)',tab,'TOUT (°C)'
      write (16,*) 'TIME (MIN)',tab,'LENGTH (CM)',tab,'ELU (VF)',tab,
     1'ADS (G/CC)',tab,'LIQ (G/CC)',tab,'TOT (G/CC)'
      j=1
      write (11,102) time,tab,-1.,tab,0.,tab,stem1c,tab,stem1c,
     1tab,stem1c
      do while (j.le.m)
      poszn=length*(j-1)/(m-1)
      write (11,102) time,tab,poszn,tab,f(j,1),tab,temc(j,1),tab,
     1temc(j,5),tab,temc(j,n)
      write (16,400) time,tab,poszn,tab,c(j,1),tab,f(j,1),tab,y2(j,1),
     1tab,fx(j,1)
      j=j+1
      enddo
      write (11,102) time,tab,-2.,tab,0.,tab,stem2c,tab,stem2c,tab,
     1stem2c
c Perform transient calculations.
      indk=1
      s=1
      time2=0.
      dt=dts
      do while (s.le.stp.and.vtop.gt.5.)
      if (s.gt.stp2) dt=dtf
c Calculate heat transfer coefficients
      sigma=1.356e-12
      emiss=0.6
```

```
 tavg=0.
 q=1
 do while (q.le.m)
 tavg=tavg+temp(q,n)
 q=q+1
 enddo
 tavg=tavg/m
 hrad=emiss*sigma*(tavg**3+tavg**2*tamb
1+tavg*tamb**2+tamb**3)
 cpair=3.5*1.9872*29.
 xkair=0.0262/100*0.2389
 visair=0.00018
 pr=cpair*visair/xkair
 rhoamb=29./82.057/tamb
 rhoair=29./82.057/tavg
 rayl=grav*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
1/visair**2*pr
 if (rayl.gt.0.) then
 nu=0.59*rayl**0.25
 endif
 if (nu.lt.1.) nu=1.
 hconv=nu*xkair/(length+lup+ldown)
 vel2=30.0
 re=2.*radius*vel2*rhoamb/visair
 nu3=0.683*re**0.466*pr**(1./3.)
 hconv3=xkair*nu3/2./radius
 if (hconv.lt.hconv3) hconv=hconv3
 htc=hconv+hrad
 hrad=emiss*sigma*(stem1**3+stem1**2*tamb+stem1*tamb**2+tamb**3)
 rhoair=29./82.057/stem1
 rayl=grav*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
1/visair**2*pr
 if (rayl.gt.0.) then
 nu=0.59*rayl**0.25
 endif
 if (nu.lt.1.) nu=1.
 hconv=nu*xkair/(length+lup+ldown)
 rayl2=grav*(2.*radius)**3*rhoamb*(rhoamb-rhoair)/visair**2*pr
 if (rayl2.gt.0.) then
 nu2=0.54*rayl2**0.25
 endif
 if (nu2.lt.1.) nu2=1.
 hconv2=nu2*xkair/2./radius
 hconv2t=(2.*radius*lup*hconv+radius**2*hconv2)/(2.*radius*lup
1+radius**2)
 vel2=30.0
 re=2.*radius*vel2*rhoamb/visair
 nu3=0.683*re**0.466*pr**(1./3.)
 hconv3=xkair*nu3/2./radius
 if (hconv2t.lt.hconv3) hconv2t=hconv3
 htc2t=hrad+hconv2t
 hrad=emiss*sigma*(stem2**3+stem2**2*tamb+stem2*tamb**2+tamb**3)
 rhoair=29./82.057/stem2
 rayl=grav*(length+lup+ldown)**3*rhoamb*(rhoamb-rhoair)
1/visair**2*pr
 if (rayl.gt.0.) then
 nu=0.59*rayl**0.25
 endif
 if (nu.lt.1.) nu=1.
 hconv=nu*xkair/(length+lup+ldown)
 rayl2=grav*(2.*radius)**3*rhoamb*(rhoamb-rhoair)/visair**2*pr
 if (rayl2.gt.0.) then
 nu2=0.27*rayl2**0.25
```

```
       endif
       if (nu2.lt.1.) nu2=1.
       hconv2=nu2*xkair/2./radius
       hconv2b=(2.*radius*ldown*hconv+radius**2*hconv2)/(2.*radius*ldown
      1+radius**2)
       vel2=30.0
       re=2.*radius*vel2*rhoamb/visair
       nu3=0.683*re**0.466*pr**(1./3.)
       hconv3=xkair*nu3/2./radius
       if (hconv2b.lt.hconv3) hconv2b=hconv3
       htc2b=hrad+hconv2b
c Calculate changes in temperatures and concentrations in the liquid volumes
c above and below the resin bed.
       sy21a=0.
       sy22a=0.
       sc1a=0.
       sc2a=0.
       stem1a=sy21*0.89*qdot*dt/rhcp2t
       stem2a=sy22*0.89*qdot*dt/rhcp2b
       i=1
       do while (i.le.n)
       stem1a=stem1a+(bz(1,i)*dz/lup+vzm(1)*dt/lup/dz)
      1*area(i)/areat*(temp(2,i)-stem1)
       stem2a=stem2a+(bz(m,i)*dz/ldown+vzm(m)*dt/ldown/dz)
      1*area(i)/areat*(temp(mm1,i)-stem2)
       sy21a=sy21a+(bzy(1,i)*dz/lup+vzm(1)*dt/lup/dz)
      1*area(i)/areat*(y2old(2,i)/vfo-sy21)
       sy22a=sy22a+(bzy(m,i)*dz/ldown+vzm(m)*dt/ldown/dz)
      1*area(i)/areat*(y2old(mm1,i)/vfo-sy22)
       sc1a=sc1a+(bzc(1,i)*dz/lup+vzm(1)*dt/lup/dz)
      1*area(i)/areat*(cold(2,i)-sc1)
       sc2a=sc2a+(bzc(m,i)*dz/ldown+vzm(m)*dt/ldown/dz)
      1*area(i)/areat*(cold(mm1,i)-sc2)
       i=i+1
       enddo
       stem1a=stem1a+htc2t*(radius**2+2.*lup*radius)/lup/radius**2
      1*dt/rhcp2t*(tamb-stem1)
       stem2a=stem2a+htc2b*(radius**2+2.*ldown*radius)/ldown/radius**2
      1*dt/rhcp2b*(tamb-stem2)
c Initialize changes in concentrations.
       j=1
       do while (j.le.m)
       i=1
       do while (i.le.n)
       dy2a(j,i)=0.
       dca(j,i)=0.
       i=i+1
       enddo
       j=j+1
       enddo
c Calculate temperature and concentration changes in middle of column.
       q=2
       do while (q.le.mm1)
       phi(q,n)=temp(q,n)+fx(q,n)*0.89*qdot*dt/rhcp(q,n)
       phi(q,n)=phi(q,n)+bz(q,n)*(temp(q+1,n)+temp(q-1,n)-2.*temp(q,n))
      1+htc*dt/rhcp(q,n)*(1./dr+1./2./rr1(n))*(tamb-temp(q,n))
      2+br(q,nn1)*(temp(q,nn1)-temp(q,n))
      3+2.*br1(q,n)*(temp(q,n)-temp(q,nn1))
      4+vzm(q)*rhcp2(q,n)/rhcp(q,n)*dt/dz**2*(temp(q+1,n)+temp(q-1,n)
      5-2.*temp(q,n))
       dy2a(q,n)=dy2a(q,n)+bzy(q,n)/vfo*(y2old(q+1,n)+y2old(q-1,n)
      1-2.*y2old(q,n))
      2+bry(q,nn1)/vfo*(y2old(q,nn1)-y2old(q,n))
```

