ornl

ORNL/TM-13035

**OAK RIDGE
NATIONAL
LABORATORY**

LOCKHEED MARTIN

User's Manual for the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating Low-Level Radioactive Waste Disposal Facility Source Terms (Version 2.0)

Alan S. Icenhour
M. Lynn Tharp

MANAGED AND OPERATED BY
LOCKHEED MARTIN ENERGY RESEARCH CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

ORNL-27 (3-96)

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O.Box 62, Oak Ridge, TN 37831; prices available from (423) 576-8401, FTS 626-8401.

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

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

ORNL/TM-13035

User's Manual for the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating Low-Level Radioactive Waste Disposal Facility Source Terms (Version 2.0)

Alan S. Icenhour
M. Lynn Tharp

Date Published: August 1996

MASTER

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831-6495
managed by
LOCKHEED MARTIN ENERGY RESEARCH CORP.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-96OR22464

CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

Table **Page**

LIST OF EXHIBITS

LIST OF ABBREVIATIONS, ACRONYMS, INITIALISMS, AND CHEMICAL NOTATIONS

ANSI	American National Standards Institute, Inc.
CIIDF	Class II Disposal Facility
$Ca(OH)_2$	Calcium hydroxide
C–S–H	Calcium-silicate-hydrate
CO_2	Carbon dioxide
FLOTHRU	Subroutine in SOURCE1 and SOURCE2 that calculates radionuclide releases as a result of diffusion
KOH	Potassium hydroxide
LLW	Low-level radioactive waste
NaOH	Sodium hydroxide
ORNL	Oak Ridge National Laboratory
ORR	Oak Ridge Reservation
SOURCE	Used when referring to both the SOURCE1 and SOURCE2 computer codes
SWSA 6	Solid Waste Storage Area 6

LIST OF SYMBOLS

English

A = surface area over which oxygen diffuses to the reinforcement (cm^2)

$\left(\frac{A}{b}\right)$ = cross-sectional area of steel reinforcement per unit width of slab (m)

a = width of waste cell (in.)

a_u = unit width of concrete member (in.)

b = length of waste cell (in.)

Ca_c = $Ca(OH)_2$ concentration in concrete (mol/L)

Ca_p = $Ca(OH)_2$ concentration in concrete pore solution (mol/L)

Ca_l = fractional groundwater release rate of $Ca(OH)_2$ ($year^{-1}$)

C_c = concrete cover thickness (m)

C_d = depth of the compression block (in.)

C_e = concentration of sulfate as ettringite at the time at which spalling occurs (mol/m^3)

C_f = CO_2 concentration ahead of carbonation front (mol/L)

CL_{gw} = chloride ion concentration in groundwater (mol/L)

CL_i = initial chloride ion concentration in concrete (mol/L)

CL_s = chloride ion concentration at steel reinforcement (mol/L)

c_o = groundwater sulfate concentration (mol/m^3)

C_s = CO_2 concentration at surface of concrete (mol/L)

C_{str} = compressive strength of concrete ($lb/in.^2$)

C_t = concrete-member thickness (m)

C_{tc} = thickness of corrosion layer under conditions of free expansion (in.)

C_x = concentration of CO_2 bound in concrete (mol/L)

C_1 = concentration of contaminant in the inner layer (g/cm^3)

C_2 = concentration of contaminant in the outer layer (g/cm^3)

d = effective depth of steel (distance from the top of the slab to the center of the steel reinforcement) (m)

d_c = concrete cover thickness on tension face (in.)

D_{Cl} = effective diffusivity of chloride in concrete (m^2/s)

D_{CO2} = diffusion coefficient of CO_2 in concrete (m^2/s)

d_{cr} = crack depth (in.)

d_{cv} = distance from concrete face to center of steel reinforcement (in.)

D_{fx} = flexural rigidity of floor in the x-direction (lb-in.^{2})

D_{fy} = flexural rigidity of floor in the y-direction (lb-in.^{2})

D_i = "intrinsic" diffusion coefficient of sulfate ions in water-saturated cement (m^2/s)

$\frac{d[O_2]}{dx}$ = dissolved oxygen concentration gradient (g/cm^4)

D_o = effective diffusivity of oxygen through concrete (cm^2/s)

D_r = flexural rigidity of the roof (lb-in.^{2})

d_t = distance from steel reinforcement in tension to compression face of concrete (in.)

D_w = flexural rigidity of wall (lb-in.^{2})

D_y = effective diffusion coefficient of $Ca(OH)_2$ in concrete (m^2/s)

D_1 = effective diffusion coefficient for the contaminant in layer 1 (cm^2/s)

D_2 = effective diffusion coefficient for the contaminant in layer 2 (cm^2/s)

E = Young's modulus (Pa)

E_c = modulus of elasticity of concrete (lb/in.^{2})

E_r = modulus of elasticity of corrosion product (lb/in.^{2})

E_s = modulus of elasticity of steel (MPa or lb/in.^{2}, as appropriate)

F_c = concentrated force (lb/in.)

$f_c^/$ = specified compressive strength of concrete (MPa)

f_r = modulus of rupture (lb/in.^{2})

f_s = friction angle of soil backfill (deg)

f_t = fraction of $Ca(OH)_2$ remaining in concrete member as a function of position and time (unitless)

F_w = compressive force on wall at height z (lb/in.)

f_w = friction angle of waste (deg)

f_{ws} = yield strength of cast iron (lb/in.^{2})

f_y = specified yield strength of steel reinforcement (MPa or lb/in.^{2}, as appropriate)

h_c = height of vault wall (in.)

H_f = fraction of hydrated CaO (unitless)

h_f = floor thickness (in.)

h_m = concrete member thickness (in.)

h_r = roof thickness (in.)

h_s = silo or well height (in.)

h_w = waste thickness (in.)

h_{wl} = thickness of wall (in.)

I = water percolation rate through vault (m/year)

I_c = cracking moment of inertia in the x- or y-direction (in.^{4})

I_e = effective moment of inertia per unit width of concrete member (in.^{3})

I_g = moment of inertia of concrete section (in.^{4})

I_m = total water percolation rate (cm/month)

I_r = vertical water percolation rate (cm/month)

I_{lpx} = trigonometric function (unitless)

I_{lpy} = trigonometric function (unitless)

I_{2px} = trigonometric function (unitless)

I_{2py} = trigonometric function (unitless)

J_o = oxygen flux at the steel reinforcement (g/s)

k = carbonation coefficient ($m/s^{0.5}$)

$\tilde{k}$ = modulus of the subgrade reaction ($lb/in.^3$)

K_d = distribution coefficient (mL/g)

L = mass of radionuclide leached because of advection (g)

L_{cm} = thickness of corrugated steel liner on compression face (in.)

l_f = length of floor (in.)

L_{tn} = thickness of corrugted steel liner on tension face (in.)

M = bending moment due to uniform loading in x or y direction (lb–in./in.)

M_{cr} = cracking moment per unit width $\left(\frac{\text{lb–in.}}{\text{in.}}\right)$

M_{my} = modified moment (lb-in.)

M_r = radial component of bending moment (lb-in./in.)

M_t = tangential component of bending moment (lb-in./in.)

M_u = ultimate flexural strength (lb-in./in.)

M_{uc} = ultimate strength of the wall in compression (lb/in.)

M_x = bending moment resulting from uniform loading in the x-direction parallel to width of floor or roof (lb-in./in.)

M_{xh} = bending moment resulting from hydrostatic pressures in the x-direction parallel to the width of the wall (lb-in./in.)

M_y = bending moment resulting from uniform loading in the y-direction parallel to length of floor or roof (lb-in./in.)

M_{yh} = bending moment resulting from hydrostatic pressures in the y-direction parallel to the length of the wall (lb-in./in.)

N_{ac} = ultimate strength or critical buckling strength under axial compression (lb/in.)

N_{rc} = ultimate strength or critical buckling strength under ring compression (lb/in.)

N_{θ} = ring compression force (lb/in.^{2})

P = maximum hydrostatic pressure (lb/in.^{2})

P_i = internal pressure due to corrosion (lb/in.^{2})

P_x = applied concentrated load caused by wall in x-direction (lb/in.)

P_y = applied concentrated load caused by wall in y-direction (lb/in.)

Q = shear force on roof of well or silo (lb/in.)

q = water infiltration rate (cm/s)

Q_{rmax} = maximum shear force on floor (lb/in.)

Q_o = initial mass of radionuclide in the waste (g)

Q_r = radionuclide release entering recharge component (g/month)

q_r = uniform load on vault, silo, or well roof (lb/in.^{2})

Q_t = total radionuclide release from disposal facility (g/month)

q_w = uniform load on vault, silo, or well wall (lb/in.^{2})

Q_x = shear force resulting from uniform loading in x-direction (lb/in.)

Q_{xh} = shear force resulting from hydrostatic loading in the x-direction (lb/in.)

Q_y = shear force resulting from uniform loading in y-direction (lb/in.)

Q_{yh} = shear force resulting from hydrostatic loading in the y-direction (lb/in.)

R = degradation rate (m/s)

r = distance from center of silo or well roof (in.)

R_d = retardation factor (dimensionless)

r_c = radius of remaining steel reinforcement (in.)

R_f = retardation factor for $Ca(OH)_2$ in concrete (unitless)

r_o = original radius of steel reinforcement (in.)

R_r = roof reaction (lb/in.)

R_{ry} = roof reaction in y direction at height z (lb/in.)

R_{rx} = roof reaction in x direction (lb/in.)

r_s = radius of silo or well (in.)

s = soil cover thickness (in.)

S_{d1} = diameter of steel reinforcement in direction 1, and closest to concrete outer tension face (in.)

S_m = mean crack spacing (in.)

S_{m1} = mean crack spacing in direction 1 (in.)

S_{p2} = spacing of steel reinforcement in direction 2, perpendicular to direction 1 (in.)

St_m = maximum concrete compressive stress (lb/in.^{2})

St_n = tensile stress in steel reinforcement (lb/in.^{2})

S_{tn} = area of steel reinforcement in tension per unit width (in.^{2}/in.)

t = time (s)

t_{spall} = time at which spalling occurs (s)

t_1, t_2 = the bounds of the time period of interest (s)

V_{cr} = shear force at which cracking occurs (lb/in.)

V_f = shear force at which well wall fails (lb/in.)

W = waste thickness (cm)

$\hat{W}$ = width of concrete member (m)

WCR = water-cement ratio (unitless)

w_f = width of floor (in.)

W_m = mean crack width (in.)

w_r = width of roof in (x, y) direction (in.)

$W(x,y)$ = deflection of roof at location (x,y) (in.)

X = depth of carbonation (m)

x = spatial position (cm)

X_{spall} = reaction zone thickness at which spalling occurs (m)

y = distance from centerline (m)

y_t = distance from the centroidal axis to the tensile face of the concrete (in.)

z = wall height (in.)

Greek

α = roughness factor for fracture path (unitless)

β = linear strain caused by a mole of sulfate reacted in 1 m^3 (m^3/mol)

β_1 = a factor used in the equivalent rectangular stress diagram for concrete at the ultimate load (dimensionless)

Δ = thickness of the free expansion layer (in.)

ϵ_c = concrete porosity (dimensionless)

ϵ_c' = ultimate concrete strain (dimensionless)

ϵ_{sh} = shrinkage strain of concrete (in./in.)

ϵ_y = yield strain of steel (dimensionless)

γ = fracture surface energy of concrete (J/m^2)

κ = $\sqrt{D_2/D_1}$ (dimensionless)

λ_d = radioactive decay constant (s^{-1})

λ_L = leach rate constant (s^{-1}),

λ_x = $\left[\frac{\tilde{k}l_f}{4D_{fx}}\right]^{0.25}$ $\left(in.^{-1}\right)$

λ_y = $\left[\frac{\tilde{k}w_f}{4D_{fy}}\right]^{0.25}$ (in.^{-1})

μ_c = Poisson's ratio for concrete (unitless)

μ_r = Poisson's ratio of corrosion product (unitless)

ϕ = strength reduction factor (unitless)

ρ = reinforcement ratio (dimensionless)

ρ_b = bulk density of waste (g/cm^3)

ρ_c = density of reinforced concrete (lb/in.^{3})

ρ_{lim} = limiting reinforcement ratio (dimensionless)

ρ_s = density of soil cover (lb/in.^{3})

ρ_w = density of waste (lb/in.^{3})

σ_x = stress at surface of concrete (lb/in.^{2})

$\sigma_{\theta E}$ = maximum tangent stress (lb/in.^{2})

$\sigma_{\theta Q}$ = tangent stress at point Q (lb/in.^{2})

θ = relative saturation (i.e., volume of water in waste/volume of waste) (dimensionless)

ACKNOWLEDGMENTS

Version 1.0 of the SOURCE1 and SOURCE2 computer codes was developed by Rob Shuman, N. Chau, and E. A. Jennrich of Rogers & Associates Engineering Corporation. The authors are especially grateful to Rob Shuman for his many helpful comments and his advice during the development of Version 2.0. At the Oak Ridge National Laboratory, Steve Storch and Royes Salmon provided excellent critical review and comment. Martin Tull, Don Lee, Herschel Godbee, and Jerry Klein were outstanding sources of support for this work. Ralph Sharpe and Catherine Shappert both provided insightful and timely editorial review. Finally, Sue McDaniel's expert word processing skills and advice resulted in the preparation of this report.

ABSTRACT

The SOURCE1 and SOURCE2 computer codes (collectively called the SOURCE computer codes) calculate source terms (i.e., radionuclide release rates) for performance assessments of low-level radioactive waste (LLW) disposal facilities. SOURCE1 is used to simulate radionuclide releases from tumulus-type facilities. SOURCE2 is used to simulate releases from silo-, well-, well-in-silo-, and trench-type disposal facilities. The SOURCE codes (a) simulate the degradation of engineered barriers (e.g., concrete and metal containers) and (b) provide an estimate of the source term for LLW disposal facilities.

The SOURCE computer codes were originally developed by Rogers & Associates Engineering Corporation for the Oak Ridge National Laboratory (ORNL). These codes have been used in the radiological performance assessments of the Solid Waste Storage Area 6 (SWSA 6) and Class II LLW disposal sites. Both sites are located on the Oak Ridge Reservation. Numerous disposal technologies have been used at SWSA 6, including tumulus, silos, wells, wells-in-silos, and trenches. The Class II disposal facility is designed to use tumulus-type disposal technology.

This manual summarizes the major changes that have been effected since the codes were originally developed. These revisions include incorporation of a new advective transport model into SOURCE1 and SOURCE2, development of a new model for SOURCE1 that calculates the degradation and failure of a tumulus-type concrete pad and leachate collection system, improvement of routines for controlling water infiltration and radionuclide inventory inputs, and expansion of options for obtaining output summaries. An overview of both SOURCE1 and SOURCE2 is presented. This overview includes objectives, conceptual model summary, code structure, and computing system requirements. A detailed description of the mathematical models used to implement the conceptual model is provided. Input data requirements and output options are also summarized. A description of the FLOTHRU computer program, a subroutine in both SOURCE1 and SOURCE2 that calculates radionuclide releases as a result of diffusion, is presented in Appendix A. Also included in the appendixes are a glossary of code variables, sample input data files and corresponding output files, and listings for both computer codes.

1. INTRODUCTION

1.1 BACKGROUND

The SOURCE1 and SOURCE2 computer codes (collectively called the SOURCE computer codes) were originally developed by Rogers & Associates Engineering Corporation for the Oak Ridge National Laboratory (ORNL).[1] These codes have been used in the radiological performance assessments of the Solid Waste Storage Area 6 (SWSA 6)[2] and Class II Low-Level Radioactive Waste (LLW) disposal sites. Both disposal sites are located on the Oak Ridge Reservation (ORR). Numerous disposal technologies have been used at SWSA 6, including tumulus, silos, wells-in-silos, wells, and trenches. The Class II disposal facility (CIIDF) is designed to use tumulus-type disposal technology.

The SOURCE computer codes were developed to use in the performance assessments of the various types of disposal technologies used at ORNL. SOURCE1 is applicable to tumulus-type facilities, while SOURCE2 can be applied to silo, well-in-silo, well, and trench-type facilities. The SOURCE codes (a) simulate the degradation of engineered barriers (e.g., concrete and metal containers) and (b) provide an estimate of the source term (i.e., radionuclide release rate) for LLW disposal facilities.

The original version (Version 1.0) of the SOURCE codes was modified by ORNL. These modifications were made to improve conceptual models, to increase flexibility of the computer codes, and to correct discrepancies identified during the SOURCE code verification process. The modifications to the SOURCE codes resulted in Version 2.0, which is documented in this user's manual. The following paragraphs provide a brief presentation of major changes that have been made to the SOURCE codes. Section 2 provides an overview of the SOURCE1 and SOURCE2 computer codes, including objectives, conceptual model summary, code structure, and computing system requirements. The conceptual and mathematical models used in the SOURCE codes are discussed in Sect. 3. Detailed information required to construct input data sets and a description of output files are given in Sect. 4. Appendix A provides a description of the algorithm for FLOTHRU, a subroutine in both SOURCE1 and SOURCE2, which calculates the release of radionuclides as a result of diffusion. Appendix B contains a glossary of the variables used in the codes. Sample input data files and corresponding output files are provided in Appendix C. Appendix D contains listings of the SOURCE1 and SOURCE2 computer codes.

1.2 SUMMARY OF MAJOR REVISIONS TO THE SOURCE1 AND SOURCE2 COMPUTER CODES

Several revisions have been incorporated into Version 2.0 of the SOURCE codes. Major revisions are summarized in this section to highlight the differences between Versions 1.0 and 2.0. Numerous minor revisions to the codes and code documentation have also been made. These minor revisions have been incorporated throughout this document. Major revisions include incorporation of a new advective transport model into SOURCE1 and SOURCE2, development of a new model for SOURCE1 that calculates the degradation and failure of a tumulus-type concrete pad and leachate collection system, improvement of routines for controlling water infiltration and radionuclide inventory inputs, and expansion of options for obtaining output summaries.

1.2.1 New Advective Transport Model

A new advective transport model was incorporated into the SOURCE codes to improve the simulation of the time dependence of the radionuclide inventory in the disposal facility. This analytical model was developed based on work presented in ref. 3. A detailed derivation of the model can be found in ref. 4.

The total radionuclide release during a time-step is calculated by the following formula:

$$L = \frac{\lambda_L}{\lambda_L + \lambda_d} Q_o \left[e^{-(\lambda_L + \lambda_d)t_1} - e^{-(\lambda_L + \lambda_d)t_2} \right] , \tag{1.1}$$

where

L = mass of radionuclide leached because of advection (g),

λ_L = leach rate constant (s^{-1}),

λ_d = radioactive decay constant (s^{-1}),

Q_o = initial mass of radionuclide in the waste (g), and

t_1, t_2 = the bounds of the time period of interest (s).

The leach rate constant, λ_L, is given by

$$\lambda_L = \frac{q}{W \theta R_d} , \tag{1.2}$$

where

q = water infiltration rate (cm/s),

W = waste thickness (cm),

θ = relative saturation (i.e., volume of water in waste/volume of waste) (dimensionless), and

R_d = retardation factor (dimensionless).

Finally, the retardation factor, R_d, can be calculated by the following equation:

$$R_d = 1 + \frac{\rho_b}{\theta} K_d \quad , \tag{1.3}$$

where

ρ_b = bulk density of waste (g/cm^3) and

K_d = distribution coefficient (mL/g).

In ref. 4, comparisons were made between the new advective transport model and the original model in the SOURCE codes. To perform these comparisons, a number of simulations were conducted using the SOURCE1 and SOURCE2 codes. These simulations allowed for examination of various radionuclides, half-lives, distribution coefficients, radionuclide inventories, and types of disposal. In general, the two advective models produced similar results although the original model predicted a slightly higher cumulative radionuclide release than the new model. A detailed description of the advective model comparisons can be found in ref. 4.

1.2.2 Degradation Models for Concrete Pad and Leachate Collection System

The tumulus-type disposal facility in use at ORNL has a steel-reinforced pad on which disposal vaults are placed and a leachate collection system, which collects water that infiltrates through the waste and reaches the concrete pad. Hence, as long as the pad and collection system are intact and perform correctly, any radionuclide releases from the waste should be captured and not released to the environment. Routines that simulate the degradation and failure of the concrete pad and the leachate collection system have been developed and incorporated into the SOURCE1 code.

1.2.2.1 Concrete Pad Degradation Model

The SOURCE1 code predicts the performance of concrete vaults in a tumulus-type disposal facility. However, the original version of SOURCE1 did not account for the presence of a reinforced concrete pad under the vaults. This pad, while intact, should divert water to the leachate collection system. To incorporate the performance of the concrete pad into SOURCE1, a compressive failure model was assumed. Failure was estimated by calculating the reinforcement ratio.[5] The reinforcement ratio is defined by

$$\rho = \left(\frac{A}{b}\right)\frac{1}{d} \quad , \tag{1.4}$$

where

ρ = reinforcement ratio (dimensionless),

$\left(\frac{A}{b}\right)$ = cross-sectional area of steel reinforcement per unit width of slab (m), and

d = effective depth of steel (distance from the top of the slab to the center of the steel reinforcement) (m).

The reinforcement ratio at which compressive failure may occur is called the *limiting reinforcement ratio* and is given by[5]

$$\rho_{\text{lim}} = \frac{\epsilon_c'}{\epsilon_c' + \epsilon_y} 0.85\beta_1 \frac{f_c'}{f_y} \quad , \tag{1.5}$$

where

ρ_{lim} = limiting reinforcement ratio (dimensionless),

ϵ_c' = ultimate concrete strain (for this application, taken as 0.003) (dimensionless),

ϵ_y = yield strain of steel (dimensionless),

β_1 = a factor used in the equivalent rectangular stress diagram for concrete at the ultimate load (dimensionless),

f_c' = specified compressive strength of concrete (MPa), and

f_y = specified yield strength of steel reinforcement (MPa).

The yield strain of the steel reinforcement can be calculated by

$$\epsilon_y = \frac{f_y}{E_s} \quad , \tag{1.6}$$

where

E_s = modulus of elasticity of steel reinforcement (for this application, taken as 200,000 MPa) (MPa).

The value of β_1 is determined as follows:[5]

$$\beta_1 = 0.85 \text{ for } f_c' \leq 30 \text{ MPa or}$$

$$\beta_1 = 0.85 - 0.08\left(\frac{f_c' - 30}{10}\right) \text{ for } f_c' > 30 \text{ MPa} .$$

The values of the reinforcement ratio and the limiting reinforcement ratio are evaluated at annual time steps in SOURCE1. These two values are compared; when the reinforcement ratio exceeds the limiting value, the pad is said to have failed hydraulically. Failure of the pad will allow leachate to be released to the environment. Values of both ρ and ρ_{lim} will change because of the degradation of the concrete. The concrete is simulated to degrade by using the sulfate attack and calcium hydroxide leaching subroutines in SOURCE1. Corrosion of reinforcing steel was not considered because the rates of sulfate attack and calcium hydroxide leaching were judged to greatly exceed the rate of degradation resulting from corrosion. Sulfate attack results in the spalling off of the concrete cover on the reinforcing steel. Hence, as the effective depth of the steel decreases, the reinforcement ratio increases. Leaching of calcium hydroxide from the concrete pad results in reduced concrete strength. Therefore, as the compressive strength of the concrete decreases, the limiting reinforcement ratio decreases. Both of the concrete degradation mechanisms result in a decrease of the margin between the reinforcement ratio and the limiting reinforcement ratio, ultimately resulting in pad failure.

1.2.2.2 Leachate Collection System Degradation Model

Water that reaches an intact concrete pad of a tumulus-type facility will be diverted to a leachate collection system. This system consists of piping, valves, collection sumps, and monitoring equipment. Ideally, with a properly functioning system, all leachate will be collected, and no release of radionuclides to the environment will occur.

As with the concrete pad, the original version of the SOURCE1 code did not simulate the performance and degradation of the leachate collection system. A model has subsequently been developed that describes the functionality fraction of the collection system as a function of time.

The functionality fraction is defined as the ratio of the amount of radionuclide in the collected leachate to the total radionuclide release from the disposal vaults and can vary from 0 to 1. With a value of 1, the leachate collection system is fully functional, and no radionuclides are released to the environment. A zero value indicates a completely degraded system which allows all leached radionuclides to be released to the environment.

The initial functionality fraction and the length of the institutional control period are input parameters to the SOURCE1 code. The functionality fraction degrades linearly to zero from the beginning of the simulation until the end of the institutional control period. The degradation of the collection system is assumed to result from piping and valve leaks or failures, flow obstructions within the system, leakage or overflow of collection sumps, degraded monitoring equipment, etc. At the end of the institutional control period, no maintenance of the collection system is assumed to occur. Hence, no credit is taken for the collection system after the end of institutional control. Additionally, if the concrete pad is predicted to fail hydraulically before the end of institutional control, the functionality fraction is set to zero at the time of pad failure.

1.2.3 Variation of Water Infiltration Input

In the original version of the SOURCE codes, only one set of water infiltration values could be input. This set consisted of 12 values of water infiltration data (1 value for each month in the year) which were used for each year of the simulation. Because simulations are typically performed for periods of 1000 years or greater, water infiltration would certainly vary with time. The SOURCE codes were modified to allow for variation of water infiltration data. The one set of infiltration values in the input data file was replaced with the name of a file which contains multiple sets of infiltration data. Each set corresponds to a defined time period during the disposal facility performance simulation. For example, six such periods have been defined by ORNL for tumulus-type disposal facilities: (1) the active-use period, during which vaults are placed on the tumulus pad; (2) the capping period, during which the facility is covered with an engineered cap; (3) the cap-decline period, during which the cap weathers and degrades; (4) the grass-cover period, during which the facility is covered with grass and vegetation; (5) the forest-succession period, during which small trees and bushes begin to grow on top of the facility; and (6) the forest-cover period, during which the disposal facility becomes completely covered by trees. Representative water infiltration values can be developed for each of these periods, and with the modifications to the SOURCE codes, these values can be applied during the appropriate time period.

1.2.4. Variation of Radionuclide Inventory Input

In the original version of the SOURCE codes, only one value of radionuclide inventory (for each radionuclide being simulated) could be input. This input represented a disposal at the beginning of the simulation (i.e., "time zero") with no further disposals. However, at many sites, waste disposal may have occurred over a number of years. For example, disposal operations at the SWSA 6 site occurred over a period of more than 20 years. The input to the SOURCE codes was changed to allow simulation of variable disposal of a radionuclide over a period of years. The code user defines the time periods of disposal and provides the amount disposed during each time period. Additionally, some disposal sites may have multiple disposal facilities that began disposal operations at different times. To address this situation, the user supplies a reference year for beginning the simulation, which will ensure that (a) all simulations start at the same point in time and (b) the time dependence of the waste disposal is properly represented.

1.2.5 Addition of Output Files

Version 1.0 of the SOURCE codes contained three output files. One file provided a summary of input data and of engineered barrier degradation. Another file provided, as a function of time, calculated radionuclide releases that recharge to groundwater. The third file provided, also as a function of time, calculated radionuclide releases that flow laterally in the shallow storm-flow region. To provide more information from each simulation, five new output files were created for SOURCE1, and three new output files were created for SOURCE2. Summaries of the input and output file structures for SOURCE1 and SOURCE2 are presented in Sect. 4.

The output files now available for the SOURCE codes provide a wide variety of data from a source term simulation. Additionally, the output files have been structured to allow for use of the output data by both spreadsheet and graphing software. These types of software applications aid in quality assurance checks and interpretation of simulation results.

2. COMPUTER CODE OVERVIEW

The modeling methodology used in simulating the long-term performance of LLW disposal facilities at the SWSA 6 and CIIDF sites has been incorporated into two separate computer codes. The SOURCE1 code models the performance of tumulus-type technology used at Tumulus I and II, the Interim Waste Management Facility (IWMF), and the CIIDF (Fig. 2.1). The SOURCE2 computer code models the performance of disposal silos (Fig. 2.2), wells-in-silos, wells, and trenches. The code objectives, a brief conceptual model summary, and computing system requirements are presented in the following subsections.

2.1 CODE OBJECTIVES

The SOURCE computer codes are used in the evaluation of source terms for LLW disposal facilities. Four major objectives for the SOURCE codes are to:

- Provide for the simulation of the long-term performance and degradation of engineered barriers used in LLW disposal facilities.
- Provide for the simulation of radionuclide releases from LLW disposal facilities. These simulations should include the mechanisms of advection and diffusion and should account for radioactive decay and sorption of radionuclides.
- Provide for the coupling of calculations of engineered barrier degradation with calculations of radionuclide releases.
- Provide sufficient output data to evaluate simulation results and for use in subsequent performance assessment calculations.

2.2 CONCEPTUAL MODEL SUMMARY

The routines of the SOURCE codes have four primary functions: structural analysis, simulation of concrete and metal barrier degradation, cracking analyses, and nuclide-leaching calculations. The structural analysis routine establishes initial bending moments and shear forces. The concrete and metal barrier degradation routines simulate the deterioration of engineered barriers with time. The cracking analyses routines calculate moments and shears required for concrete cracking and compare these values with the moments and shears evaluated in the structural analysis.

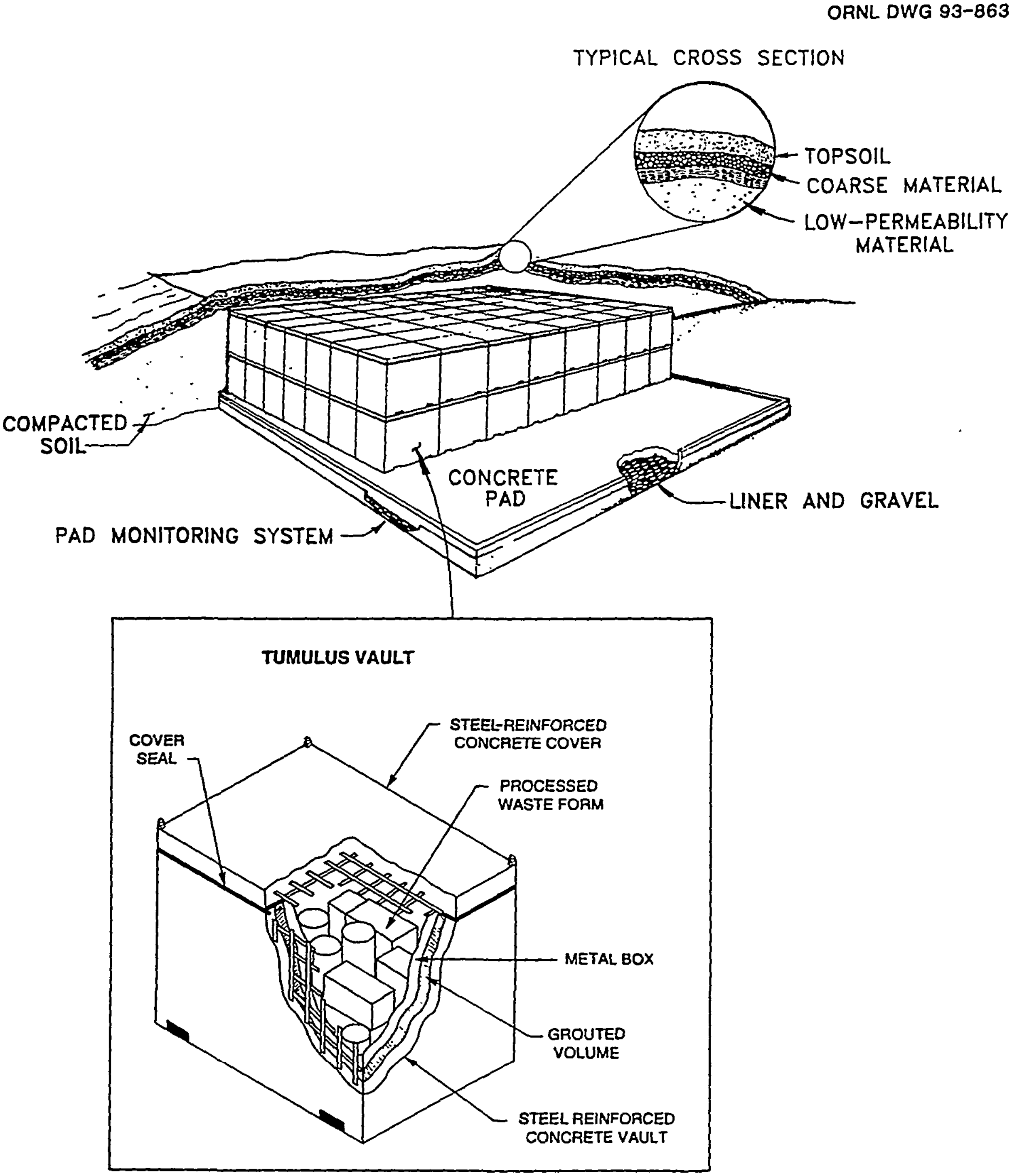

Fig. 2.1. Representative tumulus-type disposal facility modeled by SOURCE1.