```
3+2.*bry1(q,n)/vfo*(y2old(q,n)-y2old(q,nn1))
4+vzm(q)/vfo*dt/dz**2*(y2old(q+1,n)+y2old(q-1,n)-2.*y2old(q,n))
 dca(q,n)=dca(q,n)+an1*(bzc(q,n)*(cold(q+1,n)+cold(q-1,n)
1-2.*cold(q,n))
2+brc(q,nn1)*(cold(q,nn1)-cold(q,n))
3+2.*brc1(q,n)*(cold(q,n)-cold(q,nn1))
4+vzm(q)*dt/dz**2*(cold(q+1,n)+cold(q-1,n)-2.*cold(q,n)))
 if (vz(q,n).gt.0.) then
 phi(q,n)=phi(q,n)-vz(q,n)*dt/dz*rhcp2(q,n)/rhcp(q,n)
1*(temp(q,n)-temp(q-1,n))
 dy2a(q,n)=dy2a(q,n)-vz(q,n)*dt/dz/vfo*(y2old(q,n)-y2old(q-1,n))
 dca(q,n)=dca(q,n)-an1*vz(q,n)*dt/dz*(cold(q,n)-cold(q-1,n))
 endif
 if (vz(q,n).lt.0.) then
 phi(q,n)=phi(q,n)-vz(q,n)*dt/dz*rhcp2(q,n)/rhcp(q,n)
1*(temp(q+1,n)-temp(q,n))
 dy2a(q,n)=dy2a(q,n)-vz(q,n)*dt/dz/vfo*(y2old(q+1,n)-y2old(q,n))
 dca(q,n)=dca(q,n)-an1*vz(q,n)*dt/dz*(cold(q+1,n)-cold(q,n))
 endif
 q=q+1
 enddo
 q=2
 do while (q.le.mm1)
 r=2
 do while (r.le.nn1)
 phi(q,r)=temp(q,r)+fx(q,r)*0.89*qdot*dt/rhcp(q,r)
 phi(q,r)=phi(q,r)+bz(q,r)*(temp(q+1,r)+temp(q-1,r)-2.*temp(q,r))
1+br(q,r-1)*(temp(q,r-1)-temp(q,r))
2+br(q,r)*(temp(q,r+1)-temp(q,r))
3+br1(q,r)*(temp(q,r+1)-temp(q,r-1))
4+vzm(q)*rhcp2(q,r)/rhcp(q,r)*dt/dz**2*(temp(q+1,r)+temp(q-1,r)
5-2.*temp(q,r))
 dy2a(q,r)=dy2a(q,r)+bzc(q,r)/vfo*(y2old(q+1,r)+y2old(q-1,r)
1-2.*y2old(q,r))
2+bry(q,r-1)/vfo*(y2old(q,r-1)-y2old(q,r))
3+bry(q,r)/vfo*(y2old(q,r+1)-y2old(q,r))
4+bry1(q,r)/vfo*(y2old(q,r+1)-y2old(q,r-1))
5+vzm(q)/vfo*dt/dz**2*(y2old(q+1,r)+y2old(q-1,r)-2.*y2old(q,r))
 dca(q,r)=dca(q,r)+an1*(bzc(q,r)*(cold(q+1,r)+cold(q-1,r)
1-2.*cold(q,r))
2+brc(q,r-1)*(cold(q,r-1)-cold(q,r))
3+brc(q,r)*(cold(q,r+1)-cold(q,r))
4+brc1(q,r)*(cold(q,r+1)-cold(q,r-1))
5+vzm(q)*dt/dz**2*(cold(q+1,r)+cold(q-1,r)-2.*cold(q,r)))
 if (vz(q,r).gt.0.) then
 phi(q,r)=phi(q,r)-vz(q,r)*dt/dz*rhcp2(q,r)/rhcp(q,r)
1*(temp(q,r)-temp(q-1,r))
 dy2a(q,r)=dy2a(q,r)-vz(q,r)*dt/dz/vfo*(y2old(q,r)-y2old(q-1,r))
 dca(q,r)=dca(q,r)-an1*vz(q,r)*dt/dz*(cold(q,r)-cold(q-1,r))
 endif
 if (vz(q,r).lt.0.) then
 phi(q,r)=phi(q,r)-vz(q,r)*dt/dz*rhcp2(q,r)/rhcp(q,r)
1*(temp(q+1,r)-temp(q,r))
 dy2a(q,r)=dy2a(q,r)-vz(q,r)*dt/dz/vfo*(y2old(q+1,r)-y2old(q,r))
 dca(q,r)=dca(q,r)-an1*vz(q,r)*dt/dz*(cold(q+1,r)-cold(q,r))
 endif
 if (vr(q,r).gt.0.) then
 phi(q,r+1)=phi(q,r+1)-vr(q,r)*dt*af2(r)*rhcp2(q,r)/rhcp(q,r)
1*(temp(q,r+1)-temp(q,r))
 dy2a(q,r+1)=dy2a(q,r+1)-vr(q,r)*dt*af2(r)/vfo
1*(y2old(q,r+1)-y2old(q,r))
 dca(q,r+1)=dca(q,r+1)-an1*vr(q,r)*dt*af2(r)
1*(cold(q,r+1)-cold(q,r))
```

```
       endif
       if (vr(q,r).lt.0.) then
       phi(q,r-1)=phi(q,r-1)-vr(q,r)*dt*af1(r)*rhcp2(q,r)/rhcp(q,r)
      1*(temp(q,r)-temp(q,r-1))
       dy2a(q,r-1)=dy2a(q,r-1)-vr(q,r)*dt*af1(r)/vfo
      1*(y2old(q,r)-y2old(q,r-1))
       dca(q,r-1)=dca(q,r-1)-an1*vr(q,r)*dt*af1(r)
      1*(cold(q,r)-cold(q,r-1))
       endif
       r=r+1
       enddo
       q=q+1
       enddo
       j=1
       do while (j.le.m)
       i=1
       do while (i.le.n)
       if (y2old(j,i)+dy2a(j,i).lt.0.) then
       dy2a(j,i)=-y2old(j,i)
       endif
       i=i+1
       enddo
       j=j+1
       enddo
c Calculate changes in plutonium concentrations due to adsorption.
c This is accomplished by calling the subroutine crite to determine if the bulk
c solution is saturated with plutonium.  If the solution is saturated, the
c subroutine feeder is called to calculate the adsorption rate.
       q=2
       do while (q.le.mm1)
       r=2
       do while (r.le.n)
       call crite (q,r,m)
       r=r+1
       enddo
       q=q+1
       enddo
c Calculate the amount of plutonium adsorbed by calling subroutine feeder.  A
c check is added to ensure that the rate of adsorption does not exceed the
c rate of mass transfer by convection and dispersion.  If this should occur,
c numerical instabilities would result.
       indk=1
       ffac=1.
       k2=1
       do while (k2.le.20.and.indk.eq.1)
       q=2
       do while (q.le.mm1)
       r=2
       do while (r.le.n)
       dy2=dy2a(q,r)
       dc=dca(q,r)
       if (ind3(q,r).eq.1) then
       call feeder (q,r,m,dy2,dc,df,ffac)
       endif
       f(q,r)=fold(q,r)+df
       y2(q,r)=y2old(q,r)+dy2
       c(q,r)=cold(q,r)+dc
       x(q,r)=xold(q,r)+df/ffeed
       r=r+1
       enddo
       q=q+1
       enddo
       indk=0
```

```
      i=2
      do while (i.le.n)
      j=1
      do while (j.le.m-3)
      if (y2(j,i).lt.y2(j+1,i).and.y2(j+1,i).gt.y2(j+2,i).and.
     1y2(j+2,i).lt.y2(j+3,i)) indk=1
      if (y2(j,i).gt.y2(j+1,i).and.y2(j+1,i).lt.y2(j+2,i).and.
     1y2(j+2,i).gt.y2(j+3,i)) indk=1
      if (c(j,i).lt.c(j+1,i).and.c(j+1,i).gt.c(j+2,i).and.
     1c(j+2,i).lt.c(j+3,i)) indk=1
      if (c(j,i).gt.c(j+1,i).and.c(j+1,i).lt.c(j+2,i).and.
     1c(j+2,i).gt.c(j+3,i)) indk=1
      j=j+1
      enddo
      i=i+1
      enddo
      if (indk.eq.1) ffac=ffac/2.
      k2=k2+1
      enddo
c Update conditions in liquid layers at top and bottom of resin bed.
      stem1=stem1+stem1a
      sy21=sy21+sy21a
      sc1=sc1+sc1a
      stem2=stem2+stem2a
      sy22=sy22+sy22a
      sc2=sc2+sc2a
c Specify symmetry condition at column center.
      j=1
      do while (j.le.m)
      phi(j,1)=phi(j,2)
      temp(j,1)=temp(j,2)
      temc(j,1)=temc(j,2)
      c(j,1)=c(j,2)
      cold(j,1)=cold(j,2)
      y2(j,1)=y2(j,2)
      y2old(j,1)=y2old(j,2)
      f(j,1)=f(j,2)
      fold(j,1)=fold(j,2)
      fx(j,1)=fx(j,2)
      xold(j,1)=xold(j,2)
      j=j+1
      enddo
c Specify top and bottom conditions.
      i=1
      do while (i.le.n)
      phi(1,i)=stem1
      y2(1,i)=sy21*vfo
      c(1,i)=sc1
      phi(m,i)=stem2
      y2(m,i)=sy22*vfo
      c(m,i)=sc2
      i=i+1
      enddo
c Update temperatures and concentrations.  Calculate densities and viscosities.
      stem1c=stem1-273.
      stem2c=stem2-273.
      den1t=1.6603-1.9894e-3*stem1c
      den1b=1.6603-1.9894e-3*stem2c
      den2t=0.99683-1.3010e-4*stem1c-2.4358e-6*stem1c**2
      den2b=0.99683-1.3010e-4*stem2c-2.4358e-6*stem2c**2
      xx1t=den1t*sc1/(den1t*sc1+den2t*(1.-sc1))
      xx1b=den1b*sc2/(den1b*sc2+den2b*(1.-sc2))
      cno3t=sc1*den1t*1000./63.-4.*sy21*1000./238.
```