ORNL DWG 93-864

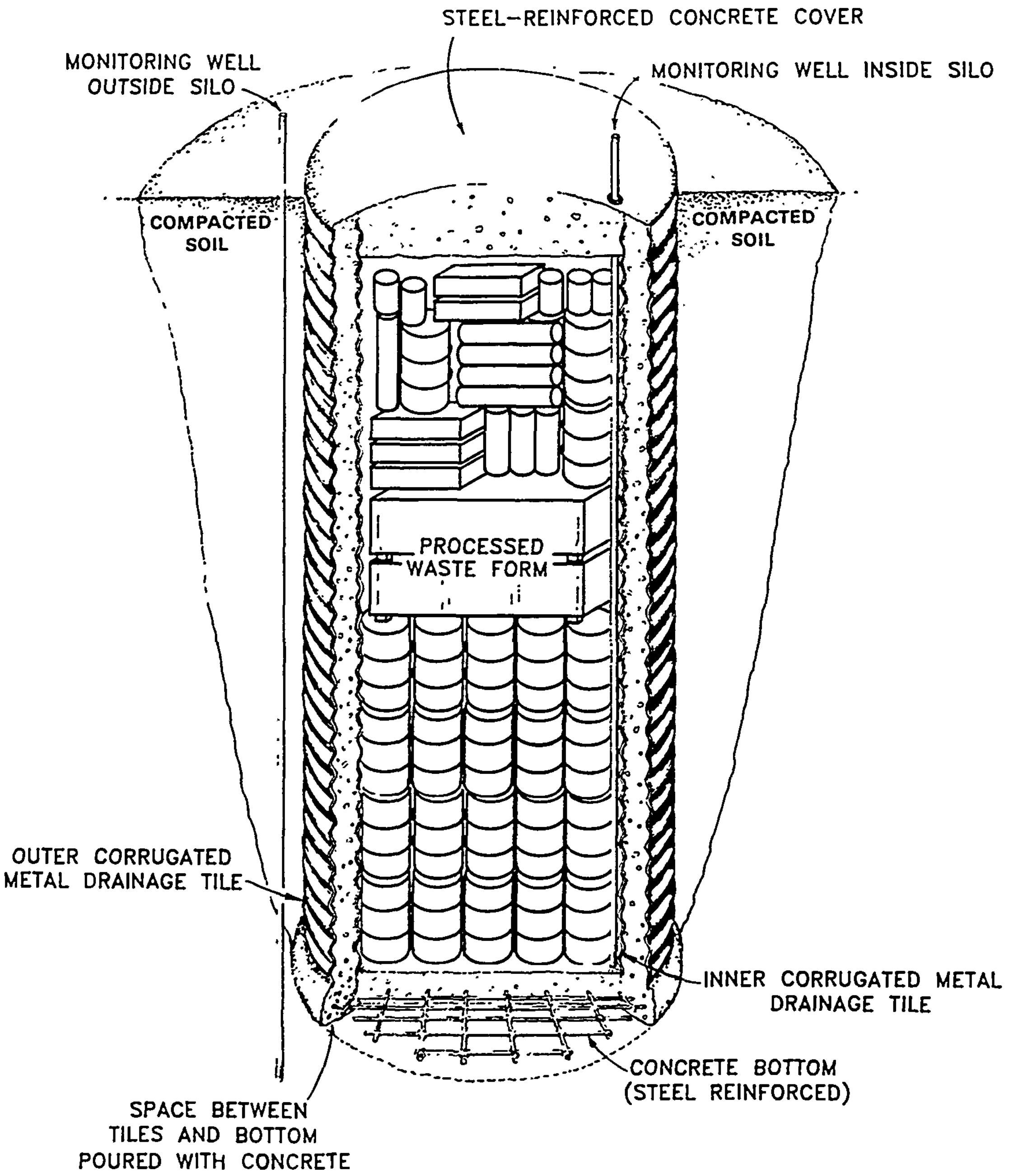

Fig. 2.2. Representative silo-type disposal facility modeled by SOURCE2. (SOURCE2 is applicable to silo, well-in-silo, and trench disposal technologies.)

Moments and shears required for cracking vary as the engineered facility degrades. The leaching routines calculate the release rate of nuclides to the environment. A detailed illustration of the logic flow used in the SOURCE computer codes to model the aforementioned processes is provided in Fig. 2.3. The structural analysis is performed once at the beginning of a simulation. The concrete and metal barrier degradation and cracking analyses are performed each year by using annual time-steps. Nuclide release rates are calculated by using monthly time-steps.

Before the annual simulation begins, a structural analysis of the disposal facility is conducted to establish the moments and forces placed on the various structural components. For the roof, walls, and floor, the SOURCE codes calculate the uniform load, bending moments resulting from uniform loading, and shear and compressive forces. The walls are subjected also to hydrostatic pressures caused by the backfill and the waste. Bending moments and shear forces are calculated for the walls based on these hydrostatic pressures. The bending moments and shear forces attributed to hydrostatic pressures are added to the bending moments and shear forces for the uniform load to give total bending moments and shear forces for the walls.

Following the structural analysis, the computer codes enter an annual loop in which chemical and physical deterioration of the concrete and steel barriers used in the disposal facility is modeled. Properties of the structural members of the facility are updated to reflect degradation and are used in cracking analyses of the roof, walls, and floor of the disposal facility to assess the structure's ability to bear the loads placed upon it. The deterioration of the concrete barriers is simulated with respect to the removal of calcium hydroxide from the cement matrix, sulfate attack of the concrete, and corrosion of steel reinforcement. Concrete component properties, including strength, thickness, and pH, are updated for each year of the simulation to reflect projected rates of deterioration. Failure rates of iron and steel (used as liners, wells, containers, etc.) are determined by using a linear failure model.

As the engineered structure is weakened by chemical and physical attack, a point is reached at which time the structure is no longer able to bear the loads placed upon it. Under these conditions, the engineered barriers will crack or, otherwise, fail. Failure is judged in terms of the capability of the facility to isolate the waste from water percolating through the disposal site. When the disposal facility is no longer hydraulically intact, the engineered barriers are assumed to offer no benefit. The cracking analyses are performed if hydraulic failure of the disposal facility has not occurred. The cracking moment, cracking shear, and ultimate strength are each calculated for the roof, walls, and floor and compared with the moments and forces calculated in the structural analysis. If the

ORNL DWG 93A-738

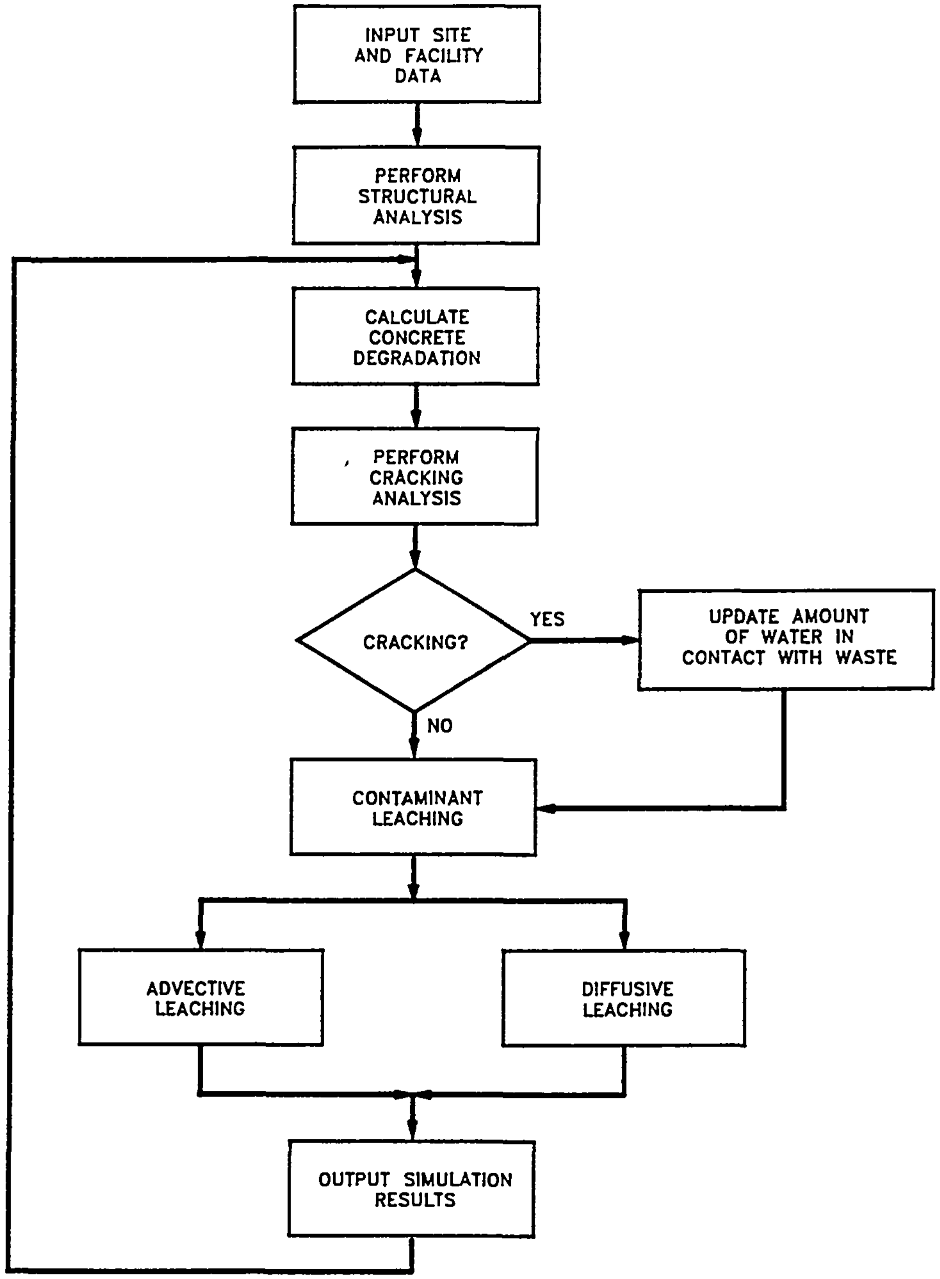

Fig. 2.3. Logic flow of the SOURCE computer codes.

calculated moments or shears exceed the cracking moments or shear forces, the structural member is projected to crack. Fracture characteristics, including depth, spacing, and width, are calculated with the onset of cracking. Cracking or spalling of concrete members of the disposal facility may result from corrosion of the steel reinforcement. In the event of the former, fracture characteristics are calculated. Concrete-member thicknesses are updated in the event that spalling of the concrete surface occurs. Figure 2.4 demonstrates the logic flow of the structural and cracking analyses that are performed to estimate the time of failure of the disposal facility.

Radionuclide release rates from waste disposal facilities are a function of the integrity of the waste (or waste form) and the engineered barriers used in construction of the facility (e.g., concrete and metal containers). When intact, these barriers minimize the contact of water with the waste, thereby minimizing releases of radionuclides. As the barriers deteriorate, over time, water can more readily contact the waste and mobilize radionuclides, thus accelerating releases to the environment.

The SOURCE computer codes consider two mechanisms through which waste radionuclides are released into the environment: advection (bulk flow driven by hydraulic pressure differences) and diffusion (nuclide movement driven by concentration differences). The calculated total release rate resulting from advection and diffusion is compared with the rate of release dictated by the solubility limit of the nuclide in water. If the solubility limit is exceeded, the release rate is adjusted to the solubility-limited rate. As a disposal facility degrades, the percolation rate of water through the waste increases. Thus, except for cases constrained by solubility, advective releases will increase with degradation and, in general, dominate the total release. The total release is divided into two components: one that recharges to groundwater and a second that flows laterally in the shallow-subsurface flow region of the site.

2.3 CODE STRUCTURE

The SOURCE1 code consists of the main program, 18 subroutines, and 3 functions; the SOURCE2 code consists of the main program, 20 subroutines, and 3 functions. The code hierarchy of the SOURCE1 and SOURCE2 code is illustrated in Figs. 2.5 and 2.6, respectively. A brief description of the functions performed by the program modules is provided in Table 2.1.

The SOURCE codes require keyboard input to specify the name of the primary input file. This file contains data describing the disposal site and features of the disposal facility under consideration and nuclide-specific information. Also, the name of the file containing site-specific water infiltration

ORNL DWG 93A-737

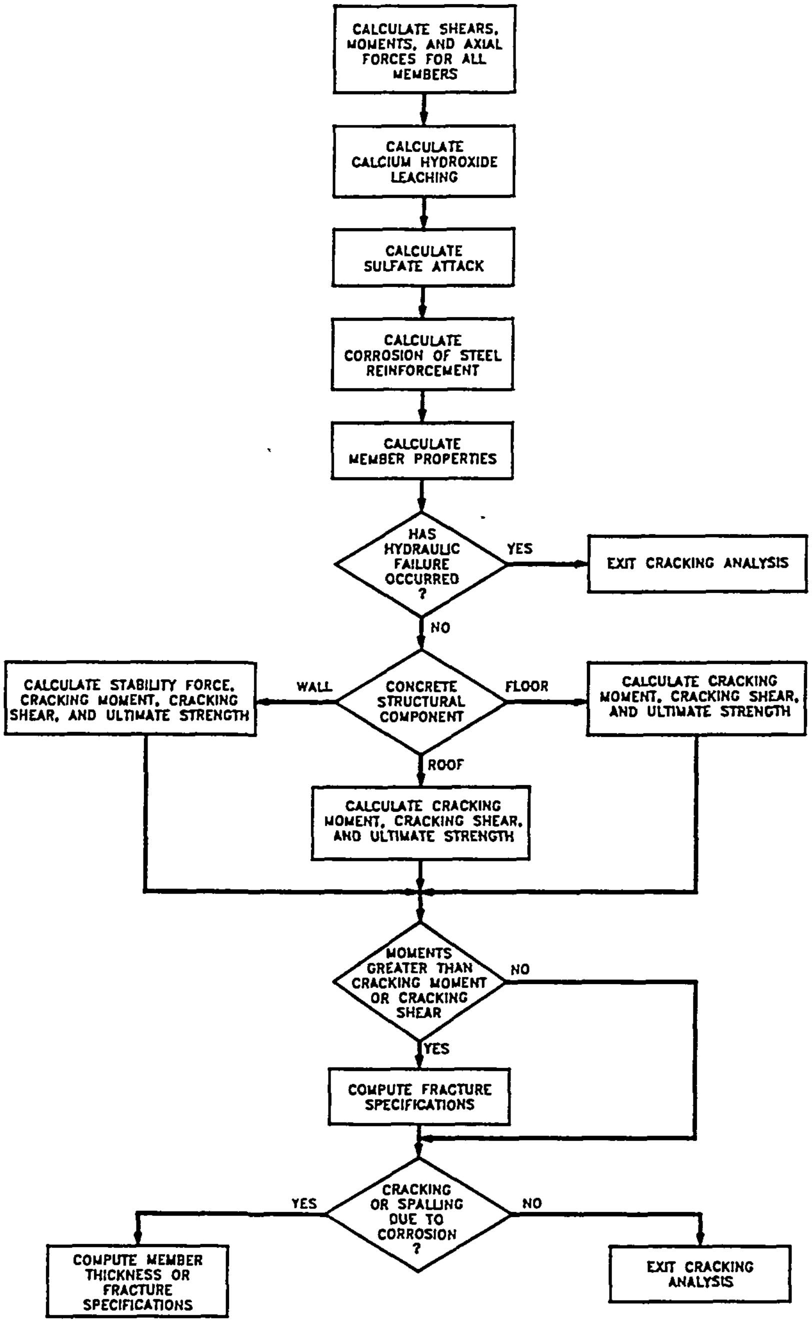

Fig. 2.4. Logic flow of the concrete-degradation and -cracking subroutines.

ORNL DWG 96-4996

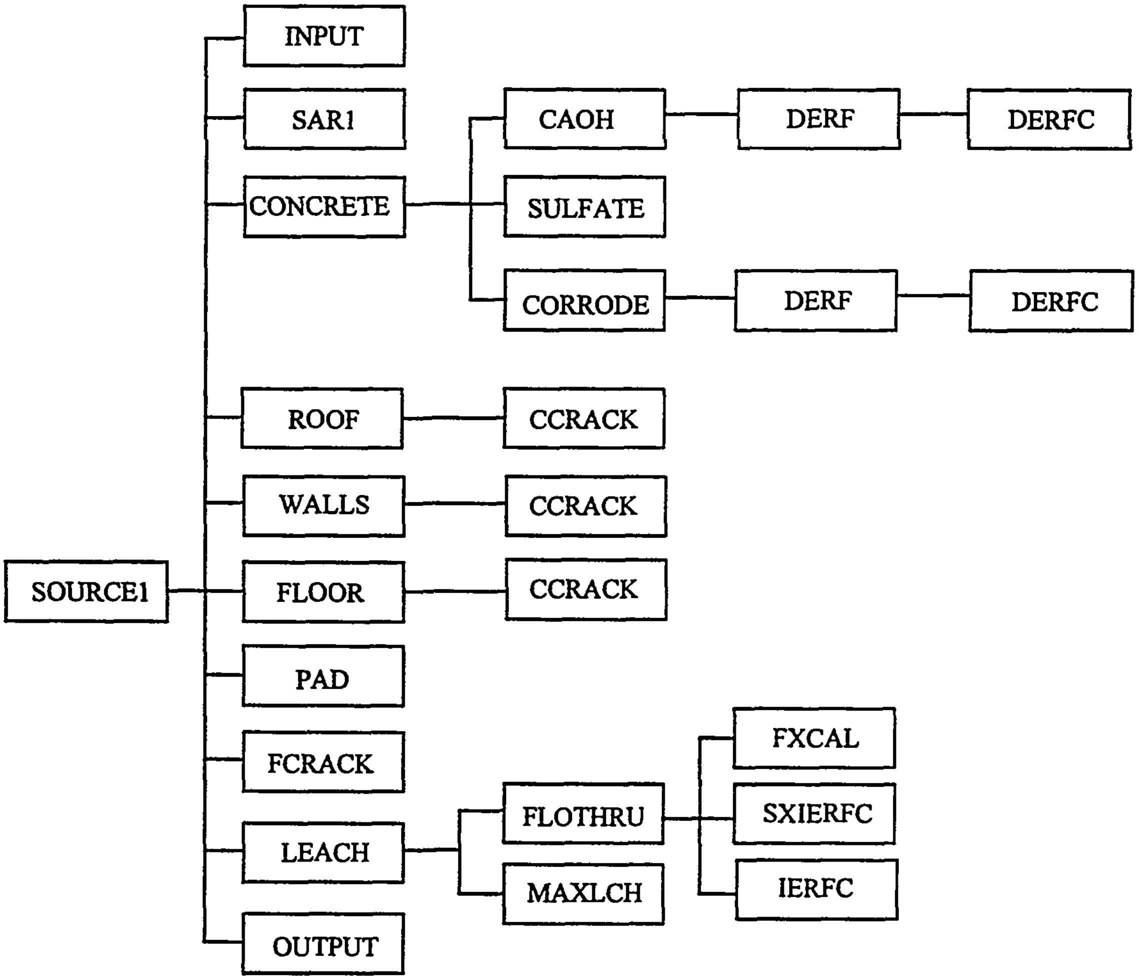

Fig. 2.5. SOURCE1 code hierarchy for modeling tumulus-type disposal facilities.

ORNL DWG 96-4997

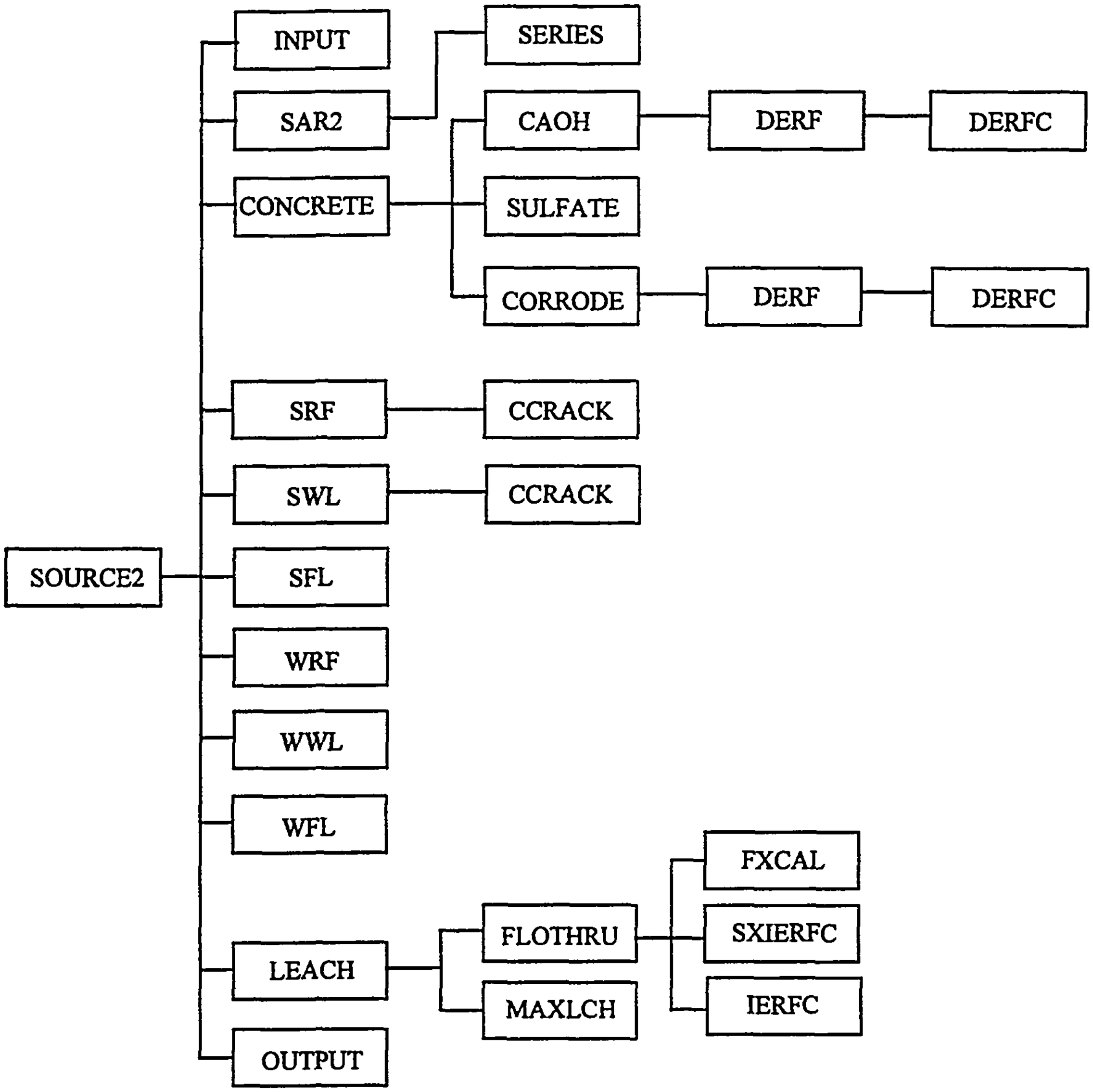

Fig. 2.6. SOURCE2 code hierarchy for modeling disposal silos and wells.

Table 2.1. SOURCE1 and SOURCE2 program module description

Module	Purpose
SOURCE	Main program, coordinates subroutine calls
CAOH	Calculates changes in concrete member strength and pH resulting from leaching of calcium hydroxide
CCRACK	Performs cracking analysis for cracking resulting from corrosion
CONCRETE	Coordinates calls to concrete degradation subroutines
CORRODE	Calculates initiation and propagation of corrosion of steel reinforcement
DERF	Calculates the error function [erf(z)]
DERFC	Calculates the complimentary error function [erfc(z)]
FCRACK	Calculates number of vaults that have undergone cracking
FLOOR	Performs cracking analysis for vault floor
FLOTHRU	Calculates radionuclide releases as a result of diffusion
FXCAL	Solves transcendental equation used in calculating releases as a result of diffusion
IERFC	Performs the recursive computation of the repeated integrals of the complimentary error function
INPUT	Reads input data file and performs preliminary calculations
LEACH	Coordinates leaching calculations and calculates radionuclide releases resulting from advection
MAXLCH	Calculates solubility limits on radionuclide leaching
OUTPUT	Prints summaries of input information, concrete analyses, inventory, and radionuclide release rates
PAD	Predicts the performance of the reinforced concrete pad under the vaults
ROOF	Performs cracking analysis for vault roof
SAR1	Performs structural analysis for vault roof, walls, and floor
SAR2	Performs structural analysis for silo and well roof, wall, and floor
SERIES	Approximates a series used in calculating the maximum shear force on a silo or well floor
SFL	Performs cracking analysis for silo floor

Table 2.1. (continued)

Module	Purpose
SRF	Performs cracking analysis for silo roof
SULFATE	Calculates change in concrete member thickness resulting from sulfate attack
SWL	Performs cracking analysis for silo wall
SXIERFC	Function used in calculating releases as a result of diffusion
WALLS	Performs cracking analysis for vault walls
WFL	Performs cracking analysis for well floor
WRF	Performs cracking analysis for well roof
WWL	Performs cracking analysis for well wall

data is provided in this file. The water infiltration data are composed of multiple sets of monthly infiltration values; each of these sets corresponds to defined time periods during the disposal facility performance simulation.

The SOURCE1 code has seven output options for summarizing the results of the simulation, and the SOURCE2 code has five options for summarizing the results. Both SOURCE1 and SOURCE2 have options for providing a summary of the input data, concrete analyses, remaining inventories, leach rates, cumulative amounts leached, leach rates attributed to advection and diffusion, and the leach rates partitioned to the recharge and lateral components. In addition, SOURCE1 summarizes the inventory and leach rate for advection and diffusion for intact and cracked vaults. A detailed discussion of the structure of the input and output files can be found in Sect. 4.

2.4 SYSTEM REQUIREMENTS

The SOURCE computer codes conform to the American National Standards Institute, Inc. (ANSI), FORTRAN Programming Language standard X3.9–1978. The codes have been executed on UNIX workstations and IBM™-compatible personal computers.

3. CONCEPTUAL AND MATHEMATICAL MODELING METHODOLOGY

The conceptual and mathematical modeling methodology used in the SOURCE computer codes is discussed in the following subsections. This discussion considers the approaches taken in (1) modeling concrete degradation, (2) performing the structural and cracking analyses for the disposal structures, (3) partitioning water through the disposal facility, and (4) modeling advective and diffusive releases of radionuclides from waste.

At this point, a note on the units used in this manual is in order. The reader will notice that a mixture of metric and English units is used in the definitions presented in this and other sections. This mixture is a carryover from the original versions of the SOURCE1 and SOURCE2 codes.[1] The use of two unit systems stems primarily from the structural analysis and cracking calculations being performed using English units while the degradation and leaching calculations are performed using metric units. Although working with two sets of units is somewhat inconvenient, it was determined that the cost of converting SOURCE1 and SOURCE2 entirely to metric units was not justified. Therefore, the user is strongly cautioned to observe the unit requirements for each input parameter. Specific unit requirements for each input variable are presented in Sect. 4.

3.1 CONCRETE DEGRADATION MODELING

Modes of concrete degradation are considered in terms of surface- and bulk-attack mechanisms. Surface-attack mechanisms are initiated at the surface of the concrete component and progress inward, over time. Bulk-attack mechanisms modify the properties of the entire concrete component uniformly.

Sulfate attack is generally considered the most significant surface attack mechanism in the context of waste repositories.[6] In areas characterized by cold winters, freeze-thaw cycling may also represent a serious threat to concrete in disposal facilities. In terms of the bulk-attack processes, the most notable degradation processes are likely to be calcium hydroxide leaching and alkali-aggregate attack.

Corrosion of reinforced steel may also undermine the ability of engineered disposal facilities to isolate waste therein from the environment. This process differs from the surface- and bulk-attack processes noted previously because it does not directly alter the properties of the concrete.

The models used in simulating concrete degradation are discussed in Sects. 3.1.1 through 3.1.3. The deterioration processes considered in the SOURCE computer codes include sulfate attack, calcium hydroxide leaching, and corrosion of steel reinforcement.

3.1.1 Sulfate Attack

Sulfate attack generally manifests itself in the form of expansion and, ultimately, cracking of concrete. It may also result in a progressive loss of strength and mass resulting from deterioration in the cohesiveness of the cement hydration products.

Three steps are recognized in the deterioration of concrete as a result of sulfate attack:[7]

- Sulfate ions from the environment penetrate into the concrete, usually by diffusion.
- Sulfate ions react expansively with certain aluminum-containing phases in the concrete to form ettringite.
- The resulting internal expansion causes stress, cracking, and exfoliation of the concrete surface.

These aspects of the degradation process are incorporated into the sulfate attack model used in the SOURCE computer codes.

The sulfate attack model is based on the work of Atkinson and Hearne.[7] In this model, the reaction zone is assumed to spall out when it reaches a critical thickness given by

$$X_{spall} = \frac{2\alpha\gamma(1 - \mu_c)}{E(\beta C_e)^2} , \tag{3.1}$$

where

X_{spall} = reaction zone thickness at which spalling occurs (m),
α = roughness factor for fracture path (unitless),
γ = fracture surface energy of concrete (J/m^2),
μ_c = Poisson's ratio for concrete (unitless),
E = Young's modulus (Pa),
β = linear strain caused by a mole of sulfate reacted in 1 m^3 (m^3/mol), and
C_e = concentration of sulfate as ettringite at the time at which spalling occurs (mol/m^3).

This critical thickness is achieved at a time

$$t_{spall} = \frac{X_{spall}^2 C_e}{2 D_i c_o} , \tag{3.2}$$

where

t_{spall} = time at which spalling occurs (s),
D_i = "intrinsic" diffusion coefficient of sulfate ions in water-saturated cement (m^2/s), and
c_o = groundwater sulfate concentration (mol/m^3).

The rate of degradation, R, is defined as X_{spall}/t_{spall} (m/s). Then, based on Eqs. (3.1) and (3.2), R is given by

$$R = \frac{E \beta^2 c_o C_e D_i}{\alpha \gamma (1 - \mu_c)} . \tag{3.3}$$

As sulfate attack progresses into the concrete member, it is assumed that the affected layers spall off, thus effectively reducing the thickness of the concrete member.

It is necessary to use an iterative method to determine the concentration of sulfate as ettringite, C_e, and the degradation rate caused by sulfate attack. The starting approximation for C_e is calculated assuming that the alumina has been completely converted to ettringite in the reacted zone.[7] Zero-order values of the X_{spall}, t_{spall}, and R are calculated on this basis, and t_{spall} is compared with the time required for the reaction to go to completion. If t_{spall} is not great enough to permit complete reaction, t_{spall} and C_e are iterated to self-consistency using the reaction kinetics expression described in ref. 7.

3.1.2 Calcium Hydroxide Leaching

Calcium hydroxide [$Ca(OH)_2$] leaching results in a loss of strength in the concrete as well as a lowering of the pH of the material. A loss of strength will affect the concrete structure's ability to withstand the loads placed upon it. Declines in the pH of the concrete may lead to depassivation of the steel reinforcement, thereby promoting corrosion of the steel.

$Ca(OH)_2$ may be leached from the concrete because of diffusion and advection. The loss of $Ca(OH)_2$ from concrete members from diffusion is calculated by solving the following equation:

$$\frac{df_t}{dt} = - D_y \frac{d^2 f_t}{dy^2} \quad , \tag{3.4}$$

where

f_t = fraction of $Ca(OH)_2$ remaining in concrete member as a function of position and time (unitless),

t = time (s),

D_y = effective diffusion coefficient of $Ca(OH)_2$ in concrete (m^2/s), and

y = distance from centerline (m).

The initial conditions that apply are

$$f_t(y, t = 0) = \begin{cases} 1 & \text{for } |y| < \frac{\hat{W}}{2} \\ 0 & \text{for } |y| > \frac{\hat{W}}{2} \end{cases} \quad , \tag{3.5}$$

where

$\hat{W}$ = width of concrete member (m).

The equation for f_t is given by

$$f_t = 0.5 \text{ erf } \frac{(y + \hat{W}/2)R_f}{2\left(D_y t\right)^{0.5}} - 0.5 \text{ erf } \frac{(y - \hat{W}/2)R_f}{2\left(D_y t\right)^{0.5}} \quad , \tag{3.6}$$

where

R_f = retardation factor for $Ca(OH)_2$ in concrete (unitless).

The $Ca(OH)_2$ retardation factor, R_f, is given by

$$R_f = 1 + \frac{Ca_c}{\epsilon_c Ca_p} \quad , \tag{3.7}$$

where

Ca_c = $Ca(OH)_2$ concentration in concrete (mol/L),
ϵ_c = concrete porosity (dimensionless), and
Ca_p = $Ca(OH)_2$ concentration in concrete pore solution (mol/L).

The potential for leaching of $Ca(OH)_2$ through advective mechanisms will depend upon the nature of the groundwater. If the groundwater is saturated or supersaturated with calcium carbonate, no dissolution of $Ca(OH)_2$ will occur. Groundwater which is not saturated with calcium carbonate may leach $Ca(OH)_2$ as the groundwater passes through the concrete.