```
 cno3b=sc2*den1b*1000./63.-4.*sy22*1000./238.
 if (cno3t.lt.0.) cno3t=0.
 if (cno3b.lt.0.) cno3b=0.
 denst=den2t+0.4277*sy21*1000./238.+0.031*cno3t
 densb=den2b+0.4277*sy22*1000./238.+0.031*cno3b
 capst=1.0104-(1.419-0.3*(stem1c-20.)/80.)*xx1t
1+(2.005-0.6*(stem1c-20.)/80.)*xx1t**2
2-(1.147-0.3*(stem1c-20.)/80.)*xx1t**3
 capsb=1.0104-(1.419-0.3*(stem2c-20.)/80.)*xx1b
1+(2.005-0.6*(stem2c-20.)/80.)*xx1b**2
2-(1.147-0.3*(stem2c-20.)/80.)*xx1b**3
 rhcp2t=denst*capst
 rhcp2b=densb*capsb
 tcn1t=6.1388e-4+1.3951e-6*stem1c
 tcn2t=1.3518e-3+2.7903e-6*stem1c
 tcn1b=6.1388e-4+1.3951e-6*stem2c
 tcn2b=1.3518e-3+2.7903e-6*stem2c
 tcnmt=tcn1t*xx1t+tcn2t*(1.-xx1t)
 tcnmb=tcn1b*xx1t+tcn2b*(1.-xx1t)
 vist=0.01002*10.**((1.3272*(20.-stem1c)-0.001053*
1(stem1c-20.)**2)/(stem1c+105.))
 visb=0.01002*10.**((1.3272*(20.-stem2c)-0.001053*
1(stem2c-20.)**2)/(stem2c+105.))
 diffyt=8.314/96500**2*stem1*(1./1.+1./3.)/(1./69.5
1+1./71.4)
 diffct=8.314/96500**2*stem1*(1./1.+1./1.)/(1./349.8
1+1./71.4)
 diffyb=8.314/96500**2*stem2*(1./1.+1./3.)/(1./69.5
1+1./71.4)
 diffcb=8.314/96500**2*stem2*(1./1.+1./1.)/(1./349.8
1+1./71.4)
 i=1
 do while (i.le.n)
 j=1
 do while (j.le.m)
 temp(j,i)=phi(j,i)
 if (iht.eq.2.and.i.eq.n) temp(j,i)=tamb
 temc(j,i)=phi(j,i)-273.0
 if (iht.eq.2.and.i.eq.n) temc(j,i)=tamb-273.0
 cold(j,i)=c(j,i)
 fold(j,i)=f(j,i)
 y2old(j,i)=y2(j,i)
 fx(j,i)=f(j,i)+y2(j,i)
 xold(j,i)=x(j,i)
 den1(j,i)=1.6603-1.9894e-3*temc(j,i)
 den2(j,i)=0.99683-1.3010e-4*temc(j,i)-2.4358e-6*temc(j,i)**2
 xx1=den1(j,i)*c(j,i)/(den1(j,i)*c(j,i)+den2(j,i)*(1.-c(j,i)))
 cno3=c(j,i)*den1(j,i)*1000./63.-4.*y2(j,i)*1000./238./vfo
 if (cno3.lt.0.) cno3=0.
 dens(j,i)=den2(j,i)+0.4277*y2(j,i)*1000./238./vfo+0.031*cno3
 tcn1=6.1388e-4+1.3951e-6*temc(j,i)
 tcn2=1.3518e-3+2.7903e-6*temc(j,i)
 tcnm(j,i)=tcn1*xx1+tcn2*(1.-xx1)
 tcnm(j,i)=0.5*(tcnm(j,i)+5.e-4)
 caps=1.0104-(1.419-0.3*(temc(j,i)-20.)/80.)*xx1
1+(2.005-0.6*(temc(j,i)-20.)/80.)*xx1**2
2-(1.147-0.3*(temc(j,i)-20.)/80.)*xx1**3
 rhcp(j,i)=0.5*1.25*0.52+0.5*dens(j,i)*caps
 rhcp2(j,i)=dens(j,i)*caps
 vis(j,i)=0.01002*10.**((1.3272*(20.-temc(j,i))-0.001053*
1(temc(j,i)-20.)**2)/(temc(j,i)+105.))
 alpha(j,i)=tcnm(j,i)/rhcp(j,i)
 diffy(j,i)=8.314/96500**2*temp(j,i)*(1./1.+1./3.)/(1./69.5
```

```
     1+1./71.4)
      diffc(j,i)=8.314/96500**2*temp(j,i)*(1./1.+1./1.)/(1./349.8
     1+1./71.4)
      bz(j,i)=(alpha(j,i)+2.*dpart*abs(vz(j,i))*rhcp2(j,i)/rhcp(j,i))
     1*dt/dz**2
      bzy(j,i)=(diffy(j,i)+2.*dpart*abs(vz(j,i)))*dt/dz**2
      bzc(j,i)=(diffc(j,i)+2.*dpart*abs(vz(j,i)))*dt/dz**2
      j=j+1
      enddo
      i=i+1
      enddo
      j=1
      do while (j.le.m)
      br(j,1)=(alpha(j,1)+0.4*dpart*abs(vz(j,1))*rhcp2(j,1)/rhcp(j,1))
     1*dt*4./dr**2
      br1(j,1)=0.
      bry(j,1)=(diffy(j,1)+0.4*dpart*abs(vz(j,1)))*dt*4./dr**2
      bry1(j,1)=0.
      brc(j,1)=(diffc(j,1)+0.4*dpart*abs(vz(j,1)))*dt*4./dr**2
      brc1(j,1)=0.
      j=j+1
      enddo
      i=2
      do while (i.le.n)
      j=1
      do while (j.le.m)
      br(j,i)=(alpha(j,i)+0.4*dpart*abs(vz(j,i))*rhcp2(j,i)
     1/rhcp(j,i))*dt/dr**2
      br1(j,i)=(alpha(j,i)+0.4*dpart*abs(vz(j,i))*rhcp2(j,i)
     1/rhcp(j,i))*dt/dr/rr1(i)/2.
      bry(j,i)=(diffy(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr**2
      bry1(j,i)=(diffy(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr/rr1(i)/2.
      brc(j,i)=(diffc(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr**2
      brc1(j,i)=(diffc(j,i)+0.4*dpart*abs(vz(j,i)))*dt/dr/rr1(i)/2.
      j=j+1
      enddo
      i=i+1
      enddo
c Calculate the average density at each level.
      j=1
      do while (j.le.m)
      rrr=dr/2.
      dsum=rrr**2*dens(j,1)
      i=2
      do while (i.le.nn1)
      rro=rrr
      rrr=rrr+dr
      dsum=dsum+(rrr**2-rro**2)*dens(j,i)
      i=i+1
      enddo
      rro=rrr
      rrr=rrr+dr/2.
      dsum=dsum+(rrr**2-rro**2)*dens(j,n)
      zden(j)=dsum/rrr**2
      j=j+1
      enddo
c Calculate superficial velocities.  This calculation is performed using a
c Richardson iteration.
      j=1
      do while (j.le.m)
      gzo(j,1)=0.
      gzo(j,n)=0.
      j=j+1
```

```
       enddo
       fact1=1./(2./dr**2+2./dz**2)
       fact2=1./dz**2
       k=1
       do while (k.le.200)
       i=2
       do while (i.le.nn1)
       fact3=1./dr**2-0.5/rr1(i)/dr
       fact4=1./dr**2+0.5/rr1(i)/dr
       fact5=dpart**2*vfo**4/((1.-vfo)**2*200.)*grav*rr1(i)/2./dr
       gzn(1,i)=fact1*(fact2*(gzo(3,i)-2.*gzo(2,i)+3.*gzo(1,i))
      1+fact3*gzo(1,i+1)+fact4*gzo(1,i-1)
      2+fact5/vis(1,i)*(dens(1,i+1)-dens(1,i-1)))
       gzn(m,i)=fact1*(fact2*(gzo(mm4,i)-2.*gzo(mm1,i)+3.*gzo(m,i))
      1+fact3*gzo(m,i+1)+fact4*gzo(m,i-1)
      2+fact5/vis(m,i)*(dens(m,i+1)-dens(m,i-1)))
       j=2
       do while (j.le.mm1)
       gzn(j,i)=fact1*(fact2*(gzo(j+1,i)+gzo(j-1,i))+fact3*gzo(j,i+1)
      1+fact4*gzo(j,i-1)+fact5/vis(j,i)*(dens(j,i+1)-dens(j,i-1)))
       j=j+1
       enddo
       i=i+1
       enddo
       i=2
       do while (i.le.nn1)
       j=2
       do while (j.le.mm1)
       gzo(j,i)=gzn(j,i)
       j=j+1
       enddo
       i=i+1
       enddo
       k=k+1
       enddo
       i=2
       do while (i.le.nn1)
       vr(1,i)=(gzo(2,i)-gzo(1,i))/dz/rr1(i)
       vr(m,i)=(gzo(m,i)-gzo(mm1,i))/dz/rr1(i)
       vz(1,i)=(gzo(1,i-1)-gzo(1,i+1))/2./dr/rr1(i)
       vz(m,i)=(gzo(m,i-1)-gzo(m,i+1))/2./dr/rr1(i)
       j=2
       do while (j.le.mm1)
       vr(j,i)=(gzo(j+1,i)-gzo(j-1,i))/2./dz/rr1(i)
       vz(j,i)=(gzo(j,i-1)-gzo(j,i+1))/2./dr/rr1(i)
       j=j+1
       enddo
       i=i+1
       enddo
       j=1
       do while (j.le.m)
       vz(j,1)=vz(j,2)
       vz(j,n)=(gzo(j,nn1)-gzo(j,n))/dr/rr1(n)
       j=j+1
       enddo
c Calculate mixing in the resin bed. This is modeled using an axial dispersion
c model for bubble column mixing.
       j=1
       do while (j.le.m)
       vzm(j)=0.
       k=j-1
       do while (k.ge.1)
       wf=exp(-(dfloat(j-k)+dfloat(j-k-1))*dz/mlen2)
```