Langelier[8] has developed the calcium carbonate saturation (or Langelier) index as a means of characterizing the degree of calcium carbonate saturation of groundwater. The index takes into account the effects of temperature, total dissolved solids, total alkalinity, pH, and calcium content on the saturation characteristics of the groundwater. A negative value for the Langelier index denotes a groundwater which is not saturated with calcium carbonate [i.e., one capable of leaching $Ca(OH)_2$ from concrete]. Index values equal to or greater than zero indicate calcium carbonate saturation.

Data taken from Langelier[8] are used in a regression to estimate the saturation index as a function of the total dissolved solids and temperature of the groundwater. When predicted values of the index are positive, losses of $Ca(OH)_2$ are modeled to occur as a result of diffusion only. When the groundwater is not saturated with calcium carbonate (i.e., the Langelier index is negative), advective leaching of $Ca(OH)_2$ is calculated using[9]

$$Ca_l = I \frac{Ca_p}{C_t Ca_c} \quad , \tag{3.8}$$

where

Ca_l = fractional groundwater release rate of $Ca(OH)_2$ ($year^{-1}$),
I = water percolation rate through vault (m/year), and
C_t = concrete-member thickness (m).

The presence of other ions in the groundwater may influence the rate at which $Ca(OH)_2$ is leached from the concrete. Atkinson[9] reports that magnesium and carbonate are among the species most likely to speed the loss of $Ca(OH)_2$. The effect of these species is modeled using Eq. (3.8), replacing the pore solution concentration of $Ca(OH)_2$, Ca_p, with the sum of this concentration and the groundwater concentrations of magnesium and carbonates.

The quantities of $Ca(OH)_2$ lost from the concrete member by diffusion and advection are summed to determine the total amount of the constituent leached from the concrete. The concentration of $Ca(OH)_2$ in the concrete is adjusted downward to reflect these losses. All $Ca(OH)_2$ leached from the concrete is assumed to be drawn from the calcium-silicate-hydrate (C-S-H) system of the concrete. The calcium incorporated into the relatively less-soluble phases of the concrete is not considered.

Changes in the pH of concrete as a result of the loss of $Ca(OH)_2$ have been well documented.[9] The pH of the concrete is maintained at levels greater than approximately 12.5 in the presence of alkalis, NaOH, and KOH. As these highly soluble species are lost because of leaching, the pH declines until it reaches 12.5, at which point the pH of the concrete is controlled primarily by the $Ca(OH)_2$ content of the concrete.

Changes in the pH of the concrete are modeled as alkalis and $Ca(OH)_2$ are leached from the concrete. Based on the data of Greenberg and Chang,[10] the pH is modeled to decline linearly from the initial pH of the concrete, as specified by the user, to 12.5 in direct proportion to the reduction in NaOH and KOH in the concrete. The rates of loss of these species from leaching by diffusion and advection are calculated using Eqs. (3.6) and (3.8), respectively.

Following the complete loss of NaOH and KOH from the concrete, changes in the pH of the concrete are modeled as a function of the $Ca(OH)_2$ content. Using data from the work of Greenberg and Chang,[10] concrete pH was regressed on the Ca:Si ratio of the material. As $Ca(OH)_2$ is leached from the concrete, the Ca:Si ratio is updated, and the pH of the concrete is estimated using this regression.

In addition to the pH effects noted, the loss of $Ca(OH)_2$ will result also in a reduction in the strength of the concrete. The loss in strength has been estimated to be approximately 1.5% for every 1.0% of the $Ca(OH)_2$ leached from the concrete.[11] Based on this relationship, the compressive strengths of the concrete members are updated to reflect losses of $Ca(OH)_2$.

3.1.3 Corrosion of Steel Reinforcement

The damage to concrete resulting from the corrosion of steel reinforcement manifests itself in expansion, cracking, and spalling of the concrete member. The reinforced concrete member may suffer structural damage because of (a) the loss of the bond between the steel and concrete and (b) the loss of reinforcement cross-sectional area.

Steel reinforcement is generally passivated because of the alkaline nature of the liquid phase in the concrete pores and, hence, does not undergo corrosion. The passive layer on the steel may be destroyed through a direct lowering of the pH of the concrete via carbonation or because of chloride ion penetration to the steel. Both mechanisms of depassivation are considered in the SOURCE computer codes.

Carbonation of the concrete occurs as a result of the diffusion of carbon dioxide (CO_2) into the material. The depth of carbonation is given by[12]

$$X = kt^{0.5} \quad , \tag{3.9}$$

where

X = depth of carbonation (m) and
k = carbonation coefficient ($m/s^{0.5}$).

Given the value of k, the depth of penetration of the carbonation front into the concrete can be determined for any specified time.

The carbonation coefficient is calculated using[12]

$$\frac{C_x - C_s}{g\left(k/2D_{CO2}^{0.5}\right)} + C_x - C_f = 0 \quad , \tag{3.10}$$

where

C_x = concentration of CO_2 bound in concrete (mol/L),
C_s = CO_2 concentration at surface of concrete (mol/L),
C_f = CO_2 concentration ahead of carbonation front (mol/L), and
$g\left(k/2D_{CO2}^{0.5}\right)$ = function, where
D_{CO2} = diffusion coefficient of CO_2 in concrete (m^2/s).

The function $g\left(k/2D_{CO2}^{0.5}\right)$ is given by

$$g\left(k/2D_{CO2}^{0.5}\right) = \pi^{0.5}\frac{k}{2D_{CO2}^{0.5}}\exp\left(\frac{k^2}{4D_{CO2}}\right)\operatorname{erf}\left(\frac{k}{2D_{CO2}^{0.5}}\right) \quad . \tag{3.11}$$

Equations (3.10) and (3.11) are combined to arrive at a solution for k, the carbonation coefficient. Assuming that the concentration of CO_2 ahead of the carbonation front is zero (i.e., that the carbonation front is discontinuous), this solution can be simplified to yield

$$k^2 = \frac{4D_{CO2}}{\pi^{0.5}}\left(\frac{C_s}{C_x} - 1\right) \quad . \tag{3.12}$$

When using Eq. (3.12) to evaluate k, the model takes into account that a portion of the CO_2 diffusing into the concrete is bound by concrete constituents and does not penetrate to the steel reinforcement. This bound CO_2 plays no role in depassivation. The amount of CO_2 bound in the concrete is set equal to amount of hydrated lime in the concrete.[12] The quantity of hydrated lime is calculated as the product of the CaO content in the concrete and the degree of hydration. The degree of hydration, estimated based on the water/cement ratio for Portland cements,[12] is given by

$$H_f = 0.4 + 0.5(WCR) \quad , \tag{3.13}$$

where

H_f = fraction of hydrated CaO (unitless) and
WCR = water-cement ratio (unitless).

Given the carbonation coefficient, k, the depth of carbonation is calculated using Eq. (3.9). As stated previously, this solution assumes that the carbonation front is discontinuous. At that time when the front has penetrated to the depth of the steel reinforcement, depassivation of the steel is assumed to occur, and corrosion is thus initiated.

Steel reinforcement may also be depassivated as a result of the penetration of chloride ions to the steel surface. Using a standard solution to Fick's first law of diffusion, the chloride ion concentration at the steel is calculated as

$$CL_s = CL_i + \left(CL_{gw} - CL_i\right)\left[1 - \mathrm{erf}\left(\frac{C_c}{2\left(D_{Cl}t\right)^{0.5}}\right)\right] , \tag{3.14}$$

where

CL_s = chloride ion concentration at steel reinforcement (mol/L),
CL_i = initial chloride ion concentration in concrete (mol/L),
CL_{gw} = chloride ion concentration in groundwater (mol/L),
C_c = concrete cover thickness (m), and
D_{Cl} = effective diffusivity of chloride in concrete (m^2/s).

The concentration of chloride ions at the steel reinforcement required to depassivate the steel has been considered by numerous investigators. Hausmann[13] found that the pH of the concrete had an effect on the level of chloride ions required to initiate corrosion. In studies carried out using NaOH and $Ca(OH)_2$ solutions, it was found that a chloride ion to hydroxide ion concentration ratio of 0.61 was sufficient to depassivate the steel.

Using the results of Hausmann, a chloride-ion-to-hydroxide-ion ratio of 0.61 is assumed to result in depassivation of the steel. The hydroxide ion concentration used in calculating this ratio is determined using the modeled concrete pH. As discussed in Sect. 3.1.2, the pH of the concrete changes with time as NaOH, KOH, and $Ca(OH)_2$ are leached from the material.

Upon depassivation of the steel reinforcement by either carbonation or chloride penetration, corrosion is modeled to occur at a rate determined by the rate of diffusion of oxygen to the steel. The molar flow of oxygen to the surface of the steel reinforcement is modeled using Fick's first law of diffusion:

$$J_o = - D_o A \frac{d\left[O_2\right]}{dx} , \tag{3.15}$$

where

J_o = oxygen flux at the steel reinforcement (g/s),

D_o = effective diffusivity of oxygen through concrete (cm^2/s),

A = surface area over which oxygen diffuses to the reinforcement (cm^2), and

$\frac{d[O_2]}{dx}$ = dissolved oxygen concentration gradient (g/cm^4).

The rate of oxygen consumption by the corrosion reaction is assumed to be greater than the rate of oxygen diffusion to the reaction surface. Under these conditions, the corrosion rate is limited by the flux of oxygen at the steel reinforcement.

The use of epoxy-coated steel reinforcement may delay the onset of corrosion by isolating the steel from aggressive ions and oxygen. The use of epoxy coating is not assumed to delay the time of depassivation of the reinforcement. However, upon depassivation, the coating is assumed to prevent corrosion as long as it remains intact.

The effectiveness of epoxy coating on steel reinforcement is modeled using a linear failure function. Using the time at which failure of the coating begins and the time required for all epoxy coating to fail, a fraction of the reinforcement coating which has failed is calculated. This failure fraction is used to linearly adjust the projected rate of corrosion downward.

Corrosion of components other than steel reinforcement will also affect the long-term performance of LLW disposal facilities. For example, the metal boxes placed inside tumulus vaults, the corrugated steel liners used in the construction of disposal silos, and the cast iron pipes used in well construction will all eventually fail as a result of corrosion.

Corrosion of steel and iron barriers used in the tumulus, silo, well-in-silo, and well disposal technologies is considered in the SOURCE computer codes. Failure rates of these barriers are modeled using linear failure functions. The user specifies the time at which corrosion of the metal component begins and the number of years required, following this time, for the member to fail completely. Using these input data, a failure fraction is calculated for each year of the simulation.

3.2 CONCRETE STRUCTURAL AND CRACKING ANALYSES

The structural and cracking analyses serve two distinct purposes in modeling the long-term performance of LLW facilities. The structural analysis considers the loads placed on the disposal facility to determine the bending moments, shears, axial tension, and compressive forces placed on the various structural components. Because these loads vary with the structural component under consideration, this analysis is carried out for the roof, walls, and floor of each disposal facility.

The cracking analyses are concerned with the ability of the disposal facility to bear the loads placed upon it. Bending moments, shears, axial tensions, and compressive forces calculated as part of the structural analysis are compared with loads and forces at which structural failure will occur to determine the structural integrity of the disposal facility. The cracking analyses must account for the changes in concrete properties projected to occur because of physical and chemical attack. As such, it is conducted for each year of the simulation, or until hydraulic failure of the disposal facility is complete.

The structural and cracking analyses must address the structural features of each disposal facility. Even though the disposal silos and wells share a number of common features, both of these technologies differ significantly from the tumulus technology. Consequently, the following discussion considers the silos and wells, and tumulus separately.

The models used in the structural and cracking analyses are generally applicable to performance analysis of concrete, steel, and iron structural components. However, the manner in which some of these models are applied is specific to the disposal technologies in use at SWSA 6 and planned for L-II. Consequently, general application of the SOURCE computer codes to other disposal configurations requires extreme caution.

3.2.1 Tumulus Technology

The long-term performance of a tumulus disposal facility is a function of the performance of the individual vaults of which it is composed. The structural and cracking analyses performed to model the behavior of these vaults are discussed in Sects. 3.2.1.1 and 3.2.1.2, respectively. Additionally, models for the degradation and failure of the tumulus pad and leachate collection system are presented in Sects. 3.2.1.3 and 3.2.1.4.

3.2.1.1 Structural Analysis

Each structural component of the vaults has unique loading conditions placed upon it. As such, separate structural analyses are conducted for the roof or lid, walls, and floor of the vaults. A description of these analyses follows.

3.2.1.1.1. Vault roof. The roof is analyzed structurally as a simply supported, or hinged, slab. The uniform load on the roof of a vault in layer i of the tumulus disposal facility is calculated as

$$q_r = s\rho_s + ih_r\rho_c + (i-1)\left[h_w\rho_w + \left(h_f + h_r\right)\rho_c\right] \;, \tag{3.16}$$

where

q_r = uniform load on vault roof in layer i (lb/in.^{2}),
s = soil cover thickness (in.),
ρ_s = density of soil cover (lb/in.^{3}),
i = layer number (unitless),
h_r = roof thickness (in.),
ρ_c = density of reinforced concrete (lb/in.^{3}),
h_w = waste thickness (in.),
ρ_w = density of waste (lb/in.^{3}), and
h_f = floor thickness (in.).

Thermal loads on the vaults are not considered because the insulating properties of the cover material minimize thermal gradients across the concrete structural components.

The deflection of the simply supported rectangular roof resulting from the uniform load is given by

$$W(x,y) = \frac{16q_rw_r}{\pi^6D_r}\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}\frac{\sin\dfrac{m\pi x}{a}\sin\dfrac{n\pi y}{b}}{mn\left(\dfrac{m^2}{a^2}+\dfrac{n^2}{b^2}\right)^2} \;, \tag{3.17}$$

where

W(x, y) = deflection of roof at location (x, y) (in.),
w_r = width of roof in (x, y) direction (in.),
m, n = 1,3,5, . . .,
a = width of waste cell (in.),
b = length of waste cell (in.), and
D_r = flexural rigidity of the roof (lb-in.^{2}).

The flexural rigidity of the roof is calculated using

$$D_r = \frac{E_c h_r^3 w_r}{12\left(1 - \mu_c^2\right)} \quad , \tag{3.18}$$

where

E_c = modulus of elasticity of concrete (lb/in.^{2}).

Based on Eqs. (3.17) and (3.18), the bending moments resulting from uniform loading as a function of location on the roof are calculated as

$$M_x = q_r a^2 \left[\frac{16}{\pi^4} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{\sin \frac{m\pi x}{a} \sin \frac{n\pi y}{b}}{\frac{n}{m}\left(m^2 + \frac{n^2}{\left(\frac{b}{a}\right)^2}\right)^2} \right.$$

$$\left. + \mu_c \frac{\sin \frac{m\pi x}{a} \sin \frac{n\pi y}{b}}{\left(\frac{b}{a}\right)^2 \left(\frac{m}{n}\right) \left(m^2 + \frac{n^2}{\left(\frac{b}{a}\right)^2}\right)^2} \right] \tag{3.19}$$

and

$$M_y = q_r b^2 \left[\frac{16}{\pi^4} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{\sin \frac{m\pi x}{a} \sin \frac{n\pi y}{b}}{\frac{m}{n} \left(\frac{m^2}{\left(\frac{a}{b}\right)^2} + n^2 \right)^2} \right.$$

$$\left. + \mu_c \frac{\sin \frac{m\pi x}{a} \sin \frac{n\pi y}{b}}{\left(\frac{a}{b}\right)^2 \left(\frac{n}{m}\right) \left(\frac{m^2}{\left(\frac{a}{b}\right)^2} + n^2 \right)^2} \right] , \qquad (3.20)$$

where

M_x = bending moment resulting from uniform loading in the x-direction parallel to width of roof (lb-in./in.) and

M_y = bending moment resulting from uniform loading in the y-direction parallel to length of roof (lb-in./in.).

Uniform loads on the vault roof result in shear forces upon that component as well. These forces are calculated as

$$Q_x = q_r a \left\{ \frac{16}{\pi^3} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \cos \frac{m\pi x}{a} \cdot \sin \frac{n\pi y}{b} \right.$$

$$\left. \left[\frac{m^2}{n \left(m^2 + n^2 \left(\frac{a}{b}\right)^2 \right)^2} + \frac{n}{\left(m^2 \left(\frac{b}{a}\right) + n^2 \left(\frac{a}{b}\right) \right)^2} \right] \right\} \qquad (3.21)$$

and

$$Q_y = q_r b \left\{ \frac{16}{\pi^3} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \sin \frac{m\pi x}{a} \cos \frac{n\pi y}{b} \left[\frac{m}{\left(m^2\left(\frac{b}{a}\right) + n^2\left(\frac{a}{b}\right)\right)^2} + \frac{n^2}{m\left(m^2\left(\frac{b}{a}\right)^2 + n^2\right)^2} \right] \right\} , \qquad (3.22)$$

where

Q_x = shear force resulting from uniform loading in the x-direction (lb/in.) and

Q_y = shear force resulting from uniform loading in the y-direction (lb/in.).

3.2.1.1.2. Vault walls. The vault walls are subject to vertical, or uniform, loads and hydrostatic pressures. The uniform loads are calculated for a vault in layer i of the tumulus disposal facility using

$$q_w = \rho_s\left(\frac{h_r}{2} i + \left(h_f + h_w\right)(i - 1) + s \right)\left(1 - \sin f_s\right) , \qquad (3.23)$$

where

q_w = uniform load on vault wall in layer i (lb/in.^{2}) and

f_s = friction angle of soil backfill around tumulus (deg).

Hydrostatic pressures on the vault walls are a result of lateral soil pressures from the soil backfill and from the waste and grout inside the vault. This pressure is calculated as

$$P = \rho_s\left(h_w + \frac{h_r + h_f}{2} \right)\left(h - \sin f_s\right) - \rho_w\left(h_w + \frac{h_r + h_f}{2} \right)\left(1 - \sin f_w\right) , \qquad (3.24)$$

where

P = maximum hydrostatic pressure (lb/in.^{2}) and
f_w = friction angle of waste (deg).

Bending moment calculations for the vault walls must account for the uniform loads and hydrostatic pressures on the structural components. Bending moments resulting from the uniform load are calculated using Eqs. (3.19) and (3.20), substituting the uniform load on the wall for the uniform load on the roof and changing roof dimensions to those of the wall. Bending moments caused by hydrostatic pressures are calculated using

$$M_{xh} = Pa^2 \sum_{m=1}^{\infty} (m\pi)^2 \left[\frac{2(-1)^{m+1}}{\pi^5 m^5} + \left[(1-\mu_c)A_m - 2\mu_c B_m\right]\cosh\left(\frac{m\pi y}{a}\right) + (1-\mu_c)B_m\left(\frac{m\pi}{a}\right) y\sinh\left(\frac{m\pi y}{a}\right)\right] \sin\frac{m\pi x}{a} \tag{3.25}$$

and

$$M_{yh} = Pa^2 \sum_{m=1}^{\infty} (m\pi)^2 \left[\frac{2(-1)^{m+1}}{\pi^5 m^5}\mu_c + \left[(\mu_c-1)A_m - 2B_m\right]\cosh\left(\frac{m\pi y}{a}\right) + (\mu_c-1)B_m\left(\frac{m\pi}{a}\right) y\sinh\left(\frac{m\pi y}{a}\right)\right] \sin\frac{m\pi x}{a} , \tag{3.26}$$

where

M_{xh} = bending moment resulting from hydrostatic pressures in the x-direction parallel to the width of the wall (lb-in./in.) and
M_{yh} = bending moment resulting from hydrostatic pressures in the y-direction parallel to the length of the wall (lb-in./in.).

The quantities A_m and B_m are given, respectively, by

$$A_m = -\frac{\left(2 + \alpha_m \tanh \alpha_m\right)(-1)^{m+1}}{\pi^5 m^5 \cosh \alpha_m} \tag{3.27}$$

and

$$B_m = \frac{(-1)^{m+1}}{\pi^5 m^5 \cosh \alpha_m} \quad , \tag{3.28}$$

where

$$\alpha_m = \frac{m\pi b}{2a} \quad . \tag{3.29}$$

Bending moments calculated for the uniform load and hydrostatic pressure on the wall are summed to arrive at the final bending moments for the wall as a function of location. The calculations of the bending moments are repeated for each wall geometry comprising the disposal vault.

Shear forces caused by hydrostatic pressures on the vault walls are calculated as

$$Q_{xh} = 2Pa \sum_{m=1}^{\infty} (m\pi)^3 \left[\frac{(-1)^{m+1}}{\pi^5 m^5} - B_m \cosh\left(\frac{m\pi y}{a}\right)\right] \cos\left(\frac{m\pi x}{a}\right) \tag{3.30}$$

and

$$Q_{yh} = -2Pa \sum_{m=1}^{\infty} (m\pi)^3 B_m \sinh\left(\frac{m\pi y}{a}\right) \sin\left(\frac{m\pi x}{a}\right) \quad , \tag{3.31}$$

where

Q_{xh} = shear force resulting from hydrostatic loading in the x-direction (lb/in.) and

Q_{yh} = shear force resulting from hydrostatic loading in the y-direction (lb/in.).

The shear forces on vault walls resulting from uniform loads are calculated using Eqs. (3.21) and (3.22) and substituting appropriate parameters for the wall. Shear forces caused by uniform and hydrostatic loads are summed for each wall location. Calculations of shear forces are performed for each wall geometry comprising the vault.

The walls of the vaults are subject to compressive forces from the roof reaction and the weight of the walls. The compressive force, calculated as a function of height on the wall, is given by

$$F_w = R_{ry} + h_{wl} z \rho_c \quad , \tag{3.32}$$

where

F_w = compressive force on wall at height z (lb/in.),
R_{ry} = roof reaction in y direction at height z (lb/in.),
h_{wl} = thickness of wall (in.), and
z = wall height (in.)

3.2.1.1.3. Vault floor. The floor of a given vault must bear loads from the walls, including the wall weight and loads transmitted to the walls from the roof, and loads from the waste within the vault. Based on the floor geometry illustrated in Fig. 3.1, the bending moments in the region x (or y) $\leq$ a of the beam subjected to a concentrated force and moment are calculated as

$$M_x = \frac{P_x I_{1px}}{2\lambda_x\left[\sinh^2(\lambda_x w_f) - \sin^2(\lambda_x w_f)\right]} \tag{3.33}$$

and

$$M_y = \frac{P_y I_{1py}}{2\lambda_y\left[\sinh^2(\lambda_y l_f) - \sin^2(\lambda_y l_f)\right]} \quad , \tag{3.34}$$

where

M_x = bending moment in x-direction parallel to width of floor (lb-in./in.),
P_x = applied concentrated load caused by wall in x-direction (lb/in.),

I_{1px} = trigonometric function (unitless),

λ_x = $\left[\dfrac{\tilde{k} l_f}{4D_{fx}}\right]^{0.25}$ $\left(\text{in.}^{-1}\right)$,

$\tilde{k}$ = modulus of the subgrade reaction (lb/in.^{3}),

l_f = length of floor (in.),

D_{fx} = flexural rigidity of floor in the x-direction (lb-in.^{2}),

w_f = width of floor (in.),

M_y = bending moment in y-direction parallel to length of floor (lb-in./in.),

P_y = applied concentrated load caused by wall in y-direction (lb/in.),

I_{1py} = trigonometric function (unitless),

λ_y = $\left[\dfrac{\tilde{k} w_f}{4D_{fy}}\right]^{0.25}$ $\left(\text{in.}^{-1}\right)$, and

D_{fy} = flexural rigidity of floor in the y-direction (lb-in.^{2}).

ORNL-DWG 94-5975

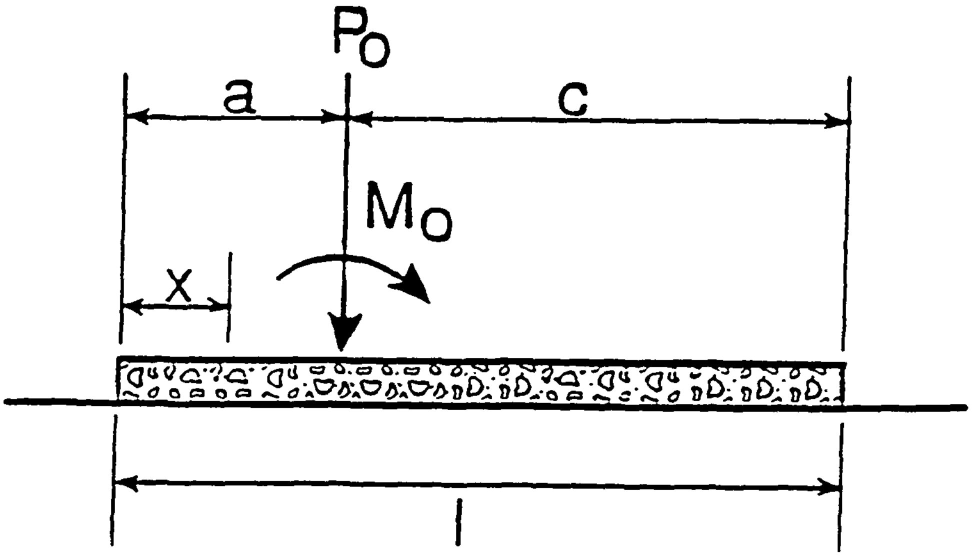

Fig. 3.1. Schematic diagram of floor geometry used in the calculation of bending moments.

Applied moments caused by the wall are not considered in the bending moment calculations because it is assumed that the floor and walls are hinged.

The concentrated load on the floor, P_x or P_y, is calculated as a function of location using

$$P_x = R_{rx} + h_{w1}\left(h_w + \frac{h_r + h_f}{2}\right)(\rho_c - \rho_w) \tag{3.35}$$

and

$$P_y = R_{ry} + h_{w1}\left(h_w + \frac{h_r + h_f}{2}\right)(\rho_c - \rho_w) \ , \tag{3.36}$$

where

R_{rx} = roof reaction in x-direction (lb/in.).

The flexural rigidity of the floor is calculated using Eq. (3.18). To calculate the quantity D_{fx} the thickness and width of the floor are substituted for h_r and w_r, respectively. The thickness and length of the floor are substituted for h_r and w_r, respectively, to calculate D_{fy}.

The parameters I_{1px} and I_{1py} are complex trigonometric functions of λ_x, λ_y, and the geometry of the structural member and are given by

$$\begin{aligned} I_{1px} = {} & 2 \sinh(\lambda_x x) \sin(\lambda_x x) \left[\sinh(\lambda_x w_f) \cos(\lambda_x a) \cosh(\lambda_x c)\right. \\ & \left. - \sin(\lambda_x w_f) \cosh(\lambda_x a) \cos(\lambda_x c)\right] - \left[\sinh(\lambda_x x) \cos(\lambda_x x)\right. \\ & \left. - \cosh(\lambda_x x) \sin(\lambda_x x)\right] \left[\sinh(\lambda_x w_f) \left\{\sin(\lambda_x a) \cosh(\lambda_x c)\right.\right. \\ & \left. - \cos(\lambda_x a) \sinh(\lambda_x c)\right\} + \sin(\lambda_x w_f) \left\{\sinh(\lambda_x a) \cos(\lambda_x c)\right. \\ & \left.\left. - \cosh(\lambda_x a) \sin(\lambda_x c)\right\}\right] \end{aligned} \tag{3.37}$$

and

$$\begin{aligned} I_{1py} = {} & 2 \sinh(\lambda_y y) \sin(\lambda_y y) \left[\sinh(\lambda_y l_f) \cos(\lambda_y a) \cosh(\lambda_y c)\right. \\ & \left.- \sin(\lambda_y l_f) \cosh(\lambda_y a) \cos(\lambda_y c)\right] - \left[\sinh(\lambda_y y) \cos(\lambda_y y)\right. \\ & \left.- \cosh(\lambda_y y) \sin(\lambda_y y)\right] \left[\sinh(\lambda_y l_f) \left\{\sin(\lambda_y a) \cosh(\lambda_y c)\right.\right. \\ & \left.- \cos(\lambda_y a) \sinh(\lambda_y c)\right\} + \sin(\lambda_y l_f) \left\{\sinh(\lambda_y a) \cos(\lambda_y c)\right. \\ & \left.\left.- \cosh(\lambda_y a) \sin(\lambda_y c)\right\}\right] \quad , \end{aligned} \tag{3.38}$$

where the parameters a and c are indicated in Fig. 3.1.

Shear forces on the floor of the vault are calculated as a function of location using

$$Q_x = \frac{P_x I_{2px}}{\left[\sinh^2(\lambda_x w_f) - \sin^2(\lambda_x w_f)\right]} \tag{3.39}$$

and

$$Q_y = \frac{P_y I_{2py}}{\left[\sinh^2(\lambda_y l_f) - \sin^2(\lambda_y l_f)\right]} \quad , \tag{3.40}$$

where

I_{2px} = trigonometric function (unitless) and

I_{2py} = trigonometric function (unitless).

The parameters I_{2px} and I_{2py} are functions of λ_x, λ_y, and the geometry of the structural member and are given by

$$\begin{aligned} L_{2px} = {} & \left[\cosh(\lambda_x x)\ \sin(\lambda_x x) + \sinh(\lambda_x x)\ \cos(\lambda_x x)\right] \left[\sinh(\lambda_x w_f)\right. \\ & \cos(\lambda_x a)\ \cosh(\lambda_x c)\ \sin(\lambda_x w_f)\ \cosh(\lambda_x a)\ \cos(\lambda_x c)\left.\right] \\ & + \sinh(\lambda_x x)\ \sin(\lambda_x s) \left[\sinh(\lambda_x w_f) \left\{\sin(\lambda_x a)\ \cosh(\lambda_x c)\right.\right. \\ & - \cos(\lambda_x a)\ \sinh(\lambda_x c)\left.\right\} + \sin(\lambda_x w_f) \left\{\sinh(\lambda_x a)\ \cos(\lambda_x c)\right. \\ & - \cosh(\lambda_x a)\ \sin(\lambda_x c)\left.\left.\right\}\right] \end{aligned} \tag{3.41}$$

and

$$\begin{aligned} L_{2py} = {} & \left[\cosh(\lambda_s y)\ \sin(\lambda_y y) + \sinh(\lambda_y y)\ \cos(\lambda_y y)\right] \left[\sinh(\lambda_y w_f)\right. \\ & \cos(\lambda_y a)\ \cosh(\lambda_y c)\ \sin(\lambda_y l_f)\ \cosh(\lambda_y a)\ \cos(\lambda_y c)\left.\right] \\ & + \sinh(\lambda_y y)\ \sin(\lambda_y s) \left[\sinh(\lambda_y l_f) \left\{\sin(\lambda_y a)\ \cosh(\lambda_y c)\right.\right. \\ & - \cos(\lambda_y a)\ \sinh(\lambda_y c)\left.\right\} + \sin(\lambda_y l_f) \left\{\sinh(\lambda_y a)\ \cos(\lambda_y c)\right. \\ & - \cosh(\lambda_y a)\ \sin(\lambda_y c)\left.\left.\right\}\right] \quad . \end{aligned} \tag{3.42}$$

3.2.1.2 Cracking Analyses

The cracking analyses are performed to assess the ability of the structural components of each vault to bear the loads placed upon it. In the event that the roof, wall(s), or floor of a vault cannot bear these loads, cracking will occur. Cracking of these components may occur as a result of shear forces or bending; cracking of the vault walls may also result from compressive loads on the structure. The manner in which these modes of cracking are modeled is discussed in the following.

Shear cracking will occur if the shear force on a concrete member exceeds the cracking shear of the member. The cracking shears for the roof, floor, and wall in the horizontal direction are calculated as the minimum of

$$V_{cr} = d_t\left(1.9\, C_{str}^{0.5} + 2500\, S_{tn} Q_x / M_x\right) \tag{3.43}$$

or

$$V_{cr} = 3.5\, C_{str}^{0.5}\, d_t \quad , \tag{3.44}$$

where

V_{cr} = shear force at which cracking occurs (lb/in.),

d_t = distance from steel reinforcement in tension to compression face of concrete (in.),

C_{str} = compressive strength of concrete (lb/in.^{2}), and

S_{tn} = area of steel reinforcement in tension per unit width (in.^{2}/in.).

The cracking shear for the wall in the vertical direction is the minimum of Eqs. (3.43) and (3.45).

$$V_{cr} = 3.5\, C_{str}^{0.5} d_t \left(1. + F_w / \left(500\, h_{w1}\right)\right)^{0.5} \quad . \tag{3.45}$$

In the event of cracking caused by shear failure, crack characteristics are determined. The depth of the single crack is the thickness of the concrete member, and the crack width is 0.013 in.

Cracking caused by bending will be initiated if the bending moments calculated for a given concrete member exceed the cracking moment for that structural component. The cracking moment is given by

$$M_{cr} = \frac{I_g f_r}{y_t a_u} \quad , \tag{3.46}$$

where

M_{cr} = cracking moment per unit width $\left(\frac{\text{lb–in.}}{\text{in.}}\right)$,

I_g = moment of inertia of concrete section (in.^{4}),

f_r = modulus of rupture (lb/in.^{2}),

y_t = distance from the centroidal axis to the tensile face of the concrete (in.), and

a_u = unit width of concrete member.

For a rectangular slab

$$I_g = \frac{a_u h_m^3}{12} \quad , \tag{3.47}$$

where

h_m = concrete member thickness (in.).