```
      if (zden(k).gt.zden(j)) then
      vzm(j)=vzm(j)+dpart**2*vfo**4/((1.-vfo)**2*400.*vis(k,1))
     1*(zden(k)-zden(j))*grav*mlen*wf
      endif
      k=k-1
      enddo
      k=j+1
      do while (k.le.m)
      wf=exp(-(dfloat(k-j)+dfloat(k-j-1))*dz/mlen2)
      if (zden(k).lt.zden(j)) then
      vzm(j)=vzm(j)+dpart**2*vfo**4/((1.-vfo)**2*400.*vis(k,1))
     1*(zden(j)-zden(k))*grav*mlen*wf
      endif
      k=k+1
      enddo
      j=j+1
      enddo
c Calculate total amounts of plutonium and plutonium adsorbed.  Renormalize to
c ensure that the total amount of plutonium is conserved.
      pua=0.
      puf=areat*(lup*sy21+ldown*sy22)
      pusn=areat*(lup*sy21+ldown*sy22)
      i=1
      do while (i.le.n)
      pua=pua+0.5*dz*area(i)*(f(1,i)+f(m,i))
      puf=puf+0.5*dz*area(i)*(fx(1,i)+fx(m,i))
      pusn=pusn+0.5*dz*area(i)*(y2(1,i)+y2(m,i))
      i=i+1
      enddo
      j=2
      do while (j.le.mm1)
      i=1
      do while (i.le.n)
      pua=pua+dz*f(j,i)*area(i)
      puf=puf+dz*fx(j,i)*area(i)
      pusn=pusn+dz*y2(j,i)*area(i)
      i=i+1
      enddo
      j=j+1
      enddo
      fpu=(pusno-pua+puao)/pusn
      sy21=sy21*fpu
      sy22=sy22*fpu
      i=1
      do while (i.le.n)
      j=1
      do while (j.le.m)
      y2old(j,i)=y2old(j,i)*fpu
      y2(j,i)=y2(j,i)*fpu
      j=j+1
      enddo
      i=i+1
      enddo
c Write transient temperatures and velocities to output files.
      time2=time2+dt
      time=time2/60.
      if (abs(nint(time/30.)-time/30.).lt.dt/2./1800.) then
c Write headers for output files.
      write (11,*) 'TIME (MIN)',tab,'POS (CM)',tab,'LOAD',tab,
     1'TCEN (°C)',tab,'TMID (°C)',tab,'TOUT (°C)'
      write (16,*) 'TIME (MIN)',tab,'LENGTH (CM)',tab,'ELU (VF)',tab,
     1'ADS (G/CC)',tab,'LIQ (G/CC)',tab,'TOT (G/CC)'
      write (10,*) 'TIME (MIN)',tab,'LENGTH (CM)',tab,'IND',tab,
```

```
     1'VFA',tab,'VFAM'
      j=1
      write (11,102) time,tab,-1.,tab,0.,tab,stem1c,tab,stem1c,
     1tab,stem1c
      do while (j.le.m)
      poszn=length*(j-1)/(m-1)
      write (11,102) time,tab,poszn,tab,f(j,1),tab,temc(j,1),tab,
     1temc(j,5),tab,temc(j,n)
  102 format (f9.4,a1,f10.5,a1,f7.5,3(a1,f9.4))
      write (16,400) time,tab,poszn,tab,c(j,1),tab,f(j,1),tab,y2(j,1),
     1tab,fx(j,1)
  400 format (f9.4,a1,f10.5,4(a1,f9.5))
      write (10,500) time,tab,poszn,tab,ind3(j,1),tab,
     1y2(j,1),tab,y2max(j,i),tab,cmx2(j,1)
  500 format (f9.4,a1,f10.5,a1,i2,3(a1,f9.5))
      j=j+1
      enddo
      write (11,102) time,tab,-2.,tab,0.,tab,stem2c,tab,stem2c,tab,
     1stem2c
      endif
c Find highest temperature.
      temch=0.
      j=1
      do while (j.le.m)
      i=1
      do while (i.le.n)
      if (temch.lt.temc(j,i)) then
      temch=temc(j,i)
      endif
      i=i+1
      enddo
      j=j+1
      enddo
      if (abs(nint(time/2.)-time/2.).lt.dt/4./60.)
     1write (12,301) time,tab,temch,tab,pua,tab,puf
  301 format (f11.4,a1,f10.5,a1,f11.5,a1,f11.5)
c Increment transient calculation loop.
      s=s+1
      enddo
c Write temperature profile to output file.
      write (17,*) stem1c
      write (17,*) stem2c
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (17,*) temc(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (17,*) temc(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write total plutonium concentrations to output file.
      write (21,*) 1000.*sy21
      write (21,*) 1000.*sy22
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
```

```
      write (21,*) 1000.*fx(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (21,*) 1000.*fx(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write solution plutonium concentrations to output file.
      write (22,*) 1000.*sy21
      write (22,*) 1000.*sy22
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (22,*) 1000.*y2(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (22,*) 1000.*y2(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write densities to output file.
      write (23,*) denst
      write (23,*) densb
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (23,*) dens(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (23,*) dens(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write axial velocities to output file.
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (26,*) 10000.*vz(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (26,*) 10000.*vz(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write radial velocities to output file.
      j=m
      do while (j.ge.1)
      i=n
```