The value of y_t is given by

$$y_t = \frac{h_m}{2} \quad . \tag{3.48}$$

Axial compression force is conservatively neglected in the roof and floor.

If the bending moments exceed the cracking moment but do not exceed the ultimate strength of the concrete member, cracks will not propagate through the entire member. If, however, the bending moments exceed the ultimate strength of the structural component, cracks will span the thickness of the member. The ultimate flexural strength of a member without compressive steel is approximated using

$$M_u = \phi S_m f_y \left(d_t - \frac{C_d}{2} \right) \quad , \tag{3.49}$$

where

M_u = ultimate flexural strength (lb-in./in.),
ϕ = strength reduction factor (unitless),
f_y = yield strength of steel reinforcement (lb/in.^{2}), and
C_d = depth of the compression block (in.).

The depth of the compression block is calculated using

$$C_d = \frac{S_m f_y}{0.85\, C_{str}} \quad . \tag{3.50}$$

If all reinforcement has been lost because of corrosion, the ultimate strength is equal to the cracking moment.

Crack characteristics are calculated as fractures due to loading and propagate through a given structural component. The depth of cracking caused by bending is calculated as the distance from the surface of the concrete to the neutral axis. Crack depth is computed using the strain compatibility relation wherein the tensile crack depth is given by

$$d_{cr} = \frac{d_t}{\dfrac{St_n}{E_s} + \dfrac{St_m}{E_c}} \left[\frac{St_n}{E_s} + \epsilon_{sh} \right] + d_c \ , \tag{3.51}$$

where

d_{cr} = crack depth (in.),

ϵ_{sh} = shrinkage strain of concrete (in./in.),

d_c = concrete cover thickness on tension face (in.),

St_n = tensile stress in steel reinforcement (lb/in.^{2}),

St_m = maximum concrete compressive stress (lb/in.^{2}), and

E_s = modulus of elasticity of steel reinforcement (for this application, taken as 200,000 MPa) (MPa).

The tensile stress in steel reinforcement is calculated using

$$St_n = \left(\frac{E_s}{E_c} \right) M \left(\frac{d_t - R_t}{I_e} \right) \ , \tag{3.52}$$

where

M = bending moment due to uniform loading in x or y direction (lb–in./in.) and

I_e = effective moment of inertia per unit width of concrete member (in.^{3}).

The quantity R_t (in.) is given by

$$R_t = \frac{-\left(\hat{\beta}_1 + \hat{\beta}_2\right) + \left[\left(\hat{\beta}_1 + \hat{\beta}_2\right)^2 - 4\alpha\left(C_c\hat{\beta}_1 - d_t\hat{\beta}_2\right)\right]^{0.5}}{2\alpha_1} \quad . \tag{3.53}$$

The quantities α_1 (unitless), $\hat{\beta}_1$ (in.), and $\hat{\beta}_2$ (in.) are calculated by

$$\alpha_1 = 0.5 \quad , \tag{3.54}$$

$$\hat{\beta}_1 = \left(\frac{\pi}{S_{p2}}\right)\left(\frac{S_{d1}}{2}\right)^2\left(\frac{E_s}{E_c} - 1\right) \quad , \tag{3.55}$$

and

$$\hat{\beta}_2 = \left(\frac{\pi}{S_{p2}}\right)\left(\frac{S_{d1}}{2}\right)^2\left(\frac{E_s}{E_c}\right) \quad , \tag{3.56}$$

where

S_{d1} = diameter of steel reinforcement in direction 1, and closest to concrete outer tension face (in.) and

S_{p2} = spacing of steel reinforcement in direction 2, perpendicular to direction 1 (in.).

The quantity I_e is given by

$$I_e = \left(\frac{1}{a_u}\right)\left[\left(\frac{M_{cr}}{M}\right)^3 I_g + \left(1 - \left(\frac{M_{cr}}{M}\right)^3\right) I_c\right] , \tag{3.57}$$

where

I_c = cracking moment of inertia in the x- or y-direction (in.^{4}).

The cracking moment of inertia is calculated using the following equation:

$$I_c = a_u\left[0.333\ R_t^3 + \hat{\beta}_1\left(R_t - C_c\right)^2 + \hat{\beta}_2\left(d_t - R_t\right)^2\right] . \tag{3.58}$$

The maximum concrete compressive stress is given by

$$St_m = M\left(\frac{R_t}{I_e}\right) . \tag{3.59}$$

In modeling the water flow characteristics of failed concrete, it is assumed that cracks achieving a depth equal to three-fourths of the remaining slab thickness functionally penetrate the slab. Prior to this, flow through the concrete is the same as that through intact concrete. If the bending moment exceeds the ultimate strength of the concrete slab, cracks penetrate immediately through the slab.

Numerous equations have been proposed for the prediction of crack spacing and width in flexural members. Nawy[14] developed a formula for calculating mean crack spacing for a two-way concrete slab:

$$S_{m1} = 0.5\,K_n \left(\frac{S_{d1} S_{p2}}{Q_d} \right)^{0.5} \quad , \tag{3.60}$$

where

S_{m1} = mean crack spacing in direction 1 (in.).

The variables K_n and Q_d are given by

$$K_n = \left[1.6 + 2.4 \left(\frac{a}{b} - 0.5 \right) \right] 0.29 \tag{3.61}$$

and

$$Q_d = \frac{S_{tn}}{24\,d_c} \quad . \tag{3.62}$$

If the bending moment exceeds the cracking moment, but not the ultimate strength of the concrete member, the mean crack width is given by

$$W_m = S_m \left(\beta_r \frac{St_n}{E_s} + \epsilon_{sh} \right) \quad , \tag{3.63}$$

where

W_m = mean crack width (in.) and
S_m = mean crack spacing (in.).

The quantity β_r is given by

$$\beta_r = \frac{d_{cr}}{d_{cr} - d_c} \quad . \tag{3.64}$$

If the bending moment exceeds the ultimate strength of the concrete member, the crack width is calculated as

$$W_m = S_m \left(\frac{f_y}{E_s} + \epsilon_{sh} \right) \quad . \tag{3.65}$$

If the compressive forces on the wall exceed the ultimate strength of the wall in compression, cracking will occur. The ultimate strength of the wall in compression is calculated as

$$M_{uc} = 0.39\, h_{w1} C_{str} \left[1. - \left(\frac{h_c}{32 h_{w1}} \right)^2 \right] \quad , \tag{3.66}$$

where

M_{uc} = ultimate strength of the wall in compression (lb/in.) and
h_c = height of vault wall (in.).

Cracking because of compression results in a single crack extending through the concrete member; the crack width is one-tenth of the height of the wall section under consideration.

Cracking of a reinforced concrete member may occur also as a result of corrosion of the steel reinforcement. As the concrete surrounding the reinforcement prevents free expansion, the steel-corrosion products will exert pressure within the concrete. Based on elasticity theory,[15] the magnitude of this internal pressure is approximated using

$$P_i = \frac{\Delta}{r_o} \frac{1}{\left[\frac{1-\mu_r}{E_r} + \frac{(1-\mu_c)r_o^2 + (1+\mu_c)d_{cv}^2}{E_c\left(d_{cv} - r_o^2\right)}\right]} \quad , \tag{3.67}$$

where

P_i = internal pressure due to corrosion (lb/in.^{2}),
Δ = thickness of the free expansion layer (in.),
r_o = original radius of steel reinforcement (in.),
μ_r = Poisson's ratio of corrosion product (unitless),
d_{cv} = distance from concrete face to center of steel reinforcement (in.), and
E_r = modulus of elasticity of corrosion product (lb/in.^{2}).

The thickness of the free expansion layer is given by

$$\Delta = r_e + C_{tc} - r_o \quad , \tag{3.68}$$

where

r_e = radius of remaining steel reinforcement (in.) and
C_{tc} = thickness of corrosion layer under conditions of free expansion (in.).

A general series form of the stress function in bipolar coordinates was given by Jeffrey.[16] This function has been applied to the situation of a semi-infinite region with a circular hole under a uniform radius pressure (Fig. 3.2). Based on this work, the stress on the surface of the concrete is given by

$$\sigma_x = -4P_i \frac{r_o^2\left(x^2 - d_{cv}^2 + r_o^2\right)}{\left(x^2 + d_{cv}^2 - r_o^2\right)^2} \quad , \tag{3.69}$$

where

σ_x = stress at surface of concrete (lb/in.^{2}) and

x = distance from point A (Fig. 3.2) along the surface of the concrete (in.).

ORNL-DWG 94-5976

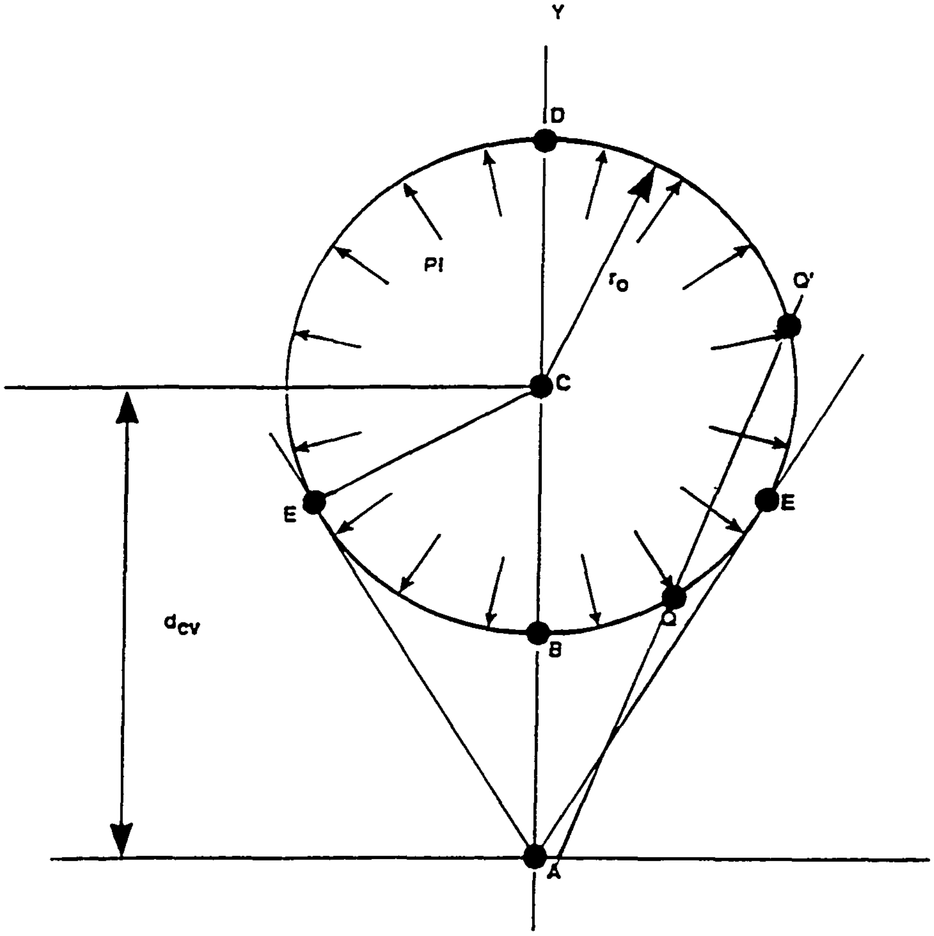

Fig. 3.2. Schematic diagram of steel and concrete geometry used in calculating stress resulting from corrosion of steel reinforcement.

The point of maximum stress occurs at point A (Fig. 3.2), where x = 0. For this case, Eq. (3.69) reduces to

$$\sigma_{xA} = \frac{4P_i r_0^2}{\left(d_{cv}^2 - r_0^2\right)} \quad . \tag{3.70}$$

The tangent stress at any point, Q or Q′, around the circular hole is given by

$$\sigma_{\theta Q} = P_i\left(1 + 2\tan^2\phi\right) \quad , \tag{3.71}$$

where

$\sigma_{\theta Q}$ = tangent stress at point Q (lb/in.^{2}).

This relationship exhibits a maximum tangent stress at point E and is calculated using

$$\sigma_{\theta E} = P_i \frac{d_{cv}^2 + r_o^2}{d_{cv}^2 - r_o^2} \quad , \tag{3.72}$$

where

$\sigma_{\theta E}$ = maximum tangent stress (lb/in.^{2}).

The magnitude of the maximum stress around the circular hole is a function of the ratio of the concrete cover thickness and the radius of the remaining steel reinforcement. The following dependencies are noted:

if $d_{cv} < 1.73r$, $\sigma_{xA} > \sigma_{\theta E} > 2P_i$;
if $d_{cv} = 1.73r$, $\sigma_{xA} = \sigma_{\theta E} = 2P_i$;
if $d_{cv} > 1.73r$, $\sigma_{xA} < \sigma_{\theta E} < 2P_i$.

These relationships provide a simple method for determining where cracking from corrosion will begin.

When the concrete cover thickness is greater than three times the diameter of the reinforcing steel, the tensile stresses around the circular boundary will approach the applied pressure P_i. Plain concrete has minimal tensile strength to resist these stresses—only 6 to 8% of the specified compressive strength of concrete. Consequently, the maximum tension stress can readily exceed the tensile strength of concrete, at which point cracking begins.

Cracking from corrosion is typically initiated internally, along the circular boundary, as the ratio of concrete cover thickness to the original radius of the steel is usually greater than 1.73.[16] As corrosion progresses, accompanied by the deterioration of the concrete cover, cracking will propagate toward the surface of the concrete slab. When the tension stress at the concrete surface equals or exceeds the tensile strength of the concrete, the cracking will penetrate the concrete cover. This cracking will occur along the length of the steel reinforcement.

Spalling out of the concrete will occur if the concrete cover over the steel reinforcement is small ($d_{cv} \leq 1.73r_o$). Under these conditions, the stresses at the concrete surface exceed both the stresses at the steel surface and the tensile strength of concrete and spalling along the reinforcement occurs.

3.2.1.3. Concrete Pad Degradation Model

Tumulus-type disposal facilities may use a steel-reinforced concrete pad upon which disposal vaults are placed. This pad, while intact, should divert water to the leachate collection system. To incorporate the performance of the concrete pad into SOURCE1, a compressive failure model is assumed. Failure is estimated by calculating the reinforcement ratio,[5] which is defined by

$$\rho = \left(\frac{A}{b}\right)\frac{1}{d} \quad , \tag{3.73}$$

where

ρ = reinforcement ratio (dimensionless),

$\left(\frac{A}{b}\right)$ = cross-sectional area of steel reinforcement per unit width of slab (m), and

d = effective depth of steel (distance from the top of the slab to the center of the steel reinforcement) (m).

The reinforcement ratio at which compressive failure may occur is called the limiting reinforcement ratio and is given by[5]

$$\rho_{lim} = \frac{\epsilon_c'}{\epsilon_c' + \epsilon_y} 0.85 \beta_1 \frac{f_c'}{f_y} \quad , \tag{3.74}$$

where

ρ_{lim} = limiting reinforcement ratio (dimensionless),
ϵ_c' = ultimate concrete strain (for this application, taken as 0.003) (dimensionless),
ϵ_y = yield strain of steel (dimensionless),
β_1 = a factor used in the equivalent rectangular stress diagram for concrete at the ultimate load (dimensionless),
f_c' = specified compressive strength of concrete (MPa), and
f_y = specified yield strength of steel reinforcement (MPa).

The yield strain of the steel reinforcement can be calculated by

$$\epsilon_y = \frac{f_y}{E_s} \quad , \tag{3.75}$$

where

E_s = modulus of elasticity of steel reinforcement (for this application, taken as 200,000 MPa) (MPa).

The value of β_1 is determined as follows:[5]

$$\beta_1 = 0.85 \text{ for } f_c' \leq 30 \text{ MPa} \tag{3.76}$$

or

$$\beta_1 = 0.85 - 0.08\left(\frac{f_c^{\prime} - 30}{10}\right) \text{ for } f_c^{\prime} > 30 \text{ MPa} \quad . \tag{3.77}$$

The values of the reinforcement ratio and the limiting reinforcement ratio are evaluated at annual time steps in SOURCE1. These two values are compared, and when the reinforcement ratio exceeds the limiting value, the pad is said to have failed hydraulically. Pad failure allows leachate to pass through it. Values of both ρ and ρ_{lim} will change because of the degradation of the concrete. The concrete is simulated to degrade by using the sulfate attack and calcium hydroxide leaching subroutines in SOURCE1. Sulfate attack results in the spalling off of the concrete cover on the reinforcing steel. Hence, as the effective depth of the steel decreases, the reinforcement ratio increases. Leaching of calcium hydroxide from the concrete pad results in reduced concrete strength. Therefore, as the compressive strength of the concrete decreases, the limiting reinforcement ratio decreases. Both of the concrete degradation mechanisms result in a decrease of the margin between the reinforcement ratio and the limiting reinforcement ratio, ultimately resulting in pad failure.

3.2.1.4 Leachate Collection System Degradation Model

Water that reaches an intact concrete pad of a tumulus-type facility will be diverted to a leachate collection system. This system consists of piping, valves, collection sumps, and monitoring equipment. Ideally, with a properly functioning system, all leachate will be collected, and no release of radionuclides to the environment will occur.

The leachate collection system degradation model describes the functionality fraction of the collection system as a function of time. The functionality fraction is defined as the ratio of the amount of radionuclide in the collected leachate to the total radionuclide release from the disposal vaults and can vary from 0 to 1. With a value of 1, the leachate collection system is fully functional, and no radionuclides are released to the environment. A zero value indicates a fully degraded system that allows all leached radionuclides to be released to the environment.

The initial functionality fraction and the length of the institutional control period are input parameters to the SOURCE1 code. The functionality fraction degrades linearly to zero from the beginning of the simulation until the end of the institutional control period. The degradation of the collection system is assumed to result from piping and valve leaks or failures, flow obstructions within the system, leakage or overflow of collection sumps, degraded monitoring equipment, etc. At the end of the institutional control period, no maintenance of the collection system is assumed to occur. Hence, no credit is taken for the collection system after the end of institutional control. Additionally, if the concrete pad is predicted to fail hydraulically before the end of institutional control, the functionality fraction is set to zero at the time of pad failure.

3.2.2 Disposal Silo and Well Technology

The structural and cracking analyses of the disposal silos, wells, and wells-in-silos are discussed in Sects. 3.2.2.1 through 3.2.2.2. Even though these analyses are similar in many respects to the analyses conducted for the tumulus disposal technology, features unique to these disposal units require additional modeling considerations.

3.2.2.1 Structural Analysis

The structural analysis of the silo and well disposal technologies considers the three structural components of each—loosely referred to as the roof, wall, and floor. These analyses differ depending upon whether the silo or well configuration is being considered. For the analysis of wells-in-silos, structural analyses of both the silo and well are performed.

3.2.2.1.1 Silo or well roof. A polar coordinate system is used in the structural analysis of the roof of the silo or well disposal unit. The roof is modeled as a simply supported circular plate under uniform loading. The load upon the roof is calculated as

$$q_r = h_r \rho_c + s \rho_s \quad , \tag{3.78}$$

where

q_r = uniform load on silo or well roof (lb/in.^{2}).

The final bending moments on the roof caused by the uniform load consist of radial and tangential components, which are given by

$$M_r = \frac{q_r}{16}\left(3 + \mu_c\right)\left(r_s^2 - r^2\right) \tag{3.79}$$

and

$$M_t = \frac{q_r}{16}\left[r_s^2\left(3 + \mu_c\right) - r^2\left(1 + 3\mu_c\right)\right] \quad , \tag{3.80}$$

where

M_r = radial component of bending moment (lb-in./in.),
M_t = tangential component of bending moment (lb-in./in.),
r_s = radius of silo or well (in.), and
r = distance from center of silo or well roof (in.).

These components are summed to arrive at the total bending moment at location (θ, r), namely

$$M_x = M_r\cos^2\theta + M_t\sin^2\theta \tag{3.81}$$

and

$$M_y = M_r\sin^2\theta + M_t\cos^2\theta \quad . \tag{3.82}$$

The shear force on the roof at distance d_t from the interior face of the wall is calculated using

$$Q = \frac{q_r\left(r_s - \frac{h_{wl}}{2} - d_t\right)}{2} \quad , \tag{3.83}$$

where

Q = shear force on roof of well or silo (lb/in.).

3.2.2.1.2 Silo or well wall. The wall of the silo or well is subject to uniform and hydrostatic pressures. Setting the origin of the coordinate system at the mid-height of the wall, the uniform pressure or load is calculated as

$$q_w = \rho_s\left(s + \frac{h_r}{2} + \frac{h_s}{2}\right)\left(1 - \sin f_s\right) - \frac{h_s}{2}\rho_w\left(1 - \sin f_w\right) \quad , \tag{3.84}$$

where

q_w = uniform load on silo or well wall (lb/in.^{2}),

h_s = silo or well height (in.),

f_s = friction angle of soil backfill around silo or well (deg), and

f_w = friction angle of waste inside silo or well (deg).

The corresponding maximum antisymmetrical hydrostatic pressure on the wall is calculated using

$$P = \frac{h_s}{2}\left(\rho_s\left(1 - \sin f_s\right) - \rho_w\left(1 - \sin f_w\right)\right) \quad . \tag{3.85}$$

Bending moments and shear forces due to uniform pressure are calculated for the silo or well wall using

$$M_y = \frac{q_w h_s^2}{4\alpha^2}\left(\frac{\sin\alpha \ \sinh\alpha}{\cos 2\alpha + \cosh 2\alpha}\cos\beta y \ \cosh\beta y - \frac{\cos\alpha \ \cosh\alpha}{\cos 2\alpha + \cosh 2\alpha}\sin\beta y \ \sinh\beta y\right) \tag{3.86}$$

and

$$Q_y = \frac{q_w h_s}{2\alpha}\left[\frac{\sin\alpha\ \sinh\alpha}{\cos2\alpha + \cosh2\alpha}(\cos\beta y\ \sinh\beta y - \sin\beta y\ \cosh\beta y)\right.$$

$$\left. - \frac{\cos\alpha\ \cosh\alpha}{\cos2\alpha + \cosh2\alpha}(\cos\beta y\ \sinh\beta y + \sin\beta y\ \cosh\beta y)\right] \quad , \qquad (3.87)$$

where

M_y = bending moment resulting from uniform loading in the y-direction (lb-in./in.) and

Q_y = shear force resulting from uniform loading in the y-direction (lb/in.).

The quantities α and β are given by

$$\alpha = \frac{\beta h_s}{2} \qquad (3.88)$$

and

$$\beta = \left[\frac{3\left(1-\mu_c^2\right)}{r_s^2 h_{w1}^2}\right]^{0.25} \quad . \qquad (3.89)$$

The silo and well walls are also subject to axial and ring compression forces. The axial compressive force on the silo wall is calculated using Eq. (3.32). This same equation is used for the well-wall calculation by substituting the density of the cast iron for the density of concrete. The ring compression force resulting from a uniform load is calculated as

$$N_\theta = q_w r_s\left[1 - \frac{2\sin\alpha\ \sinh\alpha}{\cos 2\alpha + \cosh 2\alpha}\sin\beta y\ \sinh\beta y - \frac{2\cos\alpha\ \cosh\alpha}{\cos 2\alpha + \cosh 2\alpha}\cos\beta y\ \cosh\beta y\right] , \tag{3.90}$$

where

N_θ = ring compression force (lb/in.^{2}).

The bending moments and shear forces resulting from antisymmetrical hydrostatic pressure are calculated using

$$M_y = \frac{Ph_s^2}{4\alpha^2}\left(\frac{\sin\alpha\cosh\alpha}{\cosh 2\alpha - \cos 2\alpha}\cos\beta y\ \sinh\beta y - \frac{\cos\alpha\ \sinh\alpha}{\cosh 2\alpha - \cos 2\alpha}\sin\beta y\ \cosh\beta y\right) \tag{3.91}$$

and

$$Q_y = \frac{Ph_s}{2\alpha}\left[\frac{\sin\alpha\cosh\alpha}{\cosh 2\alpha - \cos 2\alpha}\left(\cos\beta y\ \cosh\beta y - \sin\beta y\ \sinh\beta y\right) - \frac{\cos\alpha\ \sinh\alpha}{\cosh 2\alpha - \cos 2\alpha}\left(\cos\beta y\ \cosh\beta y + \sin\beta y\ \sinh\beta y\right)\right] . \tag{3.92}$$

The ring compression force caused by hydrostatic pressure is calculated using

$$N_\theta = -2Pr_s\left[\frac{y}{h_s} - \frac{\sin\alpha\ \cosh\alpha}{\cosh 2\alpha - \cos 2\alpha}\sin\beta y\ \cosh\beta y\right.$$

$$\left. - \frac{\cos\alpha\ \sinh\alpha}{\cosh 2\alpha - \cos 2\alpha}\cos\beta y\ \sinh\beta y\right] \quad . \tag{3.93}$$

Bending moments, shear forces, and ring compression forces calculated for the uniform and antisymmetrical hydrostatic pressures are summed at each location on the wall to arrive at the final values.

3.2.2.1.3 Silo or well floor. The circular floor plate is subjected to a distributed line load, or concentrated force, along its perimeter. This concentrated force is calculated as

$$F_c = R_r + h_{w1}h_s\left(\rho_c - \rho_w\right) \quad , \tag{3.94}$$

where

F_c = concentrated force (lb/in.) and

R_r = roof reaction (lb/in.).

The radial and tangential components of the bending moments for the circular floor are given by

$$M_r = -\frac{D}{L_{DK}^2}\left[C_1Z_{2r} - C_2Z_{1r} - \frac{L_{DK}}{r}\left(1-\mu_c\right)\left(C_1Z_{1r}' + C_2Z_{2r}'\right)\right] \tag{3.95}$$

and

$$M_t = -\frac{D}{L_{DK}^2}\left[\mu_c\left(C_1Z_{2r} - C_2Z_{1r}\right) + \frac{L_{DK}}{r}\left(1-\mu_c\right)\left(C_1Z_{1r}' + C_2Z_{2r}'\right)\right] \quad . \tag{3.96}$$

The final bending moments are calculated using Eqs. (3.81) and (3.82).

The maximum shear force on the floor is calculated using

$$Q_{rmax} = -\frac{D}{L_{DK}^3}\left[C_1 Z_{2r}' - C_2 Z_{1r}'\right] \quad , \tag{3.97}$$

where

Q_{rmax} = maximum shear force on floor (lb/in.).

The quantities D, C_1, C_2, and L_{DK} are calculated, respectively, as

$$D = \frac{E_c h_f^3}{12\left(1 - \mu_c^2\right)} \quad , \tag{3.98}$$

$$C_1 = -\frac{F_c}{\tilde{k} L_{DK} \psi}\left[Z_{1r} + \frac{L_{DK}}{r_s}\left(1 - \mu_c\right) Z_{2r}'\right] \quad , \tag{3.99}$$

$$C_2 = -\frac{F_c}{\tilde{k} L_{DK} \psi}\left[Z_{2r} - \frac{L_{DK}}{r_s}\left(1 - \mu_c\right) Z_{1r}'\right] \quad , \tag{3.100}$$

and

$$L_{DK} = \left(\frac{D}{\tilde{k}}\right)^{0.25} \quad . \tag{3.101}$$

The quantities Z_{1r}, Z_{2r}, Z_{1r}', and Z_{2r}' are calculated using the following equations.

$$Z_{1r} = 1 + \sum_{n=1}^{\infty} (-1)^n \frac{\left(\frac{r/L_{DK}}{2}\right)^{4n}}{((2n)!)^2} \quad , \tag{3.102}$$

$$Z_{2r} = \sum_{n=1}^{\infty} (-1)^n \frac{\left(\frac{r/L_{DK}}{2}\right)^{4n-2}}{((2n-1)!)^2} \quad , \tag{3.103}$$

$$Z_{1r}' = \sum_{n=1}^{\infty} (-1)^n \frac{2n\left(\frac{r/L_{DK}}{2}\right)^{4n-1}}{((2n)!)^2} \quad , \tag{3.104}$$

and

$$Z_{2r}' = \sum_{n=1}^{\infty} (-1)^n \frac{(2n-1)\left(\frac{r/L_{DK}}{2}\right)^{4n-3}}{((2n-1)!)^2} \quad , \tag{3.105}$$

where

$n \quad = \quad 1, 2, 3 \ldots$.

The variable ψ, used in Eqs. (3.99) and (3.100), is calculated as

$$\psi = Z_{1r}Z_{2r}' - Z_{1r}'Z_{2r} + \frac{L_{DK}}{r_s}\left(1 - \mu_c\right)\left(Z_{1r}'^2 + Z_{2r}'^2\right) \quad . \tag{3.106}$$

The quantities Z_{1r}, Z_{2r}, Z_{1r}', and Z_{2r}' are calculated using Eqs. (3.102) through (3.105), substituting the silo or well rádius, r_s, for the parameter r.

3.2.2.2 Cracking Analyses

Cracking or failure of the disposal silos, wells, and wells-in-silos occurs at the point at which the structural components can no longer bear the loads placed upon them. Cracking of the roof, wall, and floor of the silo or well may occur as a result of shear forces or bending. Cracking of the wall may occur as a result of compressive forces on the structure.

The cracking analyses for the disposal silos are similar to that performed for tumulus-type facilities in that these analyses model the initiation and propagation of cracks in concrete barriers and calculate fracture characteristics. In contrast, the cracking analyses for the wells simply determine when the roof, wall, or floor undergoes initial failure and does not calculate fracture characteristics.

Shear cracking of a silo or well occurs if the shear force on the structural member exceeds the cracking shear of the member. The cracking shears for the roof and floor in the silo and well are calculated using Eqs. (3.43) and (3.44).

The cracking shear for the silo wall in the vertical direction is the minimum of

$$V_{cr} = h_{wl}\left(1.9C_{str}^{0.5} + 2500S_{tn}Q_x/M_{my}\right) \tag{3.107}$$

and

$$V_{cr} = 3.5C_{str}^{0.5}h_{wl}\left(1. + F_w/\left(500h_{wl}\right)\right)^{0.5} \quad , \tag{3.108}$$

where

M_{my} = modified moment (lb-in.).

The modified moment, M_{my}, is given by

$$M_{my} = M_y - 0.38F_wh_{wl} \quad . \tag{3.109}$$

F_w is determined from Eq. (3.32).

A single fracture will extend through the entire concrete member because of shear cracking. The width of the fracture is 0.013 in. The shear force at which failure of the cast iron wall of the well occurs is given as

$$V_f = 0.7 h_{wl} f_{ws} \quad , \tag{3.110}$$

where

V_f = shear force at which well wall fails (lb/in.) and

f_{ws} = yield strength of cast iron (lb/in.^{2}).

The roof and floor of the silos and wells crack if the bending moment at a given location exceeds the cracking moment. The cracking moment is calculated using Eq. (3.46). Cracks will not extend through the entire member unless the bending moments exceed the ultimate strength of the member. The ultimate flexural strength of the roof and floor is calculated using Eq. (3.49). The ultimate strength for the silo wall is calculated using

$$\begin{aligned} M_u = \phi \Bigg[& S_{tn} f_y \left(d_t - \frac{C_d}{2} \right) + L_{tn} f_y \left(h_{wl} + \frac{L_{tn}}{2} - \frac{C_d}{2} \right) \\ & + L_{cm} f_y \left(\frac{C_d}{2} + \frac{L_{cm}}{2} \right) \Bigg] \quad , \end{aligned} \tag{3.111}$$

where

L_{tn} = thickness of corrugated steel liner on tension face (in.) and

L_{cm} = thickness of corrugated steel liner on compression face (in.).

The depth of the compression block is given by

$$C_d = \frac{f_y \left(S_{tm} + L_{tm} - L_{cm}\right)}{.85\, C_{str}} \quad . \tag{3.112}$$

Fracture depth, spacing, and width are calculated as cracks initiate and propagate in concrete members comprising the silos and wells-in-silos. These characteristics are calculated using the approach discussed for the tumulus-type facility (Sect. 3.2.1.2).

The wall of the silo or well may fail from axial or ring compression. In terms of the former, the silo wall will crack if the axial compression force on the member exceeds the ultimate strength of the wall in compression or critical buckling strength. The strength of the wall in axial compression is calculated as the minimum of

$$N_{ac} = 0.39\, h_{wl} C_{str} \tag{3.113}$$

and

$$N_{ac} = \frac{D_w}{w_f h_s^2} m^2 \pi^2 + E_c h_{wl} \frac{h_s^2}{r_s^2} m^2 \pi^2 \quad , \tag{3.114}$$

where

N_{ac} = ultimate strength or critical buckling strength under axial compression (lb/in.),
D_w = flexural rigidity of wall (lb/in.^{2}), and
m = 1, 2, 3... .

The flexural rigidity of the wall is calculated using Eq. (3.18), substituting the thickness and unit width of the wall for h_r and w_r, respectively.

If the ring compression force on the silo exceeds the ultimate or buckling strength of the wall, cracking will occur. The strength of the wall subject to ring compression is given by the minimum value calculated using

$$N_{rc} = \frac{D_w}{w_f r_s^2}\left(n^2 - 1 + \frac{2n^2 - 1 - \mu_c}{1 + A}\right) + \frac{E h_{wl}}{\left(n^2 - 1\right)\left(1 + A\right)^2} \quad , \tag{3.115}$$

where

N_{rc} = ultimate strength or critical buckling strength under ring compression (lb/in.) and

n = 2, 3, 4... .

The parameter A is calculated using

$$A = \left(\frac{n h_s}{\pi r_s}\right)^2 \quad . \tag{3.116}$$

Compressive forces and bending moments on the wall of the disposal well may result in failure of the well. If the combined stresses on the wall exceed the yield strength of the cast iron, failure will occur. The combined stresses are calculated as

$$N_{sc} = \frac{F_w}{h_{wl}} + \frac{6M_y}{h_{wl}^2} \quad . \tag{3.117}$$

The wall will also fail if the ring compression force on the well wall exceeds the buckling strength of the wall. The ultimate strength of the wall subject to ring compression is the minimum of

$$N_{rc} = 2h_{wl}\left(2 \times 10^6 \frac{h_{wl}}{r_s}\right)\left|\left(1 - 33.3 \frac{h_{wl}}{r_s}\right)\right| \tag{3.118}$$

and

$$N_{rc} = 3 \times 10^4 h_{wl} \quad . \tag{3.119}$$

Cracking of reinforced concrete resulting from the corrosion of the steel reinforcement is modeled using the same methodology developed for the tumulus disposal unit. This portion of the cracking analyses is performed for the roof and floor of the disposal silo only. The walls of the silo and the roof and floor of the well do not contain steel reinforcement.

3.3 FLOW PARTITIONING

A benefit of the concrete engineered barriers used in the tumulus, silo, and well disposal facilities is the material's low hydraulic conductivity. When intact, the concrete largely prevents water from contacting the disposed waste. As the concrete members deteriorate with time, cracks form and greater amounts of water may contact the waste. Eventually, the conductivity of the concrete will be no better than that of the soil backfill around the disposal facility.

To calculate radionuclide releases as a result of advection, it is necessary to estimate the amount of water percolating through the waste. The water entering a disposal area is divided into two components: a component which flows around the disposal facility and a component which contacts the waste.

The flow partitioning scheme used in the SOURCE computer codes is based on the assumption of a saturated steady-state system under a unit hydraulic gradient. Under these conditions, the amount of water percolating through the intact vaults, silos, and wells is equal to the saturated hydraulic conductivity of the concrete. As the concrete members deteriorate and crack, preferential flow of water through the fractures occurs at much greater rates.

Preliminary analyses conducted with the SOURCE computer codes have indicated that much of the ability of a disposal facility to exclude water is lost when fractures penetrate through one or more concrete members. Based on these observations, the amount of water percolating through the waste is set equal to the amount of water entering the disposal area when fractures first penetrate the disposal facility. From this point on, the amount of water contacting the waste is solely a function of the hydraulic characteristics of the site soils and soil backfill.

3.4 RADIONUCLIDE RELEASE MODELING

The SOURCE codes incorporate two mass-transport mechanisms (advection and diffusion) that are modeled in one dimension. The concentration that is calculated to be released by these two mechanisms cannot exceed the solubility limit of the assumed chemical form of a nuclide. Rates of release from disposal facilities which have not undergone significant structural failure will generally be low—that is, below detection limits. These releases are dependent largely on the relative water saturation of the waste and concrete and, for the most part, are the result of diffusion. As a facility deteriorates and undergoes cracking, water may percolate more easily through the waste. Under these conditions, leaching of radionuclides by advection can accelerate and may overshadow leaching by diffusion.

Leaching of radionuclides by advection is directly proportional to the amount of water contacting the waste and inversely proportional to the degree to which radionuclides are sorbed by the waste matrix. An analytical expression in which the radionuclide inventory is updated at preset time-steps is used to evaluate advective leaching. Leaching by diffusion is calculated using the FLOTHRU computer program (a subroutine in the SOURCE codes).[2] A description of these two leaching mechanisms is provided in the following sections and in Appendix A.

3.4.1 Advective Transport Model

The analytical model for advective transport is based on work presented in ref. 3. A detailed derivation of the model can be found in ref. 4.

The total radionuclide release during a time-step is calculated by the following formula:

$$L = \frac{\lambda_L}{\lambda_L + \lambda_d} Q_o \left[e^{-(\lambda_L + \lambda_d)t_1} - e^{-(\lambda_L + \lambda_d)t_2} \right] , \qquad (3.120)$$

where

L = mass of radionuclide leached because of advection (g),

λ_L = leach rate constant (s^{-1}),

λ_d = radioactive decay constant (s^{-1}),

Q_o = initial mass of radionuclide in the waste (g), and

t_1, t_2 = the bounds of the time period of interest (s).

The leach constant, λ_L, is given by

$$\lambda_L = \frac{q}{W\theta R_d} \quad , \tag{3.121}$$

where

q = water infiltration rate (cm/s),
W = waste thickness (cm),
θ = relative saturation (i.e., volume of water in waste/volume of waste) (dimensionless), and
R_d = retardation factor (dimensionless).

Finally, the retardation factor, R_d, can be calculated by the following equation:

$$R_d = 1 + \frac{\rho_b}{\theta} K_d \quad , \tag{3.122}$$

where

ρ_b = bulk density of waste (g/cm^3) and
K_d = distribution coefficient (mL/g).

3.4.2 Diffusion Transport Model

The diffusive transport model that is outlined in this section is a résumé of the work by Nestor presented in ref. 2. Nestor's detailed derivation of this model is presented in Appendix A.

Consider the two-layer slab representation of a waste disposal unit presented in Fig. 3.3. The inner layer, which is of half-thickness a, initially contains a contaminant with concentration C_o. The outer layer, which has thickness b–a, is initially uncontaminated. This situation is analogous to the grouted waste initially placed inside an uncontaminated concrete vault (e.g., tumulus-type disposal). If there is little bulk fluid flow through this system, then diffusion will be the dominant transport mechanism. Diffusion equations can then be written for the inner and outer layers of the disposal unit. The concentration of contaminant in the inner layer is denoted by C_1, while that in the outer layer is denoted by C_2.

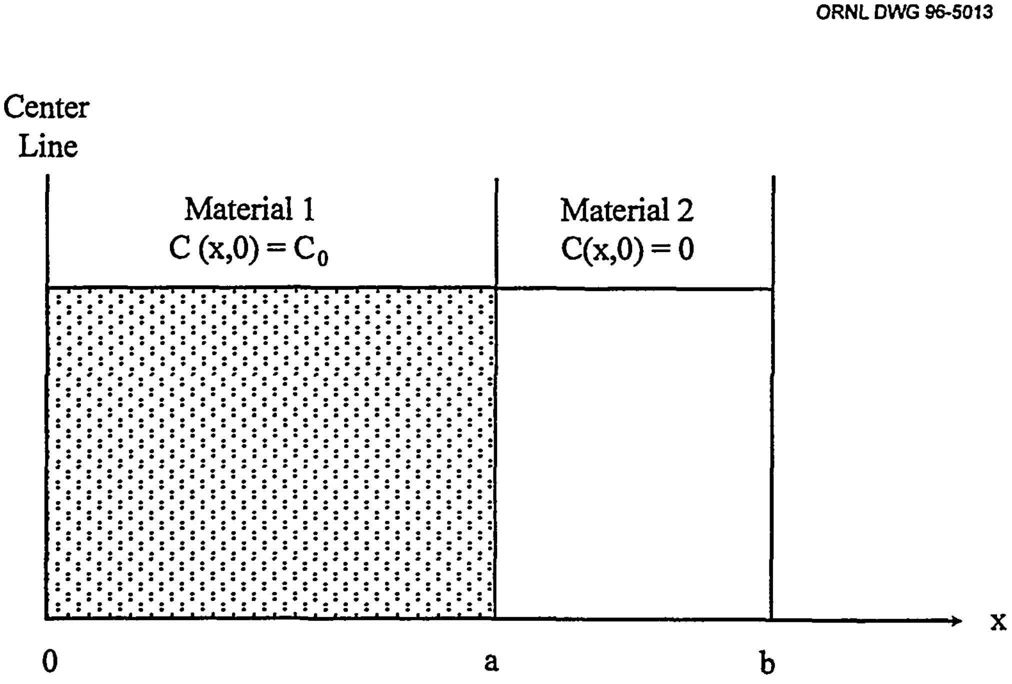

Fig. 3.3. Representation of the system modeled by the FLOTHRU computer code.

The diffusion equation for the inner layer is

$$\frac{\partial C_1}{\partial t} = D_1 \frac{\partial^2 C_1}{\partial x^2} - \lambda_d C_1 \quad , \tag{3.123}$$

where

C_1 = concentration of contaminant in the inner layer (g/cm^3),

D_1 = effective diffusion coefficient for the contaminant in layer 1 (cm^2/s), and

x = spatial position (cm).

Similarly, a diffusion equation can be written for the outer layer:

$$\frac{\partial C_2}{\partial t} = D_2 \frac{\partial^2 C_2}{\partial x^2} - \lambda_d C_2 \quad , \tag{3.124}$$

where

C_2 = concentration of contaminant in the outer layer (g/cm^3) and

D_2 = effective diffusion coefficient for the contaminant in layer 2 (cm^2/s).

Equations (3.123) and (3.124) can be solved with appropriate initial and boundary conditions. In this case, the initial conditions are

$$C_1(x, 0) = C_0 \text{ for } 0 \leq x < a \tag{3.125}$$

and

$$C_2(x, 0) = 0 \text{ for } a \leq x < b \quad . \tag{3.126}$$

The boundary conditions are

$$\left. \frac{\partial C_1}{\partial x} \right|_{x=0} = 0 \quad , \tag{3.127}$$

$$C_2(b,t) = 0 \quad , \tag{3.128}$$

$$C_1(a,t) = C_2(a,t) \quad , \tag{3.129}$$

and

$$D_1 \left.\frac{\partial C_1}{\partial x}\right|_{x=a} = D_2 \left.\frac{\partial C_2}{\partial x}\right|_{x=a} \quad . \tag{3.130}$$

The solution to Eqs. (3.123) through (3.130) is implemented using the FLOTHRU computer code, as described in ref. 2 and Appendix A. This code is incorporated as a subroutine into the SOURCE codes.

3.4.3 Calculation of Total Radionuclide Release

In order to calculate the total amount of radionuclide leaching from a disposal facility using the SOURCE codes, the advective and diffusive components are determined separately. These two components are then added together to calculate the total release. This calculated total is compared with the solubility limit of the radionuclide for the amount of water flowing through the facility. If this limit is exceeded, then the release is limited to the amount determined by solubility.

Radionuclides leached from the waste will be transported away from the disposal facility with the water percolating through the disposal facility. Two flow components are observed on the ORR. A vertical component represents recharge to the underlying aquifer at the site, while a lateral subsurface component discharges to surface waters.

Radionuclide releases from the disposal facilities are partitioned between the recharge and lateral flow components in proportion to the vertical and lateral fluxes. The amount of water which flows vertically to the aquifer is calculated as the minimum of the amount of water percolating through the disposal facility and the saturated hydraulic conductivity of the site soils. Water in excess of the saturated hydraulic conductivity is modeled as lateral sub-surface flow.

Based on the assumption that radionuclide concentrations are equal in each flow component, the amount of material entering the recharge component in a given month is given as

$$Q_r = Q_t \frac{I_r}{I_m} \quad , \tag{3.131}$$

where

Q_r = radionuclide release entering recharge flow component (g/month),

Q_t = total radionuclide release from disposal facility (g/month),

I_r = vertical water percolation rate (cm/month), and

I_m = total water percolation rate (cm/month).

The mass of material entering the lateral flow component is simply the difference of the total release and the mass of material transported to the aquifer. Annual releases for each flow component are calculated by summing the monthly releases.

4. DESCRIPTION OF THE SOURCE CODES INPUT AND OUTPUT FILES

This section provides a description of the input files required to execute the SOURCE codes and the output files created during a simulation. Sample input files and the corresponding output files are found in Appendix C.

4.1 INPUT DATA REQUIREMENTS

The input requirements for the SOURCE codes consist of (1) keyboard input, which provides the name of the primary input file; (2) the primary input file itself, which initializes approximately 115 variables required to conduct a model simulation; and (3) a secondary file, named in the primary file, which contains monthly water infiltration values.

Only the filename of the primary file should be entered at the interactive prompt (e.g., *filename*), and the number of characters should not exceed sixteen. The SOURCE codes will concatenate the extension of *.inp* to the user-supplied *filename* to open *filename*.inp as the primary input file. If the current operating system allows for the redirection of standard input, the keyboard input to the SOURCE codes can be redirected so that the *filename* is input from a file.

The primary input file sets the length of the simulation and print options, provides dimensions and design specifications of the waste disposal facility, establishes physicochemical properties for the facility and groundwater, defines nuclide-specific parameters, initializes a reference year for the simulation, and provides radionuclide inventories disposed of during specified time periods. There are two options for providing radionuclide inventories. If a positive value is input for QCASK on record 23 for SOURCE1 or QSW on record 27 for SOURCE2, this inventory will be used for the simulation, and no additional inventories will be provided. If QCASK is negative or zero, a reference year and a year of disposal with an associated inventory will be read. The variable BGNDUMP on record 25 should equal the year of disposal. Since it is assumed that a pad in a tumulus-type disposal facility will be filled during a year, SOURCE1 does not allow for multiple inventory disposals, but a reference year can be associated with the output summaries of the simulation. If the input value for QSW is negative or zero, a reference year for the simulation and radionuclide inventories disposed of during specified time periods will be read. The reference year for SOURCE1 and SOURCE2 is initialized to the earliest year of a radionuclide disposal at the waste facility. Because disposal times for specific radionuclide inventories may vary, this option provides the capability of ensuring that the simulations of all radionuclide releases at the facility represent the same time period. The output files summarizing the recharge and lateral leach rates will show leach rates of zero until a radionuclide disposal has occurred. For SOURCE2, the time periods for radionuclide disposal can be defined for a year or a range of years. For an interval of 1 year, the inventory in the input file will be added to the current inventory at the appropriate year of the simulation. If a range of years is specified, the corresponding inventory will be added to the disposed inventory during each year of the range. The last time period of disposal should specify an inventory of zero to be disposed of for the remainder of the simulation, and the value of ENDDUMP should be set to 9999999 to terminate the reading of multiple disposals. Table 4.1 provides the input variables and data formats for SOURCE1. For SOURCE2, the input variables and data formats are found in Table 4.2.

The SOURCE1 and the SOURCE2 codes are designed to update the water infiltration values from the monthly infiltration file as changes in the infiltration rates occur at the waste disposal facility. Water infiltration values can be updated annually or held constant for a range of years; the updated values are read when the year of the simulation exceeds the end year of the previous time period. A description of the parameters initialized in the infiltration file is found in Table 4.3.

The SOURCE codes are designed to model leach rates for multiple radionuclides. The number of nuclides which will be simulated is determined by the value of the variable, NONCLD, on record 22 of the SOURCE1 primary input file and record 26 of the SOURCE2 primary input file. The source codes currently allow a maximum number of ten nuclides.

4.2 OPTIONS FOR OUTPUT FILES

The SOURCE1 code has options for generating seven output files, and the SOURCE2 code has options for generating five output files. The two additional SOURCE1 files provide summary information for intact and cracked vaults. The names of the output files are the *filename* of the primary input file concatenated with default extensions set by the SOURCE codes. Table 4.4 shows the file structure of the SOURCE1 code, and Table 4.5 gives the file structure for the SOURCE2 code. The print options for selecting the output files are read on record 2 of the primary input file. A zero or missing value (a blank field) for an option will generate the output file, and the years for printing the simulation results are controlled by the associated frequency option. If a print option is requested and the associated frequency option is not specified, the simulation results will print every year.

The SOURCE codes have the option of providing an input data summary of the waste facility and concrete design parameters, chemical exposure values, and radionuclide-specific parameters in *filename*.con. This same option will create an input data summary of the monthly water infiltration values in *filename*.h2o. If the appropriate print option is selected, *filename*.con will also contain the disposal facility performance summary at selected time intervals, such as rates of concrete degradation and cracking analyses. The annual radionuclide releases into the groundwater recharge, along with the associated water volume, are summarized in *filename*.rch, and the radionuclide releases to the lateral sub-surface flow region and the associated water volume are summarized in *filename*.lat.

For the SOURCE1 code, the leach rate summaries in *filename*.rch, *filename*.lat, and *filename*.sum and the inventory summary in *filename*.sum represent leaching from the total number of vaults at the waste disposal facility. A per-vault summary of the advective leach rate, diffusive leach rate, and the total leach rate is found in *filename*.lch. The two additional output files of the SOURCE1 code provide per-vault summaries for intact and cracked vaults. The inventory and radionuclide release rates by advection and diffusion for intact vaults are summarized in *filename*.vt1. A similar summary is provided for cracked vaults in *filename*.vt2. For the SOURCE2 code, the leach-rate summaries in *filename*.rch, *filename*.lat, *filename*.sum, and *filename*.lch and the inventory summary in *filename*.sum are per silo, well, well-in-silo, or trench.

Table 4.1. Input data format for the SOURCE1 computer code

Column No.	Parameter	Description
		Record 1: Format (A80)
1–80	TITLE	Title of simulation
		Record 2: Format (2I10,I2,7(I2,I5))
1–10	NYEARS	Length of simulation (year)
11–20	INTCTRL	End of institutional control (year)
21–22	IPRINT	Option for input data summary 0 = Input data summary is printed 1 = No input data summary is printed
23–24	IPRN1	Option for recharge summary 0 = Recharge summary is printed 1 = No recharge summary is printed
25–29	IFRQ1	Option for frequency of printing recharge summary (year)
30–31	IPRN2	Option for lateral summary 0 = Lateral summary is printed 1 = No lateral summary is printed
32–36	IFRQ2	Option for frequency of printing lateral summary (year)
37–38	IPRN3	Option for concrete analyses summary 0 = Concrete analyses summary is printed 1 = No concrete analyses summary is printed
39–43	IFRQ3	Option for frequency of printing concrete analyses summary (year)
44–45	IPRN4	Option for inventory, leach rate, and cumulative leached summary 0 = Inventory, leach rate, and cumulative leached summary is printed 1 = No inventory, leach rate, and cumulative leached summary is printed
46–50	IFRQ4	Option for frequency of printing inventory, leach rate, and cumulative leached summary (year)

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 2 (continued)
51–52	IPRN5	Option for advection, diffusion, and leach rate summary 0 = Advection, diffusion, and leach rate summary is printed 1 = No advection, diffusion, and leach rate summary is printed
53–57	IFRQ5	Option for frequency of printing advection, diffusion, and leach rate summary (year)
58–59	IPRN6	Option for inventory, advection, and diffusion for intact vaults summary 0 = Inventory, advection, and diffusion for intact vaults summary is printed 1 = No inventory, advection, and diffusion for intact vaults summary is printed
60–64	IFRQ6	Option for frequency of printing inventory, advection, and diffusion for intact vaults summary (year)
65–66	IPRN7	Option for inventory, advection, and diffusion for cracked vaults summary 0 = Inventory, advection, and diffusion for cracked vaults summary is printed 1 = No inventory, advection, and diffusion for cracked vaults summary is printed
67–71	IFRQ7	Option for frequency of printing inventory, advection, and diffusion for cracked vaults summary (year)
		Record 3: Format (4I5)
1–5	LYR	Number of layers of vaults in tumulus
6–10	NUMWID	Number of vaults along width of tumulus
11–15	NUMLTH	Number of vaults along length of tumulus
16–20	NMEMBER	Number of concrete members to be modeled (= 3 if pad is not modeled, = 4 if pad is modeled)
		Record 4: Format (3E10.3)
1–10	CLWID	Vault width measured from centerline of opposite walls (in.)

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 4 (continued)
11–20	CLLTH	Vault length measured from centerline of opposite walls (in.)
21–30	CLHGHT	Vault height measured from centerline of roof to centerline of floor (in.)
		Record 5: Format (3E10.3)
1–10	CMTHK(1)	Thickness of roof (in.)
11–20	CMTHK(2)	Thickness of wall (in.)
21–30	CMTHK(3)	Thickness of floor (in.)
		Record 6: Format (6E10.3)
1–10	TENCVX(1)	Concrete cover thickness on tension face of concrete roof in x-direction (in.)
11–20	TENCVY(1)	Concrete cover thickness on tension face of concrete roof in y-direction (in.)
21–30	TENCVX(2)	Concrete cover thickness on tension face of concrete wall in x-direction (in.)
31–40	TENCVY(2)	Concrete cover thickness on tension face of concrete wall in y-direction (in.)
41–50	TENCVX(3)	Concrete cover thickness on tension face of concrete floor in x-direction (in.)
51–60	TENCVY(3)	Concrete cover thickness on tension face of concrete floor in y-direction (in.)
		Record 7: Format (6E10.3)
1–10	STLRAD(1)	Radius of steel reinforcement in roof (in.)
11–20	STLSPC(1)	Spacing of steel reinforcement in roof (in.)
21–30	STLRAD(2)	Radius of steel reinforcement in wall (in.)
31–40	STLSPC(2)	Spacing of steel reinforcement in wall (in.)

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 7 (continued)
41–50	STLRAD(3)	Radius of steel reinforcement in floor (in.)
51–60	STLSPC(3)	Spacing of steel reinforcement in floor (in.)
		Record 8: Format (4E10.3)
1–10	SUBMOD	Modulus of elasticity of the subgrade reaction (lb/in.^{3})
11–20	FLANGL	Friction angle of waste/grout in vault (deg)
21–30	SLDNS	Density of soil backfill around tumulus (g/cm^3)
31–40	SLANGL	Friction angle of soil backfill around tumulus (deg)
		Record 9: Format (4E10.3)
1–10	CVRTHK	Thickness of earthen cover (in.)
11–20	CVRDNS	Density of earthen cover (g/cm^3)
21–30	WSTDNS	Density of waste (g/cm^3)
31–40	WSTHT	Relative saturation of waste
		Record 10: Format (7E10.3)
1–10	CCDNS	Density of concrete (g/cm^3)
11–20	CCPOR	Porosity of concrete
21–30	CONPSN	Poisson's ratio for concrete
31–40	COM28D	Compressive strength of concrete at 28 d (lb/in.^{2})
41–50	WCR	Water-cement ratio
51–60	PHBEG	Initial pH of concrete
61–70	WTCMNT	Cement content of concrete (kg/m^3)

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 11: Format (4E10.3)
1–10	CLCON	Concentration of free chloride in concrete (mol/L)
11–20	CCON	Concentration of CaO in concrete (mol/L)
21–30	CFA	Coefficient used in compressive strength function
31–40	CFB	Coefficient used in compressive strength function
		Record 12: Format (3E10.3)
1–10	STLMOD	Modulus of elasticity of steel reinforcement (lb/in.^{2})
11–20	STLYLD	Yield strength of steel reinforcement (lb/in.^{2})
21–30	YNGMOD	Young's modulus of elasticity for concrete (Pa)
		Record 13: Format (3E10.3)
1–10	CACON	Concentration of calcium in C-S-H system (mol/L)
11–20	CAP	Concentration of calcium hydroxide in concrete pore solution (mol/L)
21–30	SI	Concentration of silica in C-S-H system (mol/L)
		Enter Record 14 if NMEMBER = 4
		Record 14: Format (F8.4,8E9.2)
1–8	PSTLRAD	Radius of pad steel reinforcement (in.)
9–17	CMTHK(4)	Concrete pad thickness (in.)
18–26	PSTLMOD	Modulus of elasticity of steel reinforcement (lb/in.^{2})
27–35	PSTLYLD	Yield strength of steel reinforcement (lb/in.^{2})
36–44	PCONSTR	Compressive strength of pad concrete (lb/in.^{2})
45–53	PSTLSPC	Spacing between steel reinforcing rods (in.)

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 14 (continued)
54–62	PBOTCOV	Concrete cover thickness from the center of the bottom row of steel reinforcing rods to the bottom of the pad (in.)
63–71	PWTCMNT	Weight of pad cement per unit volume concrete (kg/m^3)
72–80	PIFF	Pad initial functionality fraction
		Record 15: Format (8E10.3)
1–10	CAGW	Concentration of calcium in groundwater (mol/L)
11–20	CL	Concentration of chloride in groundwater (mol/L)
21–30	CO2	Concentration of carbon dioxide outside tumulus (mol/L)
31–40	CO3	Concentration of carbonate in groundwater (mol/L)
41–50	XMG2	Concentration of magnesium in groundwater (mol/L)
51–60	O2	Concentration of oxygen at tumulus surface (mol/L)
61–70	SO4I	Concentration of sulfate inside vault (mol/L)
71–80	SO4O	Concentration of sulfate outside vault (mol/L)
		Record 16: Format (6E10.3)
1–10	DFALK	Effective diffusivity of alkalis in concrete (m^2/s)
11–20	DFCAOH	Effective diffusivity of calcium hydroxide in concrete (m^2/s)
21–30	DFCL	Effective diffusivity of chloride in concrete (m^2/s)
31–40	DFCO2	Effective diffusivity of carbon dioxide in concrete (m^2/s)
41–50	DFO2	Effective diffusivity of oxygen in concrete (m^2/s)
51–60	DFSO4	Effective diffusivity of sulfate in concrete (m^2/s)
		Record 17: Format (3E10.3)
1–10	PHGW	Groundwater pH

Table 4.1. (continued)

Column No.	Parameter	Description
		Record 17 (continued)
11–20	TDS	Total dissolved solids in groundwater (ppm)
21–30	TEMP	Groundwater temperature (°C)
		Record 18: Format (3E10.3)
1–10	CASOL	Solubility of calcium in groundwater (mol/L)
11–20	CRBSOL	Solubility of carbonate in groundwater (mol/L)
21–30	XMGSOL	Solubility of magnesium in groundwater (mol/L)
		Record 19: Format (4E10.3)
1–10	CFT1	Time at which waste containers begin to corrode (year)
11–20	DCFT	Time required for complete corrosion of waste containers (year)
21–30	EFT1	Time at which epoxy-coating begins to fail (year)
31–40	DEFT	Time required for complete failure of epoxy-coating (year)
		Record 20: Format (4E10.3)
1–10	SITARA	Containment area per unit (m^2)
11–20	SLKR	Saturated hydraulic conductivity of the soil under the tumulus (cm/s)
21–30	SLK	Saturated hydraulic conductivity of soil backfill around tumulus (cm/s)
31–40	CCK	Saturated hydraulic conductivity of concrete (cm/s)
		Record 21: Format (A60)
1–60	WAT_INP	File name containing monthly infiltration values
		Record 22: Format (I5)
1–5	NONCLD	Number of radionuclides considered in the simulation

Table 4.1. (continued)

Column No.	Parameter	Description
Record 23: Format (A8,7E10.3)		
1–8	NUCLID	Radionuclide name
9–18	AM	Atomic mass of radionuclide
19–28	HLIFE	Radionuclide half-life (year)
29–38	SOL	Radionuclide solubility limit (mol/L)
39–48	XKD	Radionuclide distribution coefficient (mL/g)
49–58	QCASK	Radionuclide inventory in vault (g)
59–68	DFWST	Waste diffusion coefficient (m^2/s)
69–78	DFCON	Concrete diffusion coefficient (m^2/s)
Record 24: Format (I10)		
1–10	REFYEAR	Reference year for simulation
Record 25: Format (I10,10E10.3)		
1–10	BGNDUMP	Beginning year for inventory disposal
11–20	QCASK	Inventory in vault (g) for first radionuclide
21–30	QCASK	Inventory in vault (g) for second radionuclide
↓	↓	↓
101–110	QCASK	Inventory in vault (g) for tenth radionuclide

Table 4.2. Input data format for the SOURCE2 computer code

Column No.	Parameter	Description
Record 1: Format (A80)		
1–80	TITLE	Title of simulation
Record 2: Format (I10,I5,I2,5(I2,I5))		
1–10	NYEARS	Length of simulation (year)
11–15	IDFLAG	Disposal unit identification flag 1 = silo 2 = well 3 = well-in-silo
16–17	IPRINT	Option for input data summary 0 = Input data summary is printed 1 = No input data summary is printed
18–19	IPRN1	Option for recharge summary 0 = Recharge summary is printed 1 = No recharge summary is printed
20–24	IFRQ1	Option for frequency of printing recharge summary (year)
25–26	IPRN2	Option for lateral summary 0 = Lateral summary is printed 1 = No lateral summary is printed
27–31	IFRQ2	Option for frequency of printing lateral summary (year)
32–33	IPRN3	Option for concrete analyses summary 0 = Concrete analyses summary is printed 1 = No concrete analyses summary is printed
34–38	IFRQ3	Option for frequency of printing concrete analyses summary (year)
39–40	IPRN4	Option for inventory, leach rate, and cumulative leached summary 0 = Inventory, leach rate, and cumulative leached summary is printed 1 = No inventory, leach rate, and cumulative leached summary is printed
41–45	IFRQ4	Option for frequency of printing inventory, leach rate, and cumulative leached summary (year)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 2 (continued)
46–47	IPRN5	Option for advection, diffusion, and leach rate summary 0 = Advection, diffusion, and leach rate summary is printed 1 = No advection, diffusion, and leach rate summary is printed
48–52	IFRQ5	Option for frequency of printing advection, diffusion, and leach rate summary (year)
		Enter Records 3 through 7 if IDFLAG = 1 or 3
		Record 3: Format (2E10.3)
1–10	SLHGHT	Height of silo measured from centerline of roof to centerline of floor (in.)
11–20	SILRAD	Radius of silo measured to centerline of wall (in.)
		Record 4: Format (3E10.3)
1–10	CMTHK(1,1)	Thickness of silo roof (in.)
11–20	CMTHK(1,2)	Thickness of silo wall (in.)
21–30	CMTHK(1,3)	Thickness of silo floor (in.)
		Record 5: Format (6E10.3)
1–10	TENCVX(1,1)	Concrete cover thickness on tension face of silo roof in x-direction (in.)
11–20	TENCVY(1,1)	Concrete cover thickness on tension face of silo roof in y-direction (in.)
21–30	TENCVX(1,2)	Concrete cover thickness on tension face of silo wall in x-direction (in.)
31–40	TENCVY(1,2)	Concrete cover thickness on tension face of silo wall in y-direction (in.)
41–50	TENCVX(1,3)	Concrete cover thickness on tension face of silo floor in x-direction (in.)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 5 (continued)
51–60	TENCVY(1,3)	Concrete cover thickness on tension face of silo floor in y-direction (in.)
		Record 6: Format (2E10.3)
1–10	STTKCM	Thickness of corrugated steel liner on compression face of silo wall (in.)
11–20	STTKTN	Thickness of corrugated steel liner on tension face of silo wall (in.)
		Record 7: Format (6E10.3)
1–10	STLRAD(1)	Radius of steel reinforcement in silo roof (in.)
11–20	STLSPC(1)	Spacing of steel reinforcement in silo roof (in.)
21–30	STLRAD(2)	Radius of steel reinforcement in silo wall (in.)
31–40	STLSPC(2)	Spacing of steel reinforcement in silo wall (in.)
41–50	STLRAD(3)	Radius of steel reinforcement in silo floor (in.)
51–60	STLSPC(3)	Spacing of steel reinforcement in silo floor (in.)
		Enter Records 8 through 11 if IDFLAG = 2 or 3
		Record 8: Format (2E10.3)
1–10	WLHGHT	Height of well measured from centerline of roof to centerline of floor (in.)
11–20	WLRAD	Radius of well measured to centerline of wall (in.)
		Record 9: Format (3E10.3)
1–10	CMTHK(2,1)	Thickness of well roof (in.)
11–20	CMTHK(2,2)	Thickness of well wall (in.)
21–30	CMTHK(2,3)	Thickness of well floor (in.)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 10: Format (6E10.3)
1–10	TENCVX(2,1)	Concrete cover thickness on tension face of well roof in x-direction (in.)
11–20	TENCVY(2,1)	Concrete cover thickness on tension face of well roof in y-direction (in.)
21–30	TENCVX(2,2)	Concrete cover thickness on tension face of well wall in x-direction (in.)
31–40	TENCVY(2,2)	Concrete cover thickness on tension face of well wall in y-direction (in.)
41–50	TENCVX(2,3)	Concrete cover thickness on tension face of well floor in x-direction (in.)
51–60	TENCVY(2,3)	Concrete cover thickness on tension face of well floor in y-direction (in.)
		Record 11: Format (3E10.3)
1–10	WLSTR	Yield strength of cast iron pipe (lb/in.^{2})
11–20	STLPSN	Poisson's ratio for cast iron
21–30	STLDNS	Density of cast iron used in well (g/cm^3)
		Record 12: Format (4E10.3)
1–10	SUBMOD	Modulus of elasticity of the subgrade reaction (lb/in.^{3})
11–20	FLANGL	Friction angle of waste/grout in silo or well (deg)
21–30	SLDNS	Density of soil backfill around silo or well (g/cm^3)
31–40	SLANGL	Friction angle of soil backfill around silo or well(deg)
		Record 13: Format (4E10.3)
1–10	CVRTHK	Thickness of earthen cover (in.)
11–20	CVRDNS	Density of earthen cover (g/cm^3)
21–30	WSTDNS	Density of waste (g/cm^3)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 13 (continued)
31–40	WSTHT	Relative saturation of waste
		Record 14: Format (7E10.3)
1–10	CCDNS	Density of concrete (g/cm^3)
11–20	CCPOR	Porosity of concrete
21–30	CONPSN	Poisson's ratio for concrete
31–40	COM28D	Compressive strength of concrete at 28 d (lb/in.^{2})
41–50	WCR	Water-cement ratio
51–60	PHBEG	Initial pH of concrete
61–70	WTCMNT	Cement content of concrete (kg/m^3)
		Record 15: Format (4E10.3)
1–10	CLCON	Concentration of free chloride in concrete (mol/L)
11–20	CCON	Concentration of CaO in concrete (mol/L)
21–30	CFA	Coefficient used in compressive strength function
31–40	CFB	Coefficient used in compressive strength function
		Record 16: Format (3E10.3)
1–10	STLMOD	Modulus of elasticity of steel reinforcement (lb/in.^{2})
11–20	STLYLD	Yield strength of steel reinforcement (lb/in.^{2})
21–30	YNGMOD	Young's modulus of elasticity for concrete (Pa)
		Record 17: Format (3E10.3)
1–10	CACON	Concentration of calcium in C-S-H system (mol/L)
11–20	CAP	Concentration of calcium hydroxide in concrete pore solution (mol/L)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 17 (continued)
21–30	SI	Concentration of silica in C-S-H system (mol/L)
		Record 18: Format (8E10.3)
1–10	CAGW	Concentration of calcium in groundwater (mol/L)
11–20	CL	Concentration of chloride in groundwater (mol/L)
21–30	CO2	Concentration of carbon dioxide outside tumulus (mol/L)
31–40	CO3	Concentration of carbonate in groundwater (mol/L)
41–50	XMG2	Concentration of magnesium in groundwater (mol/L)
51–60	O2	Concentration of oxygen at tumulus surface (mol/L)
61–70	SO4I	Concentration of sulfate inside cask (mol/L)
71–80	SO4O	Concentration of sulfate outside cask (mol/L)
		Record 19: Format (6E10.3)
1–10	DFALK	Effective diffusivity of alkalis in concrete (m^2/s)
11–20	DFCAOH	Effective diffusivity of calcium hydroxide in concrete (m^2/s)
21–30	DFCL	Effective diffusivity of chloride in concrete (m^2/s)
31–40	DFCO2	Effective diffusivity of carbon dioxide in concrete (m^2/s)
41–50	DFO2	Effective diffusivity of oxygen in concrete (m^2/s)
51–60	DFSO4	Effective diffusivity of sulfate in concrete (m^2/s)
		Record 20: Format (3E10.3)
1–10	PHGW	Groundwater pH
11–20	TDS	Total dissolved solids in groundwater (ppm)
21–30	TEMP	Groundwater temperature (°C)