```
      do while (i.ge.1)
      write (25,*) 10000.*vr(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (25,*) 10000.*vr(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
c Write modified stream function profile to output file.  The stream function
c is modified by multiplying by the radius.  This modified function gives a
c profile of the velocity multiplied by the radius.
      j=m
      do while (j.ge.1)
      i=n
      do while (i.ge.1)
      write (27,*) 10000.*gzo(j,i)
      i=i-1
      enddo
      i=2
      do while (i.le.n)
      write (27,*) 10000.*gzo(j,i)
      i=i+1
      enddo
      j=j-1
      enddo
      stop
      end

      subroutine feeder (j,i,m,dy2,dc,df,ffac)
c Compute changes in the plutonium concentrations during feeding or washing.
      implicit double precision (a-z)
      integer i,j,m,ind,ind3
      dimension c(200,100),cold(200,100),f(200,100),fold(200,100),
     1x(200,100),xold(200,100),y2(200,100),y2old(200,100),
     2den1(200,100),ind3(200,100),cmx2(200,100),
     3y2max(200,100)
      common /elut1/c,cold,f,fold,x,xold,y2,y2old,den1,
     2y2max,y2max2,cmax2,ind3,cmx2,cfeed,dt,tau,tau2,ffeed,an1,
     3dpart,dz,xg,vfi2,vfo,ind
      if (xold(j,i).lt.1.) then
      if (xold(j,i).gt.1.d-3)
     1xte=xold(j,i)+dt/tau*xold(j,i)**2/(log((1.+xold(j,i))/
     2(1.-xold(j,i)))-2.*xold(j,i))
      if (xold(j,i).le.1.d-3.and.xold(j,i).gt.1.d-8)
     1xte=xold(j,i)+dt/tau*1.5/xold(j,i)
      if (xte.gt.1.) xte=1.
      if (xte.lt.xold(j,i)) xte=xold(j,i)
      df=ffeed*(xte-xold(j,i))
      endif
      if (xold(j,i).ge.1.) then
      df=0.
      endif
      df=ffac*df
      dy2=dy2-df
      if (y2(j,i)+dy2.lt.0.) then
      dy2=-y2(j,i)
      df=dy2+y2(j,i)
      endif
      dc=dc+4.*dy2*63./238./vfo/den1(j,i)
      if (cold(j,i)+dc.lt.0.) dc=-cold(j,i)
```

```
      return
      end

      subroutine crite (j,i,m)
c Determine if solution is saturated with plutonium.
      implicit double precision (a-z)
      integer i,j,m,ind,ind3
      dimension c(200,100),cold(200,100),f(200,100),fold(200,100),
     1x(200,100),xold(200,100),y2(200,100),y2old(200,100),
     2den1(200,100),ind3(200,100),cmx2(200,100),
     3y2max(200,100)
      common /elut1/c,cold,f,fold,x,xold,y2,y2old,den1,
     2y2max,y2max2,cmax2,ind3,cmx2,cfeed,dt,tau,tau2,ffeed,an1,
     3dpart,dz,xg,vfi2,vfo,ind
      cno3=cold(j,i)*1000./63.*den1(j,i)
      cpu=y2old(j,i)*1000./238./vfo
      cpu2=0.
      if (cno3.le.cmax2) then
      cpu2=y2max2
      else
      aa=1.
      k=1
      do while (k.le.8)
      aa=aa-(2.9*aa**3*(1.-aa)*cno3-2.9**4*0.26*(1.-aa)**4-aa**4)
     1/(3.*2.9*aa**2*(1.-aa)*cno3-2.9*aa**3*cno3
     2+4.*2.9**4*0.26*(1.-aa)**3-4.*aa**3)
      k=k+1
      enddo
      cpu2=cno3/aa-1./2.9/(1.-aa)
      endif
      y2max(j,i)=vfo*0.238*cpu2
      if (cpu.ge.cpu2) then
      ind3(j,i)=1
      else
      ind3(j,i)=0
      endif
      ab=1.
      if (cpu.gt.1.d-8) then
      k=1
      do while (k.le.5)
      ab=(1/3*ab**(4./3.)*cpu**(1./3.)+0.26**(1./3.)*2.9)/
     1(4./3.*ab**(1./3.)*cpu**(1./3.)+0.26**(1./3.)*2.9)
      k=k+1
      enddo
      endif
      if (1.-ab.gt.1.d-8)
     1cno32=ab*cpu+ab/2.9/(1.-ab)
      if (cno32.gt.cmax2.and.cpu.gt.y2max2) cno32=cmax2
      cmx2(j,i)=1.-cno3/cno32
      if (cmx2(j,i).lt.0.) cmx2(j,i)=0.
      if (cmx2(j,i).gt.1.) cmx2(j,i)=1.
      return
      end
```

Sample Input Listing for Thermal Analysis during a Flow Stoppage