Table 4.2. (continued)

Column No.	Parameter	Description
		Record 21: Format (3E10.3)
1–10	CASOL	Solubility of calcium in groundwater (mol/L)
11–20	CRBSOL	Solubility of carbonate in groundwater (mol/L)
21–30	XMGSOL	Solubility of magnesium in groundwater (mol/L)
		Enter Record 22 if IDFLAG = 1 or 3
		Record 22: Format (4E10.3)
1–10	EFT1	Time at which epoxy-coating begins to fail (year)
11–20	DEFT	Time required for complete failure of epoxy-coating (year)
21–30	XLT1	Time at which corrugated steel liners begin to corrode (year)
31–40	DLFT	Time required for complete corrosion of corrugated steel liners (year)
		Enter Record 23 if IDFLAG = 2 or 3
		Record 23: Format (2E10.3)
1–10	WFT1	Time at which cast iron pipe begins to corrode (year)
11–20	DWFT	Time required for complete corrosion of cast iron pipe (year)
		Record 24: Format (4E10.3)
1–10	SITARA	Containment area per unit (m^2)
11–20	SLKR	Saturated hydraulic conductivity of the soil under the waste unit (cm/s)
21–30	SLK	Saturated hydraulic conductivity of soil backfill around the waste unit (cm/s)
31–40	CCK	Saturated hydraulic conductivity of concrete (cm/s)
		Record 25: Format (A60)
1–60	WAT_INP	File name containing monthly infiltration values

Table 4.2. (continued)

Column No.	Parameter	Description
Record 26: Format (I5)		
1–5	NONCLD	Number of radionuclides considered in the simulation
Record 27: Format (A8,7E10.3)		
1–8	NUCLID	Radionuclide name
9–18	AM	Atomic mass of radionuclide
19–28	HLIFE	Radionuclide half-life (year)
29–38	SOL	Radionuclide solubility limit (mol/L)
39–48	XKD	Radionuclide distribution coefficient (mL/g)
49–58	QSW	Radionuclide inventory in silo, well, well-in-silo, or trench (g)
59–68	DFWST	Waste diffusion coefficient (m^2/s)
69–78	DFCON	Concrete diffusion coefficient (m^2/s)
Record 28: Format (I10)		
1–10	REFYEAR	Reference year for simulation
Repeat Record 29 as required to describe the inventory disposal periods		
Record 29: Format (2I10,10E10.3)		
1–10	BGNDUMP	Beginning year for inventory disposal
11–20	ENDDUMP	Ending year for inventory disposal
21–30	QSW	Inventory in silo, well, or well-in-silo (g) for first radionuclide
↓	↓	↓
111–120	QSW	Inventory in silo, well, or well-in-silo (g) for tenth radionuclide

Table 4.3. Input data format for the SOURCE computer codes infiltration values

Column No.	Parameter	Description
Repeat Record 1 as required to describe infiltration scenario for waste unit		
Record 1: Format (2I10,12F5.2)		
1–10	IYR1	Beginning year for using WATER(1) - WATER(12)
11–20	IYR2	Last year for using WATER(1) - WATER(12)
21–25	WATER(1)	Water infiltration rate for first month of year (cm)
26–30	WATER(2)	Water infiltration rate for second month of year (cm)
31–35	WATER(3)	Water infiltration rate for third month of year (cm)
36–40	WATER(4)	Water infiltration rate for fourth month of year (cm)
41–45	WATER(5)	Water infiltration rate for fifth month of year (cm)
46–50	WATER(6)	Water infiltration rate for sixth month of year (cm)
51–55	WATER(7)	Water infiltration rate for seventh month of year (cm)
56–60	WATER(8)	Water infiltration rate for eighth month of year (cm)
61–65	WATER(9)	Water infiltration rate for ninth month of year (cm)
66–70	WATER(10)	Water infiltration rate for tenth month of year (cm)
71–75	WATER(11)	Water infiltration rate for eleventh month of year (cm)
76–80	WATER(12)	Water infiltration rate for twelfth month of year (cm)

Table 4.4. File structure for SOURCE1

Name	Function	Output control variables	Unit
filename.inp	Input: Model parameters		1
Specified in *filename*.inp **Example:** water_turn1.dat	Input: Infiltration values		4
filename.con	Output: Summary of input information and concrete analyses	iprint, iprn3, ifrq3	7
filename.h2o	Output: Beginning year, ending year, monthly infiltration values	iprint	12
filename.rch	Output: Year, flow, recharge	iprn1, ifrq1	2
filename.lat	Output: Year, flow, lateral	iprn2, ifrq2	3
filename.sum	Output: Year, inventory, leach rate, cumulative leached	iprn4, ifrq4	10
filename.lch	Output: Year, advection, diffusion, leach rate	iprn5, ifrq5	11
filename.vt1	Output: Year, inventory, advection, diffusion	iprn6, ifrq6	14
filename.vt2	Output: Year, inventory, advection, diffusion	iprn7, ifrq7	15

Table 4.5. File structure for SOURCE2

Name	Function	Output control variables	Unit
filename.inp	Input: Model parameters		1
Specified in *filename*.inp **Example:** water_th.dat	Input: Infiltration values		4
filename.con	Output: Summary of input information and concrete analyses	iprint, iprn3, ifrq3	7
filename.h2o	Output: Beginning year, ending year, monthly infiltration values	iprint	12
filename.rch	Output: Year, flow, recharge	iprn1, ifrq1	2
filename.lat	Output: Year, flow, lateral	iprn2, ifrq2	3
filename.sum	Output: Year, inventory, leach rate, cumulative leached	iprn4, ifrq4	10
filename.lch	Output: Year, advection, diffusion, leach rate	iprn5, ifrq5	11

5. REFERENCES

1. R. Shuman, N. Chau, and E. A. Jennrich, *The SOURCE Computer Codes: Models for Evaluating the Long-Term Performance of SWSA 6 Disposal Units, Version 1.0: User's Manual*, RAE-9005/8-1, Rogers & Associates Engineering Corporation, Salt Lake City, Utah, April 1992.

2. D. W. Lee et al., *Performance Assessment for Continuing and Future Operations at Solid Waste Storage Area 6*, ORNL-6783, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, Oak Ridge, Tennessee, February 1994.

3. C. F. Baes and R. D. Sharp, " A Proposal for Estimation of Soil Leaching and Leaching Constants for Use in Assessment Models," *J. Environ. Qual.* **12**(1), 17–28 (1983).

4. A. S. Icenhour, *Analysis of Source Term Modeling for Low-Level Radioactive Waste Performance Assessments*, ORNL/TM-12908, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, Oak Ridge, Tennessee, March 1995.

5. H. J. Cowan, *Design of Reinforced Concrete Structures*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

6. A. Atkinson and J. A. Hearne, *An Assessment of the Long-Term Durability of Concrete in Radioactive Waste Repositories*, AERE-R11465, Harwell Laboratory, Oxfordshire, England, October 1984.

7. A. Atkinson and J. A. Hearne, "Mechanistic Model for the Durability of Concrete Barriers Exposed to Sulphate-Bearing Groundwaters," *Mat. Res. Soc. Symp. Proc.* **176**, 149–156 (1990).

8. W. F. Langelier, "The Analytical Control of Anti-Corrosion Water Treatment," *J. Amer. Water Works Assoc.* **28**(10), 1500–1521 (1936).

9. A. Atkinson, *The Time Dependence of pH Within a Repository for Radioactive Waste Disposal*, AERE-R11777, Harwell Laboratory, Oxfordshire, England, April 1985.

10. S. A. Greenberg and T. N. Chang, "Investigation of the Colloidal Hydrated Calcium Silicates. II. Solubility Relationships in the Calcium Oxide-Silica-Water System at 25 D," *J. Phys. Chem.* **69**, 182 (1965).

11. F. M. Lea, *The Chemistry of Cement and Concrete*, 3rd ed., Edward Arnold, Ltd., London, 1970.

12. K. Tuutti, *Corrosion of Steel in Concrete*, Swedish Cement and Concrete Research Institute, Stockholm, Sweden, 1982.

13. D. A. Hausmann, "Steel Corrosion in Concrete," *Materials Protection*, November, 19–23 (1967)

14. E. D. Nawy, "Crack Width Control in Welded Fabric Reinforced Centrally Loaded Two-Way Concrete Slabs," presentation at 62nd ACI Annual Convention in Philadelphia, March 1966.

15. A. S. Saada, *Elasticity Theory and Application*, Pergamon Press, Inc., New York, 1974.

16. G. B. Jeffrey, *Plane Stress and Plane Strain in Bipolar Coordinates*, 1920.

APPENDIX A

THE FLOTHRU COMPUTER PROGRAM

A. THE FLOTHRU COMPUTER PROGRAM

This appendix presents a derivation of the algorithms used in the FLOTHRU computer program, a subroutine in both SOURCE1 and SOURCE2. This derivation was developed by W. Nestor and was originally included in ref. 2. It is included in this appendix for completeness.

The FLOTHRU computer code calculates releases of radionuclides as a result of diffusion. This code represents the diffusion of contamination from grouted waste materials to the outside surface of a disposal facility. The disposal facility is modeled as a two-slab system. The inner slab—representing the grouted waste—is initially uniformly contaminated; the outer slab—representing the concrete components of vaults, silos, or wells—is initially uncontaminated.

Assume that we have a two-layer slab with the inner layer of half-thickness a, initially containing a contaminate with concentration C_o and decay constant λ_d, and with the outer layer of thickness b − a, initially uncontaminated. We will write C_1 for the concentration in the inner layer and C_2 for the concentration in the outer layer.

The diffusion equation for the inner layer is

$$\frac{\partial C_1}{\partial t} = D_1 \frac{\partial^2 C_1}{\partial x^2} - \lambda_d C_1 \quad , \qquad \text{(A.1)}$$

where

C_1 = concentration of contaminant in the inner layer (g/cm^3),
t = time (s),
D_1 = effective diffusion coefficient for the contaminant in layer 1 (cm^2/s),
x = spatial position (cm), and
λ_d = radioactive decay constant (s^{-1}).

Similarly, a diffusion equation can be written for the outer layer:

$$\frac{\partial C_2}{\partial t} = D_2 \frac{\partial^2 C_2}{\partial x^2} - \lambda_d C_2 \quad , \qquad \text{(A.2)}$$

where

C_2 = concentration of contaminant in the outer layer (g/cm^3) and

D_2 = effective diffusion coefficient for the contaminant in layer 2 (cm^2/s).

Equations (A.1) and (A.2) can be solved with appropriate initial and boundary conditions. In this case, the initial conditions are

$$C_1(x,\ 0) = C_0 \text{ for } 0 \leq x < a \tag{A.3}$$

and

$$C_2(x,\ 0) = 0 \text{ for } a \leq x < b \quad . \tag{A.4}$$

The boundary conditions are

$$\left.\frac{\partial C_1}{\partial x}\right|_{x=0} = 0 \quad , \tag{A.5}$$

$$C_2(b,t) = 0 \quad , \tag{A.6}$$

$$C_1(a,t) = C_2(a,t) \quad , \tag{A.7}$$

and

$$D_1 \left.\frac{\partial C_1}{\partial x}\right|_{x=a} = D_2 \left.\frac{\partial C_2}{\partial x}\right|_{x=a} \quad . \tag{A.8}$$

Taking the Laplace transform of the differential equations [i.e., Eqs. (A.1) and (A.2)], we obtain

$$D_1 \frac{d^2\overline{C_1}}{dx^2} - \lambda_d \overline{C_1} = s\overline{C_1} - C_0 \tag{A.9}$$

and

$$D_2 \frac{d^2\overline{C_2}}{dx^2} - \lambda_d \overline{C_2} = s\overline{C_2} \quad . \tag{A.10}$$

Solutions to the ordinary differential equations are

$$\overline{C_1}(x) = A_1(s) \cosh\left(x \sqrt{\frac{s + \lambda_d}{D_1}} \right) + \frac{C_0}{s + \lambda_d} \tag{A.11}$$

and

$$\overline{C_2}(x) = A_2(s) \sinh\left[(b - x) \sqrt{\frac{s + \lambda_d}{D_2}} \right] \quad . \tag{A.12}$$

Note that these solutions satisfy the transformed boundary conditions

$$\left. \frac{d\overline{C_1}}{dx} \right|_{x=0} = 0 \tag{A.13}$$

and

$$\overline{C_2}(b) = 0 \quad . \tag{A.14}$$

Applying the transformed boundary conditions at x = a, we obtain

$$\frac{C_0}{s + \lambda_d} + A_1(s)\cosh\left(a\sqrt{\frac{s + \lambda_d}{D_1}}\right) = A_2(s)\sinh\left[(b - a)\sqrt{\frac{s + \lambda_d}{D_2}}\right] \tag{A.15}$$

and

$$D_1 A_1(s)\sqrt{\frac{s + \lambda_d}{D_1}}\,\sinh\left(a\sqrt{\frac{s + \lambda_d}{D_1}}\right) = -D_2 A_2(s)\sqrt{\frac{s + \lambda_d}{D_2}}\,\cosh\left[(b - a)\sqrt{\frac{s + \lambda_d}{D_2}}\right] \quad . \tag{A.16}$$

Since we wish to find the transform of the release rate at x = b, we solve the second equation for $A_1(s)$ in terms of $A_2(s)$, and substitute the result in the first equation to solve for $A_2(s)$. Let

$$P_1 = \sqrt{\frac{s + \lambda_d}{D_1}} \tag{A.17}$$

and

$$P_2 = \sqrt{\frac{s + \lambda_d}{D_2}} \quad . \tag{A.18}$$

We then obtain

$$A_2(s) = \frac{C_0}{s + \lambda_d} \frac{\sinh\left(ap_1\right)}{\sinh\left[(b - a)p_2\right]\sinh\left(ap_1\right) + \sqrt{D_2/D_1}\cosh\left[(b - a)p_2\right]\cosh\left(ap_1\right)} \quad . \tag{A.19}$$

The transformed release rate at x = b is

$$\bar{q}(s) = -D_2 \frac{d\overline{C_2}}{dx}\bigg|_{x=b} = D_2 \sqrt{\frac{s + \lambda_d}{D_2}}\, A_2(s) \tag{A.20}$$

or

$$\bar{q}(s) = \frac{C_0}{P_2} \frac{\sinh\left(ap_1\right)}{\sinh\left[(b - a)p_2\right]\sinh\left(ap_1\right) + \sqrt{D_2/D_1}\cosh\left[(b - a)p_2\right]\cosh\left(ap_1\right)} \quad , \tag{A.21}$$

so that

$$q(t) = C_0\, e^{-\lambda_d t} g(t) \tag{A.22}$$

with

$$\bar{g}(s) = \frac{\sinh\left(a\sqrt{s/D_1}\right) / \sqrt{s/D_2}}{\sinh\left[(b-a)\sqrt{s/D_2}\right]\sinh\left(a\sqrt{s/D_1}\right) + \kappa\cosh\left[(b-a)\sqrt{s/D_2}\right]\cosh\left(a\sqrt{s/D_1}\right)} \quad , \qquad (A.23)$$

where $\kappa = \sqrt{D_2/D_1}$.

The zeros of the denominator are all on the imaginary axis; thus, we write

$$a\sqrt{s_n/D_1} = ix_n \quad , \qquad (A.24)$$

$$(b-a)\sqrt{s_n/D_1} = i\alpha x_n \quad , \qquad (A.25)$$

so that

$$\alpha = \frac{b-a}{\kappa a} \quad . \qquad (A.26)$$

We then need to solve the transcendental equation

$$f(x) = \kappa\cos(x)\cos(\alpha x) - \sin(x)\sin(\alpha x) = 0 \quad . \qquad (A.27)$$

The derivatives of f (x) are

$$f'(x) = -(\kappa + \alpha)\sin(x)\cos(\alpha x) - (1 + \kappa\alpha)\cos(x)\sin(\alpha x) \quad , \tag{A.28}$$

$$f''(x) = \left(\alpha^2 + 2\alpha\kappa + 1\right)\sin(x)\sin(\alpha x) - \left[\left(\alpha^2 + 1\right)\kappa + 2\alpha\right]\cos(x)\cos(\alpha x) \quad , \tag{A.29}$$

$$f'''(x) = C_1 \cos(x)\sin(\alpha x) + C_2 \sin(x)\cos(\alpha x) \quad , \tag{A.30}$$

where

$$C_1 = \alpha^3\kappa + 3\alpha^2 + 3\alpha\kappa + 1 \tag{A.31}$$

and

$$C_2 = \alpha^3 + 3\alpha^2\kappa + 3\alpha + \kappa \quad . \tag{A.32}$$

We have plotted f(x) for a few values of a, b, D_1, and D_2 in Figs. A.1 through A.3. We note that the first part of the curves is similar to a cosine curve; we approximate

$$f(x) = \kappa\cos(\gamma x) \quad , \tag{A.33}$$

$$f''(x) = -\gamma^2\kappa\cos(\gamma x) \quad , \tag{A.34}$$

and

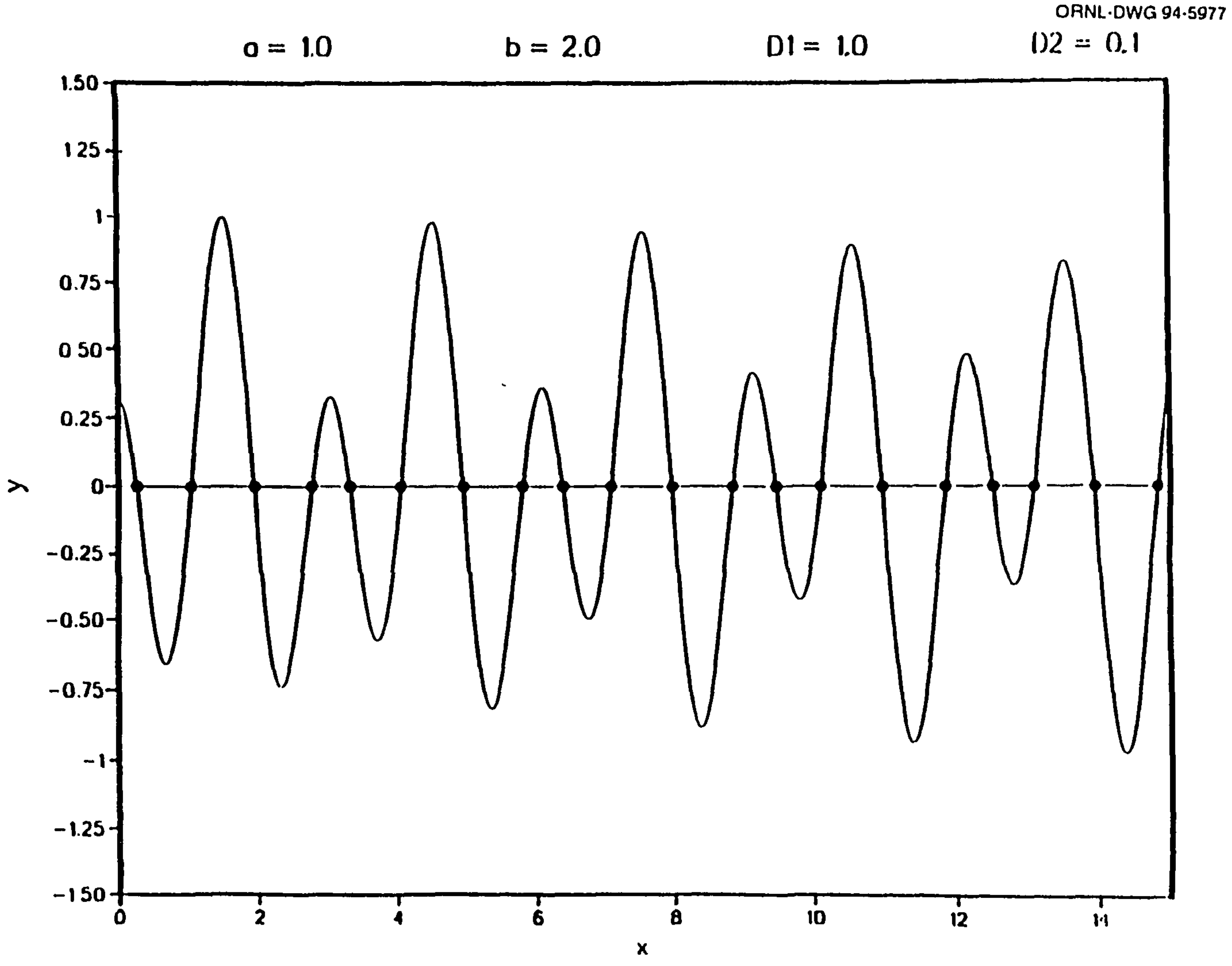

Fig. A.1. Roots of the transcendental equation (dots) for $D_2/D_1 = 0.1$.

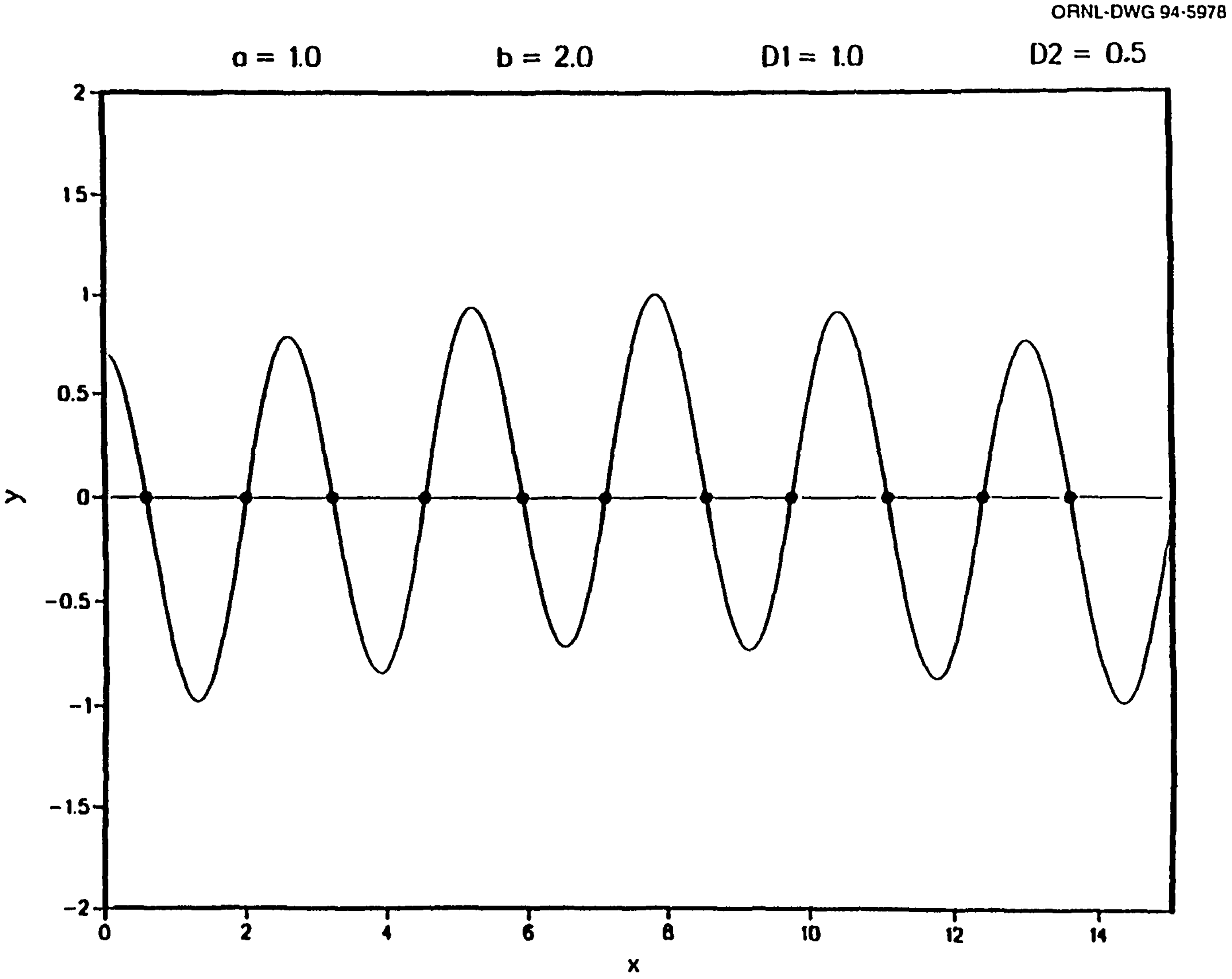

Fig. A.2. Roots of the transcendental equation (dots) for $D_2/D_1 = 0.5$.

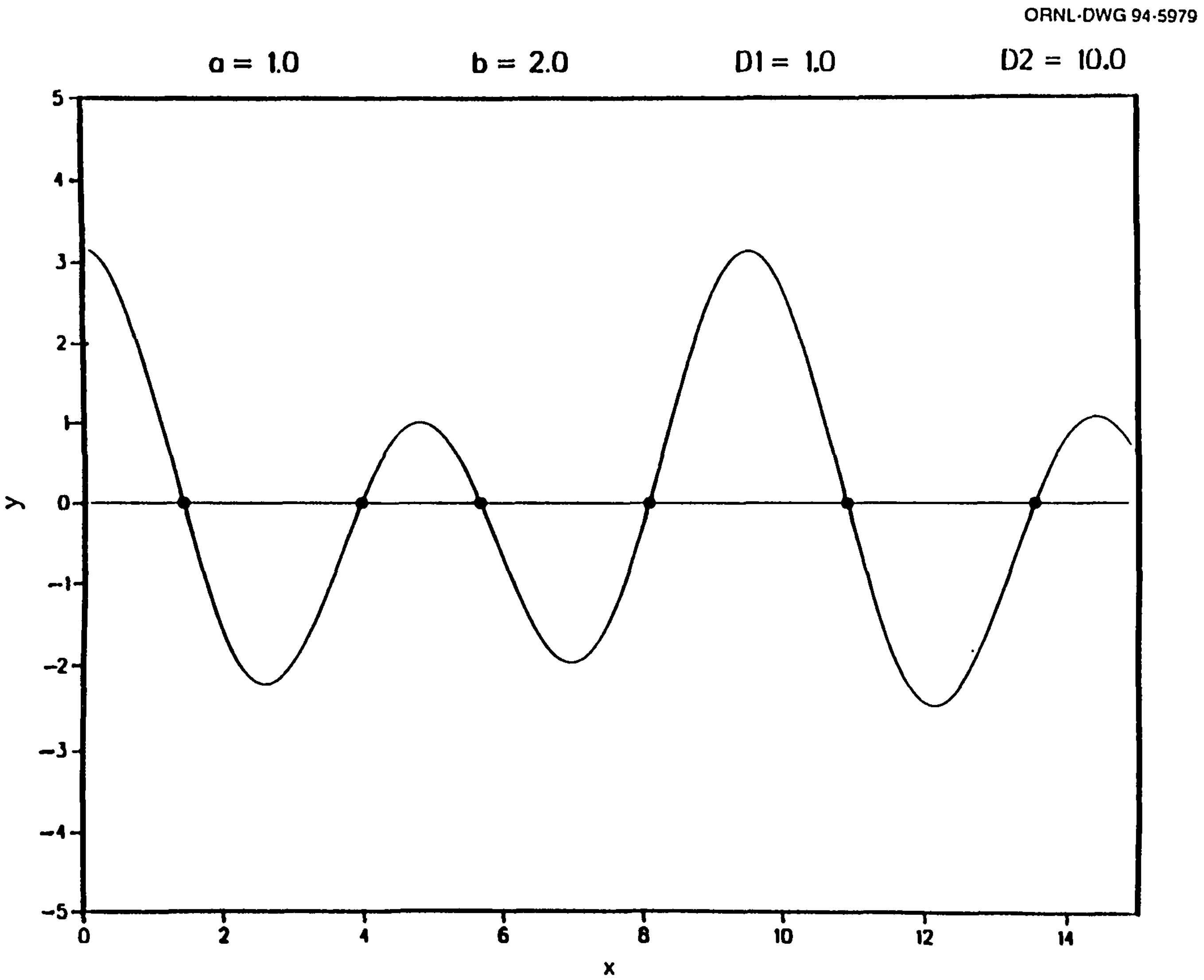

Fig. A.3. Roots of the transcendental equation (dots) for $D_2/D_1 = 10$.

$$\gamma = \sqrt{\frac{-f''(0)}{f(0)}} = \sqrt{\frac{(\alpha^2+1)\kappa + 2\alpha}{\kappa}} \quad , \tag{A.35}$$

and, for a starting estimate for the first root, we use $x_1 = \pi/2\gamma$. We show a few values of the first root and the number of Newton-Raphson iterations needed to reduce the relative error to 5×10^{-9} in Table A.1. Between successive higher roots, the curves resemble sine curves. We approximate

$$f(x) = A \sin(\gamma x) \quad , \tag{A.36}$$

$$f'(x) = \gamma A \cos(\gamma x) \quad , \tag{A.37}$$

and

$$f'''(x) = -\gamma^3 A \cos(\gamma x) \quad . \tag{A.38}$$

Table A.1. First root of $\kappa \cos x \cos \alpha x - \sin x \sin \alpha x$[a]

κ	α							
	0.05		0.10		0.15		0.20	
0.1	1.076583	3	0.859559	4	0.734800	4	0.651694	4
0.2	1.264279	3	1.075710	3	0.949012	4	0.857237	4
0.5	1.428653	3	1.312642	3	1.217087	3	1.137256	3
1.0	1.495997	1	1.427997	1	1.365910	1	1.308997	1
2.0	1.532429	2	1.495596	3	1.459970	3	1.425310	3
5.0	1.555214	3	1.539765	3	1.524287	3	1.508631	3
10.0	1.562966	3	1.555120	3	1.547167	3	1.539016	3

[a] The integer to the right of each root entry is the number of Newton-Raphson iterations required.

If x_n is a root, we evaluate

$$\gamma = \sqrt{\frac{-f'''(x_n)}{f'(x_n)}} \quad , \tag{A.39}$$

and for a starting estimate of the next root, we use

$$x_{n+1} = x_n + \frac{\pi}{\gamma} \quad . \tag{A.40}$$

Typically, three or four Newton-Raphson iterations are sufficient to obtain convergence to a relative error of 10^{-8}.

The function $\bar{g}(s)$ is of the form

$$\bar{g}(s) = \frac{P(s)}{Q(s)} \tag{A.41}$$

with

$$P(s) = \frac{\sinh\left(a\sqrt{s/D_1}\right)}{\sqrt{s/D_2}} \tag{A.42}$$

and

$$Q(s) = \sinh\left[(b-a)\sqrt{s/D_2}\right]\sinh\left(a\sqrt{s/D_1}\right) + \kappa\cosh\left[(b-a)\sqrt{s/D_2}\right]\cosh\left(a\sqrt{s/D_1}\right) \quad . \tag{A.43}$$

The inverse transform is then

$$g(t) = \sum_{n=1}^{\infty} \frac{P(s_n)}{Q'(s_n)} e^{-s_n t} \quad , \tag{A.44}$$

where the s_n are the roots (i.e., x_n) of Q(s). Carrying out the differentiation of Q(s) and substituting the values of s_n, we obtain

$$g(t) = 2D_1\kappa a \sum_{n=1}^{\infty} \frac{\sin(x_n) e^{-D_1 x_n^2 t/a^2}}{\left[\frac{a(b-a)}{\kappa} + \kappa a^2\right] \cos(\alpha x_n) \sin(x_n) + ab \sin(\alpha x_n) \cos(x_n)} \quad . \tag{A.45}$$

The cumulative amount released remaining at time t is

$$R(t) = \int_0^t q(\tau)\, e^{-\lambda_d (t-\tau)} d\tau \quad . \tag{A.46}$$

Since $q(\tau) = C_0 e^{-\lambda_d \tau} g(\tau)$,

$$R(t) = 2\kappa a C_0 e^{-\lambda_d t} \sum_{n=1}^{\infty} \frac{\left(1 - e^{-D_1 x_n^2 t/a^2}\right) \sin(x_n)}{x_n^2 \left[(\alpha + \kappa) \cos(\alpha x_n) \sin(x_n) + \frac{b}{a} \sin(\alpha x_n) \cos(x_n)\right]} \quad . \tag{A.47}$$

When $\lambda_d = 0$, $\lim\limits_{t \to \infty} \frac{R(t)}{aC_0} = 1$, so that

$$\sum_{n=1}^{\infty} \frac{\sin(x_n)}{x_n^2\left[(\alpha + \kappa)\cos(\alpha x_n)\sin(x_n) + \frac{b}{a}\sin(\alpha x_n)\cos(x_n)\right]} = \frac{1}{2\kappa} \quad ; \tag{A.48}$$

$$\frac{R(t)}{aC_0} = e^{-\lambda_d t}\left[1 - 2\kappa \sum_{n=1}^{\infty} \frac{e^{-D_1 x_n^2 t/a^2}\sin(x_n)}{x_n^2\left[(\alpha + \kappa)\cos(\alpha x_n)\sin(x_n) + \frac{b}{a}\sin(\alpha x_n)\cos(x_n)\right]}\right] \quad . \tag{A.49}$$

At small t, Eq. (A.49) has two serious computational defects. The series is slowly convergent, and the sum is nearly equal to 1(2κ), so that serious loss of significant figures will occur in the subtraction. To develop an alternative expression for small time, we note that

$$\overline{R}(s) = \frac{\overline{q}(s)}{s + \lambda_d} \tag{A.50}$$

and

$$R(t) = C_0 e^{-\lambda_d t} h(t) \quad , \tag{A.51}$$

where

$$\bar{h}(s) = \frac{\sqrt{D_2}\,\sinh\left(a\sqrt{s/D_1}\right)}{s^{3/2}\left\{\sinh\left[(b-a)\sqrt{s/D_2}\right]\sinh\left(a\sqrt{s/D_1}\right) + \kappa\cosh\left[(b-a)\sqrt{s/D_2}\right]\cosh\left(a\sqrt{s/D_1}\right)\right\}} \quad . \tag{A.52}$$

If we express all hyperbolic functions in Eq. (A.52) as exponentials and multiply numerator and denominator by

$$\exp\left(-a\sqrt{s/D_1}\right)\exp\left[-(b-a)\sqrt{s/D_2}\right] \quad , \tag{A.53}$$

we obtain, to first order,

$$\bar{h}(s) \approx \frac{2\sqrt{D_2}}{1+\kappa}\,\frac{e^{-(b-a)\sqrt{s/D_2}}}{s^{3/2}} \quad , \tag{A.54}$$

so that

$$\frac{R(t)}{aC_0} \approx \frac{4}{a(1+\kappa)}\sqrt{D_2 t}\;\mathrm{ierfc}\left(\frac{b-a}{2\sqrt{D_2 t}}\right) \quad , \tag{A.55}$$

where ierfc(x) is the integrated complementary error function.[1] To compare the two solutions we show in Table A.2 the first 19 roots of the transcendental equation with the calculated values of f(x) and f′(x); the column headed IT shows the number of Newton-Raphson iterations required. In the footnotes of the table, we have listed the sum of the first 19 terms of the series at t = 0, which should have the value 1(2κ). The time, T, makes the argument of the integrated complementary error function equal to unity.

Table A.2. Comparison of series and alternate solutions[a,b]

n	IT[c]	x_n	$f(x_n)$	$f'(x_n)$
1	2	4.18879×10^{0}	2.47794×10^{-16}	1.12500×10^{0}
2	2	6.98132×10^{0}	-5.54298×10^{-16}	-1.12500×10^{0}
3	2	9.77384×10^{0}	6.49762×10^{-16}	1.12500×10^{0}
4	1	1.25664×10^{1}	2.74089×10^{-9}	-1.12500×10^{0}
5	0	1.53589×10^{1}	-2.74098×10^{-9}	1.12500×10^{0}
6	1	1.81514×10^{1}	2.74088×10^{-9}	-1.12500×10^{0}
7	1	2.09440×10^{1}	-1.21669×10^{-7}	1.12500×10^{0}
8	2	2.37365×10^{1}	1.36664×10^{-16}	-1.12500×10^{0}
9	2	2.65290×10^{1}	1.63999×10^{-16}	1.12500×10^{0}
10	2	2.93215×10^{1}	-4.80844×10^{-17}	-1.12500×10^{0}
11	2	3.21141×10^{1}	3.93766×10^{-15}	1.12500×10^{0}
12	1	3.49066×10^{1}	1.21669×10^{-7}	-1.12500×10^{0}
13	1	3.76991×10^{1}	-2.74040×10^{-9}	1.12500×10^{0}
14	0	4.04916×10^{1}	2.74040×10^{-9}	-1.12500×10^{0}
15	1	4.32842×10^{1}	-2.74088×10^{-9}	1.12500×10^{0}
16	1	4.60767×10^{1}	1.21669×10^{-7}	-1.12500×10^{0}
17	2	4.88692×10^{1}	-2.13521×10^{-15}	1.12500×10^{0}
18	2	5.16617×10^{1}	-2.18182×10^{-15}	-1.12500×10^{0}
19	2	5.44543×10^{1}	-2.33385×10^{-15}	1.12500×10^{0}

[a] $a = 121.92$ cm, $b - a = 15.24$ cm, $D_1 = D_2 = 1.10 \times 10^{-6}$ cm^2/s, $\kappa = 1$, $\alpha = 0.125$.

[b] Sum of series [Eq. (A.48)] at t = 0: 5.00330×10^{-1}, $1/(2\kappa) = 5.00000 \times 10^{-1}$. Sum of series in Eq. (A.49) at t = T: 4.96859×10^{-1} with last term retained: 2.42018×10^{-9}, $T = (b-a)^2/4D_2$. Calculated release (series method) [Eq. (A.49)]: 6.28182×10^{-3}. Calculated release (short-time method) [Eq. (A.55)]: 6.28182×10^{-3}.

[c] IT is the number of Newton-Raphson iterations required for convergence to the root, x_n.

In some cases, we will need more than the first term. Assuming that we can neglect the terms involving D_1, the series is of the form

$$\bar{h}(s) \approx \frac{2\sqrt{D_2}}{(1+\kappa)s^{3/2}} \sum_{n=0}^{\infty} e^{-(2n+1)(b-a)\sqrt{s/D_2}} \left(\frac{1-\kappa}{1+\kappa}\right)^n \quad , \tag{A.56}$$

so that

$$h(t) \approx \frac{4\sqrt{D_2 t}}{(1+\kappa)} \sum_{n=0}^{\infty} \left(\frac{1-\kappa}{1+\kappa}\right)^n \operatorname{ierfc}\left[\frac{(2n+1)(b-a)}{2\sqrt{D_2 t}}\right] \quad . \tag{A.57}$$

Since ierfc(10) $\approx 10^{-44}$, we sum the series until the relative error is 5×10^{-9}, or until the argument of the integrated complementary error function is greater than 10. If more than three terms are required, we use the Aitken delta-squared process to accelerate the convergence.

REFERENCE

1. J. Crank, *The Mathematics of Diffusion*, 2d ed., Oxford University Press, Inc., New York, 1993.

APPENDIX B

GLOSSARY OF SOURCE-CODE PARAMETERS

B. GLOSSARY OF SOURCE-CODE PARAMETERS

The following glossary defines the major terms used in the SOURCE computer codes. Included are variables, arrays and parameter constants as implemented in the main program, functions, and subroutines.

ACOEF	Inner slab layer half-thickness (cm)
ALKLCH	Fraction of alkalis removed from concrete
AM	Radionuclide atomic mass (g/mol)
ANNPRC	Percolation rate in intact concrete (cm/year)
APER	Average fracture aperture in concrete members of silo (cm)
ATTK	Fraction of original concrete member strength
BCOEF	Outer slab layer thickness (cm)
BGNDUMP	Beginning year for inventory disposal
C1	Carbon dioxide concentration at concrete surface (mol/L)
CA	Calcium content in C-S-H system (mol/L)
CACON	Calcium concentration in C-S-H system (mol/L)
CAGW	Calcium concentration in groundwater (mol/L)
CALCH	Fraction of calcium remaining in C-S-H system
CAP	Calcium hydroxide concentration in pore solution of concrete (mol/L)
CA_SI	Calcium-to-silica ratio in C-S-H system
CASOL	Solubility of calcium in groundwater (mol/L)
CCDNS	Density of concrete (g/cm^3)
CCK	Saturated hydraulic conductivity of intact concrete (cm/s)
CCON	Average CaO concentration in concrete (mol/L)
CCPOR	Concrete porosity
CDTSTR	Direct tensile strength of concrete (lb/in.^{2})
CFA	Coefficient used in compressive strength equation
CFB	Coefficient used in compressive strength equation
CFF	Container failure fraction
CFT1	Year in which steel boxes begin to corrode
CL	Chloride concentration in groundwater (mol/L)

CLCON	Concentration of free chloride in concrete (mol/L)
CLHGHT	Vault height (in.)
CLLTH	Vault length (in.)
CLSTL	Chloride concentration at steel reinforcement (mol/L)
CLWID	Vault width (in.)
CMTHK	Concrete member thickness (in.)
CNCFRC	Concentrated force on floor of silo or well (lb-in./in.)
CNCFRX	Concentrated force on vault floor in x-direction (lb/in.)
CNCFRY	Concentrated force on vault floor in y-direction (lb/in.)
CNMNTI	Concrete moment of inertia (in.^{4})
CO2	Environmental concentration of carbon dioxide (mol/L)
CO3	Carbonate concentration in groundwater (mol/L)
COM28D	Compressive strength of concrete at 28 days (lb/in.^{2})
COMCVX	Concrete cover thickness on compression face in x-direction (in.)
COMCVY	Concrete cover thickness on compression face in y-direction (in.)
COMSTR	Time-dependent compressive strength of concrete (lb/in.^{2})
CONMOD	Modulus of elasticity for concrete (lb/in.^{2})
CONPSN	Poisson's ratio for concrete
CONSTR	Concrete strength (lb/in.^{2})
CORVOL	Volume of corrosion product (in.^{3})
CRBCOF	Carbonation coefficient
CRBSOL	Solubility of carbonates in groundwater (mol/L)
CRFRAC	Fraction of concrete member composed of fractures due to corrosion cracking
CRFRCD	Depth of fractures due to corrosion cracking (in.)
CRFRCS	Fracture spacing due to corrosion cracking (in.)
CRFRCW	Width of fractures due to corrosion cracking
CRKMNT	Cracking moment for walls in y-direction (lb-in./in.)
CRKMTF	Cracking moment for floor (lb-in./in.)
CRKMTR	Cracking moment for roof (lb-in./in.)
CRKMTW	Cracking moment for walls in x-direction (lb-in./in.)
CRMTIX	Cracking moment of inertia for x-direction (in.^{4})
CRMTIY	Cracking moment of inertia for y-direction (in.^{4})

CRPCOF	Creep coefficient
CSKSA	Vault surface area (m^2)
CSKVOL	Vault volume (m^3)
CSSTRN	Shrinkage strain of concrete
CSSTRS	Stress in steel reinforcement in compression ($lb/in.^2$)
CTSTRS	Maximum tangent stress on concrete surface ($lb/in.^2$)
CVRDNS	Density of earthen cover (g/cm^3)
CVRTHK	Thickness of earthen cover (in.)
DCFT	Time required for complete corrosion of steel boxes (year)
DECAY	Radioactive decay correction factor
DEFT	Time required for complete failure of epoxy coating (year)
DFALK	Effective diffusivity of alkalis in concrete (m^2/s)
DFCAOH	Effective diffusivity of calcium hydroxide in concrete (m^2/s)
DFCL	Effective diffusivity of chloride ions in concrete (m^2/s)
DFCO2	Effective diffusivity of carbon dioxide in concrete (m^2/s)
DFCON	Effective diffusivity of radionuclide in concrete (m^2/s)
DFO2	Effective diffusivity of oxygen in concrete (m^2/s)
DFSO4	Effective diffusivity of sulfate ions in concrete (m^2/s)
DFWST	Effective diffusivity of radionuclide in waste (m^2/s)
DHYDR	Fraction of hydration of Portland cement
DLFT	Time required for complete corrosion of corrugated steel liners (year)
DPCRB	Depth of carbonation (cm)
DPM	Number of days per month
DWFT	Time required for complete corrosion of well wall (year)
EFT1	Year in which epoxy coating on steel reinforcement begins to fail
EMOLE	Moles of radioactive element available for leaching
ENDDUMP	Ending year for inventory disposal
FCASK	Fraction of vaults that have undergone cracking
FILENAM	Name of input and output files
FLANGL	Friction angle of waste (deg)
FLAPER	Average fracture aperture in floor (cm)

FLDSTX	Distance from steel reinforcement in tension to compression surface of concrete in floor in x-direction (in.)
FLDSTY	Distance from steel reinforcement in tension to compression surface of concrete in floor in y-direction (in.)
FLFDPX	Depth of fractures due to bending in floor x-direction (in.)
FLFDPY	Depth of fractures due to bending in floor y-direction (in.)
FLFRAC	Fraction of floor that consists of fractures
FLFSPX	Spacing of fractures due to bending in floor in x-direction (in.)
FLFSPY	Spacing of fractures due to bending in floor in y-direction (in.)
FLSHR	Maximum shear force in silo floor (lb/in.)
FLUSTX	Ultimate strength of floor in x-direction (lb/in.^{2})
FLUSTY	Ultimate strength of floor in y-direction (lb/in.^{2})
FLWSHR	Maximum shear force in well floor (lb/in.)
FLXMNT	Bending moment for vault or silo floor in x-direction (lb-in./in.)
FLXSHR	Shear force for vault floor in x-direction (lb/in.)
FLYMNT	Bending moment for vault or silo floor in y-direction (lb-in./in.)
FLYSHR	Shear force for vault floor in y-direction (lb/in.)
FNAME	First extension name of input and output files
FRAC	Fraction of concrete member in silo that consists of fractures
FRCWDX	Fracture width of cracks due to bending in the x-direction (in.)
FRCWDY	Fracture width of cracks due to bending in the y-direction (in.)
FWXMNT	Bending moment for well floor in x-direction (lb-in./in.)
FWYMNT	Bending moment for well floor in y-direction (lb-in./in.)
HLIFE	Radionuclide half-life (year)
HYDRLD	Hydrostatic pressure on wall of disposal unit (lb/in.^{2})
ICL	Corrosion initiation time for concrete member due to chloride penetration (year)
ICO2	Corrosion initiation time for concrete member due to carbonation (year)
ICRACK	Flag indicating that concrete member has cracked
ICRFLG	Flag indicating steel reinforcement depassivation
IDFLAG	Disposal unit identification flag
IFAIL	Flag indicating that structural member of well has failed
IFLAGS	Flag indicating that solubility constraints were exceeded in silo or well

IFLAGS1	Flag indicating that solubility constraints were exceeded in intact vaults
IFLAGS2	Flag indicating that solubility constraints were exceeded in cracked vaults
IFRQ1	Option for frequency of printing recharge summary (year)
IFRQ2	Option for frequency of printing lateral summary (year)
IFRQ3	Option for frequency of printing concrete analyses summary (year)
IFRQ4	Option for frequency of printing inventory, leach rate, and cumulative leached summary (year)
IFRQ5	Option for frequency of printing advection, diffusion, and leach rate summary (year)
IFRQ6	Option for frequency of printing inventory, advection, and diffusion for intact vaults summary (year)
IFRQ7	Option for frequency of printing inventory, advection, and diffusion for cracked vaults summary (year)
INTCTRL	Year institutional control period ends
IPRINT	Option to print input data summary
IPRN1	Option to print recharge summary
IPRN2	Option to print lateral summary
IPRN3	Option to print concrete analyses summary
IPRN4	Option to print inventory, leach rate, and cumulative leached summary
IPRN5	Option to print advection, diffusion, and leach rate summary
IPRN6	Option to print inventory, advection, and diffusion for intact vaults summary
IPRN7	Option to print inventory, advection, and diffusion for cracked vaults summary
ISAVE	Flag indicating number of concrete members of vaults which have cracked
ISAVE1	Flag indicating number of concrete members of silo which have cracked
ISAVE2	Flag indicating number of structural members of well which have failed
ISPL	Flag indicating concrete member has spalled due to corrosion
LYR	Number of layers of vaults in tumulus
MAXNUC	Maximum number of nuclides that can be considered
MAXYR	Maximum number of years that can be simulated
NCCASK	Number of vaults which have undergone cracking
NMEMBER	Number of structural members to be considered
NOCLX	Scale factor for calculating floor overhang in x-direction
NOCLY	Scale factor for calculating floor overhang in y-direction

NONCLD	Number of radionuclides considered in simulation
NUCLID	Radionuclide name
NUMCSK	Number of vaults in tumulus
NUMLTH	Number of vaults along length of tumulus
NUMWID	Number of vaults along width of tumulus
NYEARS	Length of simulation (year)
O2	Oxygen concentration in groundwater (mol/L)
O2FLUX	Oxygen flux at steel reinforcement (mol/year)
O2GRD	Oxygen concentration gradient across concrete cover (mol/m^4)
OH	Hydroxide ion concentration in concrete pore solution (mol/L)
OMMTHK	Initial concrete member thickness (in.)
OSTLRD	Initial steel reinforcement radius (in.)
OSTTKC	Initial thickness of corrugated steel liner on compression face of silo wall (in.)
OSTTKT	Initial thickness of corrugated steel liner on tension face of silo wall (in.)
OTNCVX	Initial concrete cover on tension face in x-direction (in.)
OTNCVY	Initial concrete cover on tension face in y-direction (in.)
OVRHNG	Floor overhang measured from centerline of wall (in.)
PADCRK	Flag indicating concrete pad has failed
PBOTCOV	Concrete cover thickness from the center of the bottom row of steel reinforcing rods to the bottom of the pad (in.)
PCONSTR	Compressive strength of pad concrete (lb/in.^{2})
PH	Concrete pH
PHBEG	Initial concrete pH
PHGW	Groundwater pH
PHS	Groundwater pH at calcium carbonate saturation
PIFF	Pad initial functionality fraction (unitless)
PIPSHR	Allowable shear in well wall (lb/in.^{2})
PSTL	Internal pressure on concrete due to corrosion (lb/in.^{2})
PSTLMOD	Modulus of elasticity of steel reinforcement (lb/in.^{2})
PSTLRAD	Radius of pad steel reinforcement (in.)
PSTLSPC	Spacing between steel reinforcing rods in pad (in.)
PSTLYLD	Yield strength of steel reinforcement (lb/in.^{2})

PUSRC	Buckling strength of well wall (lb/in.)
PUSTR	Ultimate strength of well wall (lb/in.^{2})
PWTCMNT	Weight of pad cement per unit volume concrete (kg/m^3)
Q	Radionuclide inventory available for leaching (g)
QCASK	Initial radionuclide inventory (g/vault)
QCASK1	Radionuclide inventory in intact vaults (g/intact vault)
QCASK2	Radionuclide inventory in cracked vaults (g/cracked vault)
QK1	Radionuclide inventory available for leaching from intact vaults (g)
QK2	*Radionuclide inventory available for leaching from cracked vaults (g)*
QSW	Radionuclide inventory in silo or well (g)
RATMOD	Ratio of modulus of elasticity of steel and modulus of elasticity of concrete
REFYEAR	Reference year for beginning the simulation
REL	Monthly leach rate due to diffusion (g/month)
RFAPER	Average fracture aperture in roof (cm)
RFDSTX	Distance from steel reinforcement in tension to compression surface of concrete in roof in x-direction (in.)
RFDSTY	Distance from steel reinforcement in tension to compression surface of concrete in roof in y-direction (in.)
RFFDPX	Depth of fractures due to bending in roof x-direction (in.)
RFFDPY	Depth of fractures due to bending in roof y-direction (in.)
RFFRAC	Fraction of roof that consists of fractures
RFFSPX	Spacing of fractures due to bending in roof in x-direction (in.)
RFFSPY	Spacing of fractures due to bending in roof in y-direction (in.)
RFSHR	Maximum shear force in silo roof (lb/in.)
RFUSTX	Ultimate strength of roof in x-direction (lb/in.^{2})
RFUSTY	Ultimate strength of roof in y-direction (lb/in.^{2})
RFWSHR	Maximum shear force in well roof (lb/in.)
RFXMNT	Bending moment for vault or silo roof in x-direction (lb-in./in.)
RFXRXN	Roof reaction in x-direction (lb/in.)
RFXSHR	Shear force for vault roof in x-direction (lb/in.)
RFYMNT	Bending moment for vault or silo roof in y-direction (lb-in./in.)
RFYRXN	Roof reaction in y-direction (lb/in.)

RFYSHR	Shear force for vault roof in y-direction (lb/in.)
RLCH	Radionuclide release to recharge component (g)
RUPMOD	Modulus of rupture of concrete (lb/in.^{2})
RWXMNT	bending moment for well roof in x-direction (lb-in./in.)
RWYMNT	Bending moment for well roof in y-direction (lb-in./in.)
SFRCDX	Depth of fractures due to shear cracking in x-direction (in.)
SFRCDY	Depth of fractures due to shear cracking in y-direction (in.)
SFRCSX	Spacing of fractures due to shear cracking in x-direction (in.)
SFRCSY	Spacing of fractures due to shear cracking in y-direction (in.)
SFRCWX	Width of fractures due to shear cracking in x-direction (in.)
SFRCWY	Width of fractures due to shear cracking in y-direction (in.)
SHRSTR	Shear strength (lb/in.^{2})
SI	Silica concentration in C-S-H system (mol/L)
SILRAD	Radius of silo (in.)
SITARA	Disposal facility drainage area (m^2)
SLANGL	Friction angle of soil backfill around disposal facility (deg)
SLDNS	Density of soil backfill around disposal facility (g/cm^3)
SLFI	Concrete loss from inside of disposal facility due to sulfate attack (cm)
SLFO	Concrete loss from outside of disposal facility due to sulfate attack (cm)
SLHGHT	Height of silo (in.)
SLK	Saturated hydraulic conductivity of soil backfill around disposal facility (cm/s)
SLKR	Saturated hydraulic conductivity of the soil under the disposal facility (cm/s)
SO4I	Sulfate ion concentration in groundwater inside disposal facility (mol/L)
SO4O	Sulfate ion concentration in groundwater outside disposal facility (mol/L)
SOL	Radionuclide solubility (mol/L)
SSTRED	Strength reduction factor for silo wall
STARCM	Area of steel reinforcement in compression (in.^{2})
STARTN	Area of steel reinforcement in tension (in.^{2}/in.)
STLARA	Area of steel reinforcement subject to corrosive attack (m^2)
STLCOR	Thickness of corrosion layer around steel reinforcement (in.)
STLDNS	Density of cast iron used in well construction (g/cm^3)
STLMOD	Modulus of elasticity of steel reinforcement (lb/in.^{2})

STLPSN	Poisson's ratio for cast iron
STLRAD	Radius of steel reinforcement (in.)
STLSPC	Spacing of steel reinforcement (in.)
STLYLD	Yield strength of steel reinforcement (lb/in.^{2})
STRRED	Strength reduction factor
STSTRS	Tangent stress upon concrete at surface of steel reinforcement (lb/in.^{2})
STTKCM	Thickness of corrugated steel liner on compression face of silo wall (in.)
STTKTN	Thickness of corrugated steel liner on tension face of silo wall (in.)
SUBMOD	Modulus of the subgrade reaction (lb/in.^{3})
TDS	Total dissolved solids in groundwater (ppm)
TEMP	Groundwater temperature (°C)
TENCVX	Concrete cover thickness on tension face in x-direction (in.)
TENCVY	Concrete cover thickness on tension face in y-direction (in.)
TITLE	Title of simulation
TLEACH1	Leach rate from intact vaults (g/year)
TLEACH2	Leach rate from cracked vaults (g/year)
TTLWAT	Volume of water percolating through the disposal facility (m^3)
UNFLD	Uniform load on wall of disposal facility (lb/in.^{2})
VCR	Cracking shear (lb/in.^{2})
VOLFE	Volume of iron (in.^{3})
W1APER	Average fracture aperture in vault wall 1 (cm)
W1CMFY	Compressive force on vault wall 1 in y-direction (lb/in.)
W1FDPX	Wall 1 crack depth in x-direction (in.)
W1FDPY	Wall 1 crack depth in y-direction (in.)
W1FRAC	Fraction of vault wall 1 that consists of fractures
W1FSPX	Wall 1 crack spacing in x-direction (in.)
W1FSPY	Wall 1 crack spacing in y-direction (in.)
W1XMNT	Bending moment for vault wall 1 in x-direction (lb-in./in.)
W1XSHR	Shear force for vault wall 1 in x-direction (lb/in.)
W1YMNT	Bending moment for vault wall 1 in y-direction (lb-in./in.)
W1YSHR	Shear force for vault wall 1 in y-direction (lb/in.)
W2APER	Average fracture aperture in vault wall 2 (cm)

W2CMFY	Compressive force on vault wall 2 in y-direction (lb/in.)
W2FDPX	Wall 2 crack depth in x-direction (in.)
W2FDPY	Wall 2 crack depth in y-direction (in.)
W2FRAC	Fraction of vault wall 2 that consists of fractures
W2FSPX	Wall 2 crack spacing in x-direction (in.)
W2FSPY	Wall 2 crack spacing in y-direction (in.)
W2XMNT	Bending moment for vault wall 2 in x-direction (lb-in./in.)
W2XSHR	Shear force for vault wall 2 in x-direction (lb/in.)
W2YMNT	Bending moment for vault wall 2 in y-direction (lb-in./in.)
W2YSHR	Shear force for vault wall 2 in y-direction (lb/in.)
WATER	Infiltration rate of water into the disposal facility (cm/month)
WAT_INP	File name containing monthly infiltration values
WCR	Water-cement ratio of concrete
WFT1	Year in which corrosion of well wall begins
WLCMFR	Silo compressive forces due to the roof reaction and weight of the walls (lb/in.)
WLDSTX	Distance from steel reinforcement in tension to compression surface of concrete in walls in x-direction (in.)
WLDSTY	Distance from steel reinforcement in tension to compression surface of concrete in walls in y-direction (in.)
WLFDPX	Depth of fractures due to bending in wall in x-direction (in.)
WLFDPY	Depth of fractures due to bending in wall in y-direction (in.)
WLFSPX	Spacing of fractures due to bending in x-direction (in.)
WLFSPY	Spacing of fractures due to bending in y-direction (in.)
WLHGHT	Well height (in.)
WLRAD	Radius of well (in.)
WLSTR	Yield strength of steel in well wall (lb/in.^{2})
WLUSTX	Ultimate strength of wall in x-direction (lb/in.^{2})
WLUSTY	Ultimate strength of wall in y-direction (lb/in.^{2})
WLWXRC	Well ring compression force due to a uniform load (lb/in.)
WLXRC	Silo ring compression force due to hydrostatic pressure (lb/in.)
WLYMNT	Bending moment for silo wall in y-direction (lb-in./in.)
WLYSHR	Shear force in silo wall (lb/in.)

WSTDNS	Density of waste (g/cm^3)
WSTHK	Thickness of waste in disposal facility (cm)
WSTHT	Relative saturation of waste
WSTRED	Strength reduction factor for well
WTCMNT	Cement content of concrete (kg/m^3)
WWCMFR	Well compressive forces due to the roof reaction and weight of the walls (lb/in.)
WWYMNT	Bending moment for well wall in y-direction (lb-in./in.)
WWYSHR	Shear force in well wall (lb-in./in.)
XIM	Trigonometric function used in floor structural analysis
XIIM	Trigonometric function used in floor structural analysis
XIP	Trigonometric function used in floor structural analysis
XIIP	Trigonometric function used in floor structural analysis
XKD	Radionuclide distribution coefficient in waste (mL/g)
XLEACH	Total radionuclide release (g/year)
XLFF	Corrugated steel liner failure fraction
XLFT1	Year in which corrugated steel liners begin to corrode
XLI	Langelier index
XLLCH	Annual radionuclide release to lateral flow component (g)
XLOAD	Uniform load on roof of disposal facility ($lb/in.^2$)
XMG2	Magnesium ion concentration in groundwater (mol/L)
XMGSOL	Solubility of magnesium in groundwater (mol/L)
XMOLE	Moles of iron in steel reinforcement
XPERC	Percolation rate through intact and cracked concrete (cm/month)
YIM	Trigonometric function used in floor structural analysis
YIIM	Ttrigonometric function used in floor structural analysis
YIP	Trigonometric function used in floor structural analysis
YIIP	Trigonometric function used in floor structural analysis
YNGMOD	Young's modulus (Pa)

APPENDIX C

SAMPLE INPUT AND OUTPUT FILES FOR THE SOURCE1 AND SOURCE2 COMPUTER CODES

C. SAMPLE INPUT AND OUTPUT FILES FOR THE SOURCE1 AND SOURCE2 COMPUTER CODES

Sample input and output data files for SOURCE1 and SOURCE2 are presented in Exhibits C.1 through C.21. With the exception of the radionuclide inventories, values used in the input files were taken from ref. 2. A unit inventory was used in each of the input files, except for the ^{238}U file, which has a higher inventory to demonstrate solubility-limited releases. Note that in the caption of each exhibit, the filename extension associated with the file is indicated in parentheses. For purposes of these examples, a generic filename (e.g., *filename*.con) is used to demonstrate the file-naming convention. Output files that correspond to the input files are provided for one of the SOURCE1 samples and one of the SOURCE2 samples. Similar output files would be obtained for the other sample input files. The output frequency was selected to conserve space. Format requirements for the input files are presented in Tables 4.1 through 4.3 of Sect. 4. A summary of information presented in each of the output files is presented in Tables 4.4 and 4.5 in Sect. 4.

Exhibit C.1 provides a sample input file for ^{137}C in a tumulus-type facility. Note that this file requires the water infiltration input file *water.dat*. A generic example of a water infiltration input file is presented as Exhibit C.2. To use this file with the sample in Exhibit C.1, the water infiltration file should be named *water.dat*. Associated output files for the input files shown in Exhibits C.1 and C.2 are provided in Exhibits C.3 through C.10. For a performance assessment, the recharge and lateral-release output files (Exhibits C.5 and C.6, respectively) would be used in subsequent radionuclide transport calculations and probably would be written at an annual frequency. Exhibit C.11 is also an input file for ^{137}Cs in a tumulus-type disposal facility. This file differs from Exhibit C.1 in that there is no concrete pad.

Exhibit C.12 is a sample input file for ^{238}U in a silo-type facility. This file demonstrates the use of a reference year and radionuclide disposal over a range of years. Additionally, the radionuclide inventory was selected to demonstrate solubility-limited releases. Output files that correspond to a SOURCE2 simulation for ^{238}U in a silo (using Exhibits C.2 and C.12) are provided in Exhibits C.13 through C.18.

Additional sample SOURCE2 input files are presented in Exhibits C.19 through C.21. Exhibit C.19 is for ^{137}Cs in a well-type facility. Exhibit C.20 is for ^{137}Cs in a well-in-silo-type facility. Finally, Exhibit C.21 is for ^{137}Cs in a trench-type facility. The trench is modeled as a silo without barriers by using an equivalent surface-to-volume ratio. Concrete parameters are selected to result in silo (i.e., the equivalent of a trench) failure during the first year of the simulation. The trench then effectively has no engineered barriers.