```
25.
1
34.1
2.54
2.54
```

```
3.73
0.04
0.0000012
0.0000012
0.33
0.0441
0.13384
0.274
96,48
0
16628
1.0
1.0
36000
36000
```

Sample Output Listing for Thermal Analysis during a Flow Stoppage

TIME (MIN)	TMAX (°C)	PUA (GM)	PUT (GM)
.0000	40.19306	175.87492	207.35316
2.0000	42.22639	175.87492	207.35316
4.0000	44.35815	175.87492	207.35316
6.0000	46.50526	175.87492	207.35316
8.0000	48.64756	175.87492	207.35316
10.0000	50.76676	175.87492	207.35316
12.0000	52.84843	175.87492	207.35316
14.0000	54.89382	175.87492	207.35316
16.0000	56.89002	175.87492	207.35316
18.0000	58.82343	175.87492	207.35316
20.0000	60.69823	175.87492	207.35316
22.0000	62.51902	175.87492	207.35316
24.0000	64.27129	175.87492	207.35316
26.0000	65.96381	175.87492	207.35316
28.0000	67.59841	175.87492	207.35316
30.0000	69.16687	175.87492	207.35316
32.0000	70.68115	175.87492	207.35316
34.0000	72.13645	175.87492	207.35316
36.0000	73.53083	175.87492	207.35316
38.0000	74.87810	175.87492	207.35316
40.0000	76.16772	175.87492	207.35316
42.0000	77.40558	175.87492	207.35316
44.0000	78.59684	175.87492	207.35316
46.0000	79.73650	175.87492	207.35316
48.0000	80.83249	175.87492	207.35316
50.0000	81.88362	175.87492	207.35316
52.0000	82.88905	175.87492	207.35316
54.0000	83.85761	175.87492	207.35316
56.0000	84.78401	175.87492	207.35316
58.0000	85.67161	175.87492	207.35316
60.0000	86.52493	175.87492	207.35316
62.0000	87.34080	175.87492	207.35316
64.0000	88.12386	175.87492	207.35316
66.0000	88.87495	175.87492	207.35316
68.0000	89.59314	175.87492	207.35316
70.0000	90.28342	175.87492	207.35316
72.0000	90.94431	175.87492	207.35316
74.0000	91.57629	175.87492	207.35316
76.0000	92.18444	175.87492	207.35316
78.0000	92.76583	175.87492	207.35316
80.0000	93.32253	175.87492	207.35316
82.0000	93.85738	175.87492	207.35316
84.0000	94.36875	175.87492	207.35316

86.0000	94.85894	175.87492	207.35316
88.0000	95.32928	175.87492	207.35316
90.0000	95.77900	175.87492	207.35316
92.0000	96.21049	175.87492	207.35316
94.0000	96.62406	175.87492	207.35316
96.0000	97.01952	175.87492	207.35316
98.0000	97.39926	175.87492	207.35316
100.0000	97.76288	175.87492	207.35316
102.0000	98.11060	175.87492	207.35316
104.0000	98.44471	175.87492	207.35316
106.0000	98.76440	175.87492	207.35316
108.0000	99.07011	175.87492	207.35316
110.0000	99.36403	175.87492	207.35316
112.0000	99.64506	175.87492	207.35316
114.0000	99.91400	175.87492	207.35316
116.0000	100.17234	175.87492	207.35316
118.0000	100.41935	175.87492	207.35316
120.0000	100.65589	175.87492	207.35316
122.0000	100.88292	175.87492	207.35316
124.0000	101.09996	175.87492	207.35316
126.0000	101.30796	175.87492	207.35316
128.0000	101.50736	175.87492	207.35316
130.0000	101.69789	175.87492	207.35316
132.0000	101.88074	175.87492	207.35316
134.0000	102.05559	175.87492	207.35316
136.0000	102.22282	175.87492	207.35316
138.0000	102.38311	175.87492	207.35316
140.0000	102.53601	175.87492	207.35316
142.0000	102.68269	175.87492	207.35316
144.0000	102.82245	175.87492	207.35316
146.0000	102.95630	175.87492	207.35316
148.0000	103.08400	175.87492	207.35316
150.0000	103.20559	175.87492	207.35316
152.0000	103.32221	175.87492	207.35316
154.0000	103.43312	175.87492	207.35316
156.0000	103.53857	175.87492	207.35316
158.0000	103.63955	175.87492	207.35316
160.0000	103.73576	175.87492	207.35316
162.0000	103.82720	175.87492	207.35316
164.0000	103.91408	175.87492	207.35316
166.0000	103.99669	175.87492	207.35316
168.0000	104.07587	175.87492	207.35316
170.0000	104.15108	175.87492	207.35316
172.0000	104.22249	175.87492	207.35316
174.0000	104.29028	175.87492	207.35316
176.0000	104.35461	175.87492	207.35316
178.0000	104.41565	175.87492	207.35316
180.0000	104.47355	175.87492	207.35316
182.0000	104.52908	175.87492	207.35316
184.0000	104.58178	175.87492	207.35316
186.0000	104.63175	175.87492	207.35316
188.0000	104.67913	175.87492	207.35316
190.0000	104.72403	175.87492	207.35316
192.0000	104.76658	175.87492	207.35316
194.0000	104.80690	175.87492	207.35316
196.0000	104.84508	175.87492	207.35316
198.0000	104.88124	175.87492	207.35316
200.0000	104.91548	175.87492	207.35316
202.0000	104.94789	175.87492	207.35316
204.0000	104.97856	175.87492	207.35316
206.0000	105.00781	175.87492	207.35316
208.0000	105.03569	175.87492	207.35316
210.0000	105.06208	175.87492	207.35316

212.0000	105.08705	175.87492	207.35316
214.0000	105.11066	175.87492	207.35316
216.0000	105.13299	175.87492	207.35316
218.0000	105.15411	175.87492	207.35316
220.0000	105.17407	175.87492	207.35316
222.0000	105.19294	175.87492	207.35316
224.0000	105.21078	175.87492	207.35316
226.0000	105.22763	175.87492	207.35316
228.0000	105.24354	175.87492	207.35316
230.0000	105.25858	175.87492	207.35316
232.0000	105.27278	175.87492	207.35316
234.0000	105.28618	175.87492	207.35316
236.0000	105.29884	175.87492	207.35316
238.0000	105.31079	175.87492	207.35316
240.0000	105.32206	175.87492	207.35316
242.0000	105.33271	175.87492	207.35316
244.0000	105.34275	175.87492	207.35316
246.0000	105.35222	175.87492	207.35316
248.0000	105.36116	175.87492	207.35316
250.0000	105.36958	175.87492	207.35316
252.0000	105.37753	175.87492	207.35316
254.0000	105.38502	175.87492	207.35316
256.0000	105.39209	175.87492	207.35316
258.0000	105.39875	175.87492	207.35316
260.0000	105.40502	175.87492	207.35316
262.0000	105.41094	175.87492	207.35316
264.0000	105.41651	175.87492	207.35316
266.0000	105.42177	175.87492	207.35316
268.0000	105.42671	175.87492	207.35316
270.0000	105.43138	175.87492	207.35316
272.0000	105.43577	175.87492	207.35316
274.0000	105.43998	175.87492	207.35316
276.0000	105.44399	175.87492	207.35316
278.0000	105.44776	175.87492	207.35316
280.0000	105.45132	175.87492	207.35316
282.0000	105.45467	175.87492	207.35316
284.0000	105.45783	175.87492	207.35316
286.0000	105.46080	175.87492	207.35316
288.0000	105.46360	175.87492	207.35316
290.0000	105.46623	175.87492	207.35316
292.0000	105.46871	175.87492	207.35316
294.0000	105.47105	175.87492	207.35316
296.0000	105.47324	175.87492	207.35316
298.0000	105.47531	175.87492	207.35316
300.0000	105.47726	175.87492	207.35316
302.0000	105.47909	175.87492	207.35316
304.0000	105.48081	175.87492	207.35316
306.0000	105.48243	175.87492	207.35316
308.0000	105.48396	175.87492	207.35316
310.0000	105.48539	175.87492	207.35316
312.0000	105.48674	175.87492	207.35316
314.0000	105.48801	175.87492	207.35316
316.0000	105.48921	175.87492	207.35316
318.0000	105.49033	175.87492	207.35316
320.0000	105.49139	175.87492	207.35316
322.0000	105.49238	175.87492	207.35316
324.0000	105.49331	175.87492	207.35316
326.0000	105.49419	175.87492	207.35316
328.0000	105.49502	175.87492	207.35316
330.0000	105.49579	175.87492	207.35316
332.0000	105.49652	175.87492	207.35316
334.0000	105.49721	175.87492	207.35316
336.0000	105.49785	175.87492	207.35316

338.0000	105.49846	175.87492	207.35316
340.0000	105.49903	175.87492	207.35316
342.0000	105.49956	175.87492	207.35316
344.0000	105.50007	175.87492	207.35316
346.0000	105.50054	175.87492	207.35316
348.0000	105.50098	175.87492	207.35316
350.0000	105.50140	175.87492	207.35316
352.0000	105.50179	175.87492	207.35316
354.0000	105.50216	175.87492	207.35316
356.0000	105.50251	175.87492	207.35316
358.0000	105.50283	175.87492	207.35316
360.0000	105.50314	175.87492	207.35316
362.0000	105.50343	175.87492	207.35316
364.0000	105.50370	175.87492	207.35316
366.0000	105.50395	175.87492	207.35316
368.0000	105.50419	175.87492	207.35316
370.0000	105.50441	175.87492	207.35316
372.0000	105.50462	175.87492	207.35316
374.0000	105.50482	175.87492	207.35316
376.0000	105.50500	175.87492	207.35316
378.0000	105.50518	175.87492	207.35316
380.0000	105.50534	175.87492	207.35316
382.0000	105.50550	175.87492	207.35316
384.0000	105.50564	175.87492	207.35316
386.0000	105.50578	175.87492	207.35316
388.0000	105.50590	175.87492	207.35316
390.0000	105.50602	175.87492	207.35316
392.0000	105.50613	175.87492	207.35316
394.0000	105.50624	175.87492	207.35316
396.0000	105.50634	175.87492	207.35316
398.0000	105.50643	175.87492	207.35316
400.0000	105.50652	175.87492	207.35316
402.0000	105.50660	175.87492	207.35316
404.0000	105.50668	175.87492	207.35316
406.0000	105.50675	175.87492	207.35316
408.0000	105.50682	175.87492	207.35316
410.0000	105.50688	175.87492	207.35316
412.0000	105.50694	175.87492	207.35316
414.0000	105.50700	175.87492	207.35316
416.0000	105.50705	175.87492	207.35316
418.0000	105.50710	175.87492	207.35316
420.0000	105.50715	175.87492	207.35316
422.0000	105.50719	175.87492	207.35316
424.0000	105.50723	175.87492	207.35316
426.0000	105.50727	175.87492	207.35316
428.0000	105.50731	175.87492	207.35316
430.0000	105.50734	175.87492	207.35316
432.0000	105.50738	175.87492	207.35316
434.0000	105.50741	175.87492	207.35316
436.0000	105.50743	175.87492	207.35316
438.0000	105.50746	175.87492	207.35316
440.0000	105.50749	175.87492	207.35316
442.0000	105.50751	175.87492	207.35316
444.0000	105.50753	175.87492	207.35316
446.0000	105.50755	175.87492	207.35316
448.0000	105.50757	175.87492	207.35316
450.0000	105.50759	175.87492	207.35316
452.0000	105.50761	175.87492	207.35316
454.0000	105.50762	175.87492	207.35316
456.0000	105.50764	175.87492	207.35316
458.0000	105.50765	175.87492	207.35316
460.0000	105.50767	175.87492	207.35316
462.0000	105.50768	175.87492	207.35316

464.0000	105.50769	175.87492	207.35316
466.0000	105.50770	175.87492	207.35316
468.0000	105.50771	175.87492	207.35316
470.0000	105.50772	175.87492	207.35316
472.0000	105.50773	175.87492	207.35316
474.0000	105.50774	175.87492	207.35316
476.0000	105.50775	175.87492	207.35316
478.0000	105.50776	175.87492	207.35316
480.0000	105.50777	175.87492	207.35316
482.0000	105.50777	175.87492	207.35316
484.0000	105.50778	175.87492	207.35316
486.0000	105.50779	175.87492	207.35316
488.0000	105.50779	175.87492	207.35316
490.0000	105.50780	175.87492	207.35316
492.0000	105.50780	175.87492	207.35316
494.0000	105.50781	175.87492	207.35316
496.0000	105.50781	175.87492	207.35316
498.0000	105.50782	175.87492	207.35316
500.0000	105.50782	175.87492	207.35316
502.0000	105.50783	175.87492	207.35316
504.0000	105.50783	175.87492	207.35316
506.0000	105.50783	175.87492	207.35316
508.0000	105.50784	175.87492	207.35316
510.0000	105.50784	175.87492	207.35316
512.0000	105.50784	175.87492	207.35316
514.0000	105.50785	175.87492	207.35316
516.0000	105.50785	175.87492	207.35316
518.0000	105.50785	175.87492	207.35316
520.0000	105.50785	175.87492	207.35316
522.0000	105.50786	175.87492	207.35316
524.0000	105.50786	175.87492	207.35316
526.0000	105.50786	175.87492	207.35316
528.0000	105.50786	175.87492	207.35316
530.0000	105.50786	175.87492	207.35316
532.0000	105.50787	175.87492	207.35316
534.0000	105.50787	175.87492	207.35316
536.0000	105.50787	175.87492	207.35316
538.0000	105.50787	175.87492	207.35316
540.0000	105.50787	175.87492	207.35316
542.0000	105.50787	175.87492	207.35316
544.0000	105.50787	175.87492	207.35316
546.0000	105.50788	175.87492	207.35316
548.0000	105.50788	175.87492	207.35316
550.0000	105.50788	175.87492	207.35316
552.0000	105.50788	175.87492	207.35316
554.0000	105.50788	175.87492	207.35316
556.0000	105.50788	175.87492	207.35316
558.0000	105.50788	175.87492	207.35316
560.0000	105.50788	175.87492	207.35316
562.0000	105.50788	175.87492	207.35316
564.0000	105.50789	175.87492	207.35316
566.0000	105.50789	175.87492	207.35316
568.0000	105.50789	175.87492	207.35316
570.0000	105.50789	175.87492	207.35316
572.0000	105.50789	175.87492	207.35316
574.0000	105.50789	175.87492	207.35316
576.0000	105.50789	175.87492	207.35316
578.0000	105.50789	175.87492	207.35316
580.0000	105.50789	175.87492	207.35316
582.0000	105.50789	175.87492	207.35316
584.0000	105.50789	175.87492	207.35316
586.0000	105.50789	175.87492	207.35316
588.0000	105.50789	175.87492	207.35316

590.0000	105.50789	175.87492	207.35316
592.0000	105.50790	175.87492	207.35316
594.0000	105.50790	175.87492	207.35316
596.0000	105.50790	175.87492	207.35316
598.0000	105.50790	175.87492	207.35316
600.0000	105.50790	175.87492	207.35316