Exhibit C.1. Sample SOURCE1 input file for ^{137}Cs in a tumulus-type facility with a concrete pad (*filename*.inp)

```
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
       300       100 0 0   10 0   10 0  150 0   10 0   10 0   10 0   10
    3   10   11     4
 6.000E+01 8.700E+01 6.500E+01
 7.000E+00 7.000E+00 7.000E+00
 3.060E+00 3.690E+00 3.250E+00 3.750E+00 2.000E+00 2.500E+00
 3.125E-01 1.000E+01 2.500E-01 1.200E+01 2.500E-01 1.200E+01
 2.000E+03 4.000E+01 1.760E+00 3.000E+01
 7.200E+01 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.250E+01 4.030E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 2.900E+07 6.000E+04 2.000E+10
 1.750E+00 2.000E-02 7.100E-01
  0.4375 1.50E+01 2.90E+07 6.00E+04 4.00E+03 6.00E+00 6.00E+00 3.30E+02 8.00E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 3.440E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 6.000E+01 0.000E+00 2.000E+01
 4.990E+02 5.800E-07 3.500E-03 1.000E-10
water.dat
    1
Cs-137   1.370E+02 3.000E+01 1.600E+01 1.990E+01 1.000E+00 6.800E-12 5.120E-13
```

Exhibit C.2. Sample water infiltration input data (the name for this file is specified in *filename*.inp)

```
 1        2212.4212.4211.6810.9215.75 8.4823.44 6.20 8.00 6.38 5.5415.98
23          32 0.03 0.38 1.12 1.12 0.77 0.34 0.16 0.08 0.04 0.02 0.01 0.01
33          33 0.91 1.24 2.04 1.71 1.83 0.60 0.29 0.23 0.19 0.16 0.13 0.09
34          34 1.80 2.10 2.96 2.30 2.89 0.86 0.42 0.38 0.33 0.30 0.26 0.16
35          35 2.68 2.96 3.88 2.88 3.95 1.12 0.55 0.52 0.48 0.44 0.38 0.24
36          36 3.56 3.82 4.80 3.47 5.01 1.38 0.68 0.67 0.63 0.58 0.50 0.32
37          37 4.44 4.68 5.72 4.06 6.07 1.64 0.81 0.82 0.77 0.72 0.62 0.39
38          38 5.33 5.55 6.65 4.65 7.14 1.91 0.93 0.97 0.92 0.85 0.75 0.47
39          39 6.21 6.41 7.57 5.24 8.20 2.17 1.06 1.12 1.06 0.99 0.87 0.54
40          40 7.09 7.27 8.49 5.83 9.26 2.43 1.19 1.27 1.21 1.13 0.99 0.62
41          41 7.97 8.13 9.41 6.4110.32 2.69 1.32 1.41 1.36 1.27 1.11 0.70
42          42 8.86 8.9910.33 7.0011.38 2.95 1.45 1.56 1.50 1.41 1.24 0.77
43     9999999 9.74 9.8511.25 7.5912.44 3.21 1.58 1.71 1.65 1.55 1.36 0.85
```

Exhibit C.3. Sample SOURCE1 output file for input data summary and concrete analyses (*filename*.con)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------


Input Data Summary:
-------------------

 Simulation length                                      300 years
 Output edit frequency                                  150 years

     Disposal unit area                            4.99E+02 m**2
     Total dissolved solids                        3.49E+02 ppm
     Groundwater temperature                       1.50E+01 deg C
     Groundwater pH                                6.75E+00

     Saturated hydraulic conductivity:
       Recharge                                    5.80E-07 cm/s
       Soil backfill                               3.50E-03 cm/s
       Concrete                                    1.00E-10 cm/s

 Groundwater constituent concentrations:
     Ca++                                          2.10E-03 mole/L
     Cl-                                           2.04E-04 mole/L
     CO3--                                         1.00E-03 mole/L
     Mg++                                          5.21E-04 mole/L
     SO4--                                         2.62E-04 mole/L
     O2                                            3.44E-04 mole/L

 Constituent solubilities:
     Ca(OH)2                                       2.00E-02 mole/L
     CO3--                                         1.20E-03 mole/L
     Mg++                                          1.20E-03 mole/L

 Concrete constituent concentrations:
     Calcium concentration in C-S-H system         1.75E+00 mole/L
     Calcium concentration in pore fluid           2.00E-02 mole/L
     CaO content in cement                         2.11E+00 mole/L
     Free Cl-                                      1.00E-02 mole/L
     Silica concentration in C-S-H system          7.10E-01 mole/L

 Concrete design specifications:
     Compressive strength at 28 days               3.52E+02 kg/cm**2
     Poisson's ratio of concrete                   1.50E-01
     Modulus of elasticity of steel                2.04E+06 kg/cm**2
     Yield strength of steel                       4.22E+03 kg/cm**2
     Modulus of subgrade reaction                  1.41E+02 kg/cm**2
     Young's modulus of elasticity                 2.04E+05 kg/cm**2
     Concrete water/cement ratio                   4.00E-01
     Concrete density                              2.40E+00 g/cm**3
     Concrete porosity                             1.50E-01
     Cement content                                4.03E+02 kg/m**3
     Initial pH                                    1.25E+01

 Concrete pad failure model parameters:
     Radius of pad steel reinforcement             1.11E+00 cm
     Concrete pad thickness                        3.81E+01 cm
     Modulus of elasticity of steel reinforcement  2.04E+06 kg/cm**2
     Yield strength of steel reinforcement         4.22E+03 kg/cm**2
     Compressive strength of pad concrete          2.82E+02 kg/cm**2
     Spacing between steel reinforcing rods        1.52E+01 cm
     Concrete cover thickness from the center      1.52E+01 cm
     of the bottom row of steel reinforcing rods
     to the bottom of the pad
     Weight of pad cement per unit                 3.30E+02 kg/m**3
     volume concrete
```

```
    Pad initial functionality fraction          8.00E-01

Diffusion coefficients in concrete:
    NaOH, KOH                                   2.12E-11 m**2/s
    Ca(OH)2                                     1.82E-11 m**2/s
    Cl-                                         5.08E-11 m**2/s
    CO2                                         1.92E-10 m**2/s
    O2                                          2.10E-10 m**2/s
    SO4--                                       1.06E-11 m**2/s

Tumulus design specifications:
    Layers of vaults                             3
    Number of vaults wide                       10
    Number of vaults long                       11

    Vault dimensions:
      Width                                     1.52E+00 m
      Length                                    2.21E+00 m
      Height                                    1.65E+00 m

    Concrete member thickness:
      Roof                                      1.78E+01 cm
      Walls                                     1.78E+01 cm
      Floor                                     1.78E+01 cm
      Pad                                       3.81E+01 cm

    Steel reinforcement radius:
      Roof                                      7.94E-01 cm
      Walls                                     6.35E-01 cm
      Floor                                     6.35E-01 cm

    Spacing of steel reinforcement:
      Roof                                      2.54E+01 cm
      Walls                                     3.05E+01 cm
      Floor                                     3.05E+01 cm

    Concrete cover thickness on tension face:
      Roof:
        X-direction                             7.77E+00 cm
        Y-direction                             9.37E+00 cm
      Walls:
        Horizontal direction                    8.26E+00 cm
        Vertical direction                      9.52E+00 cm
      Floor:
        X-direction                             5.08E+00 cm
        Y-direction                             6.35E+00 cm

    Static load:
      Vault layer 1                             3.65E-01 kg/cm**2
      Vault layer 2                             7.10E-01 kg/cm**2
      Vault layer 3                             1.05E+00 kg/cm**2

Soil and waste properties:
    Earthen cover thickness                     1.83E+00 m
    Earthen cover density                       1.76E+00 g/cm**3
    Friction angle of waste backfill            4.00E+01 deg
    Friction angle of soil backfill             3.00E+01 deg
    Density of soil backfill                    1.76E+00 g/cm**3
    Waste density                               1.76E+00 g/cm**3
    Relative saturation of waste                3.50E-01

Concrete and waste package failure rates:
    Waste container:
      Start of failure                          0.00E+00 years
      Time to complete failure                  6.00E+01 years
    Epoxy coating:
      Start of failure                          0.00E+00 years
      Time to complete failure                  2.00E+01 years
```

```
 Nuclide-specific parameters:

                                                Diffusion coefficient
 Nuclide    Half-life Solubility   Waste     --------------------  Initial
                                     Kd        Waste     Concrete   Inventory
              (yr)     (mole/L)    (ml/g)    (m**2/s)   (m**2/s)      (g)
 -------- ---------- ---------- ---------- ---------- ----------  ----------
  Cs-137   3.00E+01   1.60E+01   1.99E+01   6.80E-12   5.12E-13    1.00E+00


Output summary:
--------------

 -----------------------------
 Annual summary for year         1
 -----------------------------

 Concrete Degradation Summary

 Concrete Member Thickness:
     Roof                                        1.77E+01 cm
     Walls                                       1.77E+01 cm
     Floor                                       1.77E+01 cm
     Pad                                         3.81E+01 cm

 Concrete loss due to sulfate attack:
     Roof                                        5.29E-02 cm
     Walls                                       5.29E-02 cm
     Floor                                       5.29E-02 cm
     Pad                                         5.29E-02 cm

 Fractional loss of yield strength
 due to Ca(OH)2 leaching:
     Roof                                        3.40E-06
     Walls                                       3.40E-06
     Floor                                       3.40E-06
     Pad                                         1.43E-06


 Corrosion results:

     Time to onset of corrosion:
       Roof                                            0 years
       Walls                                           0 years
       Floor                                           0 years

     Corrosion product layer thickness:
       Roof                                      0.00E+00 cm
       Walls                                     0.00E+00 cm
       Floor                                     0.00E+00 cm

     Remaining steel reinforcement:
       Roof                                      7.94E-01 cm
       Walls                                     6.35E-01 cm
       Floor                                     6.35E-01 cm


 Concrete Cracking Analysis

 Cracking due to corrosion of steel:
   Cask roof                                     None
   Cask walls                                    None
   Cask floor                                    None


 Cracking due to loading and shear:
   Cask roof                                     None
   Cask walls                                    None
   Cask floor                                    None
```

```
-----------------------------
Annual summary for year        150
-----------------------------

Concrete Degradation Summary

Concrete Member Thickness:
    Roof                                          9.85E+00 cm
    Walls                                         9.85E+00 cm
    Floor                                         9.85E+00 cm
    Pad                                           3.47E+01 cm

Concrete loss due to sulfate attack:
    Roof                                          7.93E+00 cm
    Walls                                         7.93E+00 cm
    Floor                                         7.93E+00 cm
    Pad                                           7.93E+00 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Roof                                          7.32E-02
    Walls                                         7.32E-02
    Floor                                         7.32E-02
    Pad                                           3.07E-02


Corrosion results:

    Time to onset of corrosion:
      Roof                                               0 years
      Walls                                              0 years
      Floor                                              0 years

    Corrosion product layer thickness:
      Roof                                        0.00E+00 cm
      Walls                                       0.00E+00 cm
      Floor                                       0.00E+00 cm

    Remaining steel reinforcement:
      Roof                                        7.94E-01 cm
      Walls                                       6.35E-01 cm
      Floor                                       6.35E-01 cm


Concrete Cracking Analysis

Cracking due to corrosion of steel:
  Cask roof                                       None
  Cask walls                                      None
  Cask floor                                      None


Cracking due to loading and shear:
  Cask roof                                       Cracked
  Cask walls                                      Cracked
  Cask floor                                      Cracked


Concrete crack characteristics:

                                            Layer of casks
                                 --------------------------------------
                                 Upper layer  middle layer  lower layer
                                 -----------  ------------  -----------

 Cask roof
    Average crack width (cm)       0.00E+00      1.30E-02      2.92E-02
    Fractional volume of cracks    0.00E+00      3.48E-04      7.22E-04

 Cask floor
    Average crack width (cm)       0.00E+00      7.46E-03      2.86E-02
    Fractional volume of cracks    0.00E+00      2.80E-04      8.51E-04
```

```
-----------------------------
Annual summary for year       300
-----------------------------

Concrete Degradation Summary

Concrete Member Thickness:
    Roof                                         1.91E+00 cm
    Walls                                        1.91E+00 cm
    Floor                                        1.91E+00 cm
    Pad                                          3.13E+01 cm

Concrete loss due to sulfate attack:
    Roof                                         1.59E+01 cm
    Walls                                        1.59E+01 cm
    Floor                                        1.59E+01 cm
    Pad                                          1.59E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Roof                                         1.05E-01
    Walls                                        1.05E-01
    Floor                                        1.05E-01
    Pad                                          4.68E-02


Corrosion results:

    Time to onset of corrosion:
      Roof                                              0 years
      Walls                                             0 years
      Floor                                             0 years

    Corrosion product layer thickness:
      Roof                                       0.00E+00 cm
      Walls                                      0.00E+00 cm
      Floor                                      0.00E+00 cm

    Remaining steel reinforcement:
      Roof                                       7.94E-01 cm
      Walls                                      6.35E-01 cm
      Floor                                      6.35E-01 cm


Concrete Cracking Analysis

Cracking due to corrosion of steel:
  Cask roof                                      None
  Cask walls                                     None
  Cask floor                                     None


Cracking due to loading and shear:
  Cask roof                                      Cracked
  Cask walls                                     Cracked
  Cask floor                                     Cracked


Concrete crack characteristics:

                                                Layer of casks
                                    --------------------------------------
                                    Upper layer  middle layer  lower layer
                                    -----------  ------------  -----------

 Cask roof
    Average crack width (cm)          4.95E-04      1.69E-02      3.33E-02
    Fractional volume of cracks       1.71E-05      5.03E-04      9.19E-04

 Cask floor
    Average crack width (cm)          0.00E+00      9.12E-03      2.55E-02
    Fractional volume of cracks       0.00E+00      4.27E-04      9.69E-04
```

Exhibit C.4. Sample SOURCE1 output of water infiltration input data (*filename*.h2o)

```
 Summary of Infiltration Data


--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------

      Year1       Year2      Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec

          1          22    12.42 12.42 11.68 10.92 15.75  8.48 23.44  6.20  8.00  6.38  5.54 15.98
         23          32     0.03  0.38  1.12  1.12  0.77  0.34  0.16  0.08  0.04  0.02  0.01  0.01
         33          33     0.91  1.24  2.04  1.71  1.83  0.60  0.29  0.23  0.19  0.16  0.13  0.09
         34          34     1.80  2.10  2.96  2.30  2.89  0.86  0.42  0.38  0.33  0.30  0.26  0.16
         35          35     2.68  2.96  3.88  2.88  3.95  1.12  0.55  0.52  0.48  0.44  0.38  0.24
         36          36     3.56  3.82  4.80  3.47  5.01  1.38  0.68  0.67  0.63  0.58  0.50  0.32
         37          37     4.44  4.68  5.72  4.06  6.07  1.64  0.81  0.82  0.77  0.72  0.62  0.39
         38          38     5.33  5.55  6.65  4.65  7.14  1.91  0.93  0.97  0.92  0.85  0.75  0.47
         39          39     6.21  6.41  7.57  5.24  8.20  2.17  1.06  1.12  1.06  0.99  0.87  0.54
         40          40     7.09  7.27  8.49  5.83  9.26  2.43  1.19  1.27  1.21  1.13  0.99  0.62
         41          41     7.97  8.13  9.41  6.41 10.32  2.69  1.32  1.41  1.36  1.27  1.11  0.70
         42          42     8.86  8.99 10.33  7.00 11.38  2.95  1.45  1.56  1.50  1.41  1.24  0.77
         43     9999999     9.74  9.85 11.25  7.59 12.44  3.21  1.58  1.71  1.65  1.55  1.36  0.85
```

Exhibit C.5. Sample SOURCE1 output of recharge release component (*filename*.rch)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------

Water and total grams in recharge component

                                        Cs-137
       Year       Water infiltration (cm)       Recharge (g)
        10           1.83034096E+01            1.12228406E-06
        20           1.83034096E+01            2.29051579E-06
        30           4.08000040E+00            2.10606104E-05
        40           1.54928007E+01            1.69660725E-05
        50           1.74531059E+01            5.13803437E-02
        60           1.74531059E+01            4.98911142E-02
        70           1.74531059E+01            3.94661352E-02
        80           1.74531059E+01            3.09160128E-02
        90           1.74531059E+01            2.40589995E-02
       100           1.74531059E+01            5.09368926E-02
       110           1.74531059E+01            3.61716822E-02
       120           1.74531059E+01            2.57364567E-02
       130           1.74531059E+01            1.83554757E-02
       140           1.74531059E+01            1.31286727E-02
       150           1.74531059E+01            9.42158233E-03
       160           1.74531059E+01            6.78717392E-03
       170           1.74531059E+01            1.08960224E-02
       180           1.74531059E+01            7.71504967E-03
       190           1.74531059E+01            5.45583107E-03
       200           1.74531059E+01            3.85183073E-03
       210           1.74531059E+01            2.71384884E-03
       220           1.74531059E+01            1.90740335E-03
       230           1.74531059E+01            1.33679295E-03
       240           1.74531059E+01            9.33853618E-04
       250           1.74531059E+01            6.50006987E-04
       260           1.74531059E+01            4.50627762E-04
       270           1.74531059E+01            3.11042415E-04
       280           1.74531059E+01            2.13683088E-04
       290           1.74531059E+01            1.46057093E-04
       300           1.74531059E+01            9.92971109E-05
```

Exhibit C.6. Sample SOURCE1 output of lateral release component (*filename.* lat)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------

Water and total grams in lateral component

                                          Cs-137
     Year       Water infiltration (cm)      Lateral (g)
      10           1.18906593E+02          5.96764630E-06
      20           1.18906593E+02          1.21796156E-05
      30           0.00000000E+00          0.00000000E+00
      40           3.12871971E+01          9.91232810E-06
      50           4.53268967E+01          1.34356841E-01
      60           4.53268967E+01          1.30547121E-01
      70           4.53268967E+01          1.03311129E-01
      80           4.53268967E+01          8.10383409E-02
      90           4.53268967E+01          6.32714480E-02
     100           4.53268967E+01          1.33854657E-01
     110           4.53268967E+01          9.54259112E-02
     120           4.53268967E+01          6.83092698E-02
     130           4.53268967E+01          4.91481237E-02
     140           4.53268967E+01          3.55774909E-02
     150           4.53268967E+01          2.59348601E-02
     160           4.53268967E+01          1.90536901E-02
     170           4.53268967E+01          2.97069252E-02
     180           4.53268967E+01          2.12500598E-02
     190           4.53268967E+01          1.51967388E-02
     200           4.53268967E+01          1.08594149E-02
     210           4.53268967E+01          7.74991605E-03
     220           4.53268967E+01          5.52061433E-03
     230           4.53268967E+01          3.92323127E-03
     240           4.53268967E+01          2.77995341E-03
     250           4.53268967E+01          1.96311902E-03
     260           4.53268967E+01          1.38088944E-03
     270           4.53268967E+01          9.67103173E-04
     280           4.53268967E+01          6.74059324E-04
     290           4.53268967E+01          4.67365026E-04
     300           4.53268967E+01          3.22240579E-04
```

Exhibit C.7. Sample SOURCE1 output of leaching summary (*filename*.sum)

```
-------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
-------------------------------------------------------------------------------

 Nuclide-specific parameters:

                                                Diffusion coefficient
 Nuclide    Half-life Solubility   Waste     ---------------------  Initial
                                     Kd       Waste      Concrete   Inventory
              (yr)     (mole/L)    (ml/g)   (m**2/s)    (m**2/s)      (g)
 --------  ---------- ---------- ---------- ---------- ----------  ----------
  Cs-137    3.00E+01   1.60E+01   1.99E+01   6.80E-12   5.12E-13    1.00E+00

 Output summary total for all vaults:

                                                 Cs-137
            -----------------------------------------------------------------------
       Year  Inventory remaining (g)  Annual leach rate (g/yr)  Cumulative leached (g)
        10         2.6192E+02                2.5321E-05                1.4950E-04
        20         2.0789E+02                4.0195E-05                4.9165E-04
        30         1.6500E+02                4.7865E-05                9.4062E-04
        40         1.3096E+02                5.1689E-05                1.4410E-03
        50         1.0131E+02                3.0956E-01                2.8844E+00
        60         7.7836E+01                2.6535E-01                5.7436E+00
        70         5.9800E+01                1.8786E-01                7.9486E+00
        80         4.6061E+01                1.3328E-01                9.5112E+00
        90         3.5562E+01                9.4924E-02                1.0622E+01
       100         2.6667E+01                1.8479E-01                1.2325E+01
 ***** End of institutional control at year     100 *****
       110         1.9783E+01                1.3160E-01                1.3865E+01
       120         1.4716E+01                9.4046E-02                1.4963E+01
       130         1.0974E+01                6.7504E-02                1.5750E+01
       140         8.2015E+00                4.8706E-02                1.6317E+01
       150         6.1416E+00                3.5356E-02                1.6727E+01
       160         4.6065E+00                2.5841E-02                1.7025E+01
       170         3.2300E+00                4.0603E-02                1.7500E+01
       180         2.2596E+00                2.8965E-02                1.7839E+01
       190         1.5766E+00                2.0653E-02                1.8081E+01
       200         1.0968E+00                1.4711E-02                1.8253E+01
       210         7.6052E-01                1.0464E-02                1.8375E+01
       220         5.2547E-01                7.4280E-03                1.8462E+01
       230         3.6165E-01                5.2600E-03                1.8524E+01
       240         2.4786E-01                3.7138E-03                1.8568E+01
       250         1.6912E-01                2.6131E-03                1.8599E+01
       260         1.1484E-01                1.8315E-03                1.8620E+01
       270         7.7590E-02                1.2781E-03                1.8635E+01
       280         5.2146E-02                8.8774E-04                1.8646E+01
 ***** Pad has cracked at year        287 *****
       290         3.4851E-02                6.1342E-04                1.8653E+01
       300         2.3158E-02                4.2154E-04                1.8658E+01


 The solubility constraints were not exceeded for Cs-137
```

Exhibit C.8. Sample SOURCE1 output of advective and diffusive release rates (*filename.* lch)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------


 Nuclide-specific parameters:

                                                Diffusion coefficient
 Nuclide     Half-life  Solubility    Waste     ---------------------    Initial
                                        Kd        Waste      Concrete   Inventory
                (yr)      (mole/L)    (ml/g)    (m**2/s)   (m**2/s)       (g)
 --------   ---------- ---------- ---------- ---------- ----------  ----------
  Cs-137     3.00E+01   1.60E+01   1.99E+01   6.80E-12   5.12E-13    1.00E+00

 Output summary per total number of vaults:

                                             Cs-137
              ----------------------------------------------------------------------
       Year  Adv leach rate (g/yr)  Dif leach rate (g/yr)  Annual leach rate (g/yr)
        10        7.6731E-08             2.5088E-26               7.6731E-08
        20        1.2180E-07             4.5826E-15               1.2180E-07
        30        1.4501E-07             3.3892E-11               1.4505E-07
        40        1.5346E-07             3.1735E-09               1.5663E-07
        50        9.3802E-04             4.8474E-08               9.3807E-04
        60        8.0380E-04             2.9335E-07               8.0409E-04
        70        5.6840E-04             8.9035E-07               5.6929E-04
        80        4.0191E-04             1.9649E-06               4.0388E-04
        90        2.8415E-04             3.4989E-06               2.8765E-04
       100        5.5476E-04             5.2119E-06               5.5997E-04
 ***** End of institutional control at year        100 *****
       110        3.9195E-04             6.8275E-06               3.9878E-04
       120        2.7676E-04             8.2293E-06               2.8499E-04
       130        1.9526E-04             9.3002E-06               2.0456E-04
       140        1.3761E-04             9.9888E-06               1.4759E-04
       150        9.6844E-05             1.0297E-05               1.0714E-04
       160        6.8043E-05             1.0263E-05               7.8306E-05
       170        1.1369E-04             9.3508E-06               1.2304E-04
       180        7.9536E-05             8.2371E-06               8.7773E-05
       190        5.5497E-05             7.0863E-06               6.2584E-05
       200        3.8610E-05             5.9694E-06               4.4580E-05
       210        2.6774E-05             4.9343E-06               3.1708E-05
       220        1.8500E-05             4.0091E-06               2.2509E-05
       230        1.2733E-05             3.2061E-06               1.5939E-05
       240        8.7277E-06             2.5263E-06               1.1254E-05
       250        5.9553E-06             1.9632E-06               7.9186E-06
       260        4.0443E-06             1.5057E-06               5.5501E-06
       270        2.7327E-06             1.1404E-06               3.8732E-06
       280        1.8367E-06             8.5338E-07               2.6901E-06
 ***** Pad has cracked at year        287 *****
       290        1.2277E-06             6.3117E-07               1.8589E-06
       300        8.1584E-07             4.6155E-07               1.2774E-06


 The solubility constraints were not exceeded for Cs-137
```

Exhibit C.9. Sample SOURCE1 output summary for intact vaults (*filename*.vt1)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------


 Nuclide-specific parameters:

                                                  Diffusion coefficient
 Nuclide     Half-life Solubility     Waste      ---------------------   Initial
                                       Kd          Waste      Concrete   Inventory
               (yr)      (mole/L)    (ml/g)     (m**2/s)    (m**2/s)       (g)
 --------  ---------- ---------- ---------- ---------- ----------  ----------
  Cs-137    3.00E+01   1.60E+01   1.99E+01   6.80E-12   5.12E-13    1.00E+00

 Output summary per number of intact vaults:

                                          Cs-137
                  ----------------------------------------------------------------------
       Year   Inventory remaining (g)  Adv leach rate (g/yr)  Dif leach rate (g/yr)
        10        7.9370E-01              7.6731E-08              2.5088E-26
        20        6.2996E-01              1.2180E-07              4.5826E-15
        30        5.0000E-01              1.4501E-07              3.3892E-11
        40        3.9685E-01              1.5346E-07              3.1735E-09
        50        3.1498E-01              1.5225E-07              4.9602E-08
        60        2.4999E-01              1.4501E-07              3.0997E-07
        70        1.9841E-01              1.1509E-07              9.7205E-07
        80        1.5746E-01              9.1338E-08              2.2107E-06
        90        1.2495E-01              7.2478E-08              4.0468E-06
       100        9.9124E-02              5.7497E-08              6.3514E-06
       110        7.8604E-02              4.5595E-08              8.8959E-06
       120        6.2294E-02              3.6135E-08              1.1427E-05
       130        4.9328E-02              2.8613E-08              1.3717E-05
       140        3.9017E-02              2.2633E-08              1.5597E-05
       150        3.0819E-02              1.7878E-08              1.6965E-05
       160        2.4303E-02              1.4098E-08              1.7783E-05
       170        0.0000E+00              0.0000E+00              0.0000E+00
       180        0.0000E+00              0.0000E+00              0.0000E+00
       190        0.0000E+00              0.0000E+00              0.0000E+00
       200        0.0000E+00              0.0000E+00              0.0000E+00
       210        0.0000E+00              0.0000E+00              0.0000E+00
       220        0.0000E+00              0.0000E+00              0.0000E+00
       230        0.0000E+00              0.0000E+00              0.0000E+00
       240        0.0000E+00              0.0000E+00              0.0000E+00
       250        0.0000E+00              0.0000E+00              0.0000E+00
       260        0.0000E+00              0.0000E+00              0.0000E+00
       270        0.0000E+00              0.0000E+00              0.0000E+00
       280        0.0000E+00              0.0000E+00              0.0000E+00
       290        0.0000E+00              0.0000E+00              0.0000E+00
       300        0.0000E+00              0.0000E+00              0.0000E+00
```

Exhibit C.10. Sample SOURCE1 output summary for cracked vaults (*filename*.vt2)

```
--------------------------------------------------------------------------------
Sample run of Cs-137 in a tumulus-type disposal facility (with concrete pad)
--------------------------------------------------------------------------------

 Nuclide-specific parameters:

                                                  Diffusion coefficient
 Nuclide     Half-life Solubility   Waste      ---------------------   Initial
                                     Kd          Waste     Concrete   Inventory
                (yr)     (mole/L)   (ml/g)     (m**2/s)   (m**2/s)       (g)
 --------  ---------- ---------- ---------- ---------- ----------  ----------
  Cs-137     3.00E+01   1.60E+01   1.99E+01   6.80E-12   5.12E-13    1.00E+00

 Output summary per number of cracked vaults:

                                      Cs-137
       -------------------------------------------------------------------------
       Year  Inventory remaining (g)  Adv leach rate (g/yr)  Dif leach rate (g/yr)
        10        0.0000E+00               0.0000E+00               0.0000E+00
        20        0.0000E+00               0.0000E+00               0.0000E+00
        30        0.0000E+00               0.0000E+00               0.0000E+00
        40        0.0000E+00               0.0000E+00               0.0000E+00
        50        2.9106E-01               2.8137E-03               4.6218E-08
        60        2.0761E-01               2.4111E-03               2.6011E-07
        70        1.4681E-01               1.7050E-03               7.2695E-07
        80        1.0381E-01               1.2055E-03               1.4733E-06
        90        7.3389E-02               8.5231E-04               2.4031E-06
       100        7.1650E-02               8.3212E-04               4.6421E-06
       110        5.0622E-02               5.8791E-04               5.7933E-06
       120        3.5743E-02               4.1512E-04               6.6305E-06
       130        2.5217E-02               2.9287E-04               7.0917E-06
       140        1.7771E-02               2.0640E-04               7.1846E-06
       150        1.2506E-02               1.4526E-04               6.9627E-06
       160        8.7867E-03               1.0206E-04               6.5026E-06
       170        9.7878E-03               1.1369E-04               9.3508E-06
       180        6.8472E-03               7.9536E-05               8.2371E-06
       190        4.7775E-03               5.5497E-05               7.0863E-06
       200        3.3236E-03               3.8610E-05               5.9694E-06
       210        2.3046E-03               2.6774E-05               4.9343E-06
       220        1.5923E-03               1.8500E-05               4.0091E-06
       230        1.0959E-03               1.2733E-05               3.2061E-06
       240        7.5110E-04               8.7277E-06               2.5263E-06
       250        5.1247E-04               5.9553E-06               1.9632E-06
       260        3.4800E-04               4.0443E-06               1.5057E-06
       270        2.3512E-04               2.7327E-06               1.1404E-06
       280        1.5802E-04               1.8367E-06               8.5338E-07
       290        1.0561E-04               1.2277E-06               6.3117E-07
       300        7.0175E-05               8.1584E-07               4.6155E-07
```

Exhibit C.11. Sample SOURCE1 input file for ^{137}Cs in a tumulus-type facility without a concrete pad (*filename*.inp)

```
Sample run of Cs-137 in a tumulus-type disposal facility
       300       100 0 0    5 0    5 0  150 0    5 0    5 0    5 0    5
    3   10   11    3
 6.000E+01 8.700E+01 6.500E+01
 7.000E+00 7.000E+00 7.000E+00
 3.060E+00 3.690E+00 3.250E+00 3.750E+00 2.000E+00 2.500E+00
 3.125E-01 1.000E+01 2.500E-01 1.200E+01 2.500E-01 1.200E+01
 2.000E+03 4.000E+01 1.760E+00 3.000E+01
 7.200E+01 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.250E+01 4.030E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 2.900E+07 6.000E+04 2.000E+10
 1.750E+00 2.000E-02 7.100E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 3.440E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 6.000E+01 0.000E+00 2.000E+01
 4.990E+02 5.800E-07 3.500E-03 1.000E-10
water.dat
    1
Cs-137   1.370E+02 3.000E+01 1.600E+01 1.990E+01 1.000E+00 6.800E-12 5.120E-13
```

Exhibit C.12. Sample SOURCE2 input file for ^{238}U in a silo-type facility (*filename*.inp)

```
Sample run of U-238 in a silo
      1000    1 0 0   25 0   25 0  200 0   25 0   25
 2.400E+02 5.100E+01
 1.200E+01 6.000E+00 1.200E+01
 5.810E+00 5.810E+00 0.000E+00 0.000E+00 5.810E+00 5.810E+00
 5.980E-02 5.980E-02
 1.875E-01 6.000E+00 0.000E+00 0.000E+00 1.875E-01 6.000E+00
 3.000E+02 4.000E+01 1.760E+00 3.000E+01
 7.200E+01 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.260E+01 3.850E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 2.900E+07 6.000E+04 2.000E+10
 1.750E+00 2.000E-02 7.100E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 1.630E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 2.000E+01 0.000E+00 5.000E+01
 1.000E+01 5.800E-07 3.500E-03 1.000E-10
water.dat
    1
U-238    2.380E+02 4.470E+09 1.460E-06 5.560E+01 0.000E+00 3.110E-14 3.500E-15
      1975
      1975      1975 1.000E+02
      1976      1980 2.000E+01
      1981      1981 2.500E+02
      1982   9999999 0.000E+00
```

Exhibit C.13. Sample SOURCE2 output file for input data summary and concrete analyses (*filename*.con)

```
--------------------------------------------------------------------------------
Sample run of U-238 in a silo
--------------------------------------------------------------------------------


Input Data Summary:
-------------------

 Simulation length                                     1000 years
 Output edit frequency                                  200 years

 Disposal technology: silo

     Disposal unit area                            1.00E+01 m**2
     Total dissolved solids                        3.49E+02 ppm
     Groundwater temperature                       1.50E+01 deg c
     Groundwater pH                                6.75E+00

     Saturated hydraulic conductivity:
       Recharge                                    5.80E-07 cm/s
       Soil backfill                               3.50E-03 cm/s
       Concrete                                    1.00E-10 cm/s

 Groundwater constituent concentrations:
     Ca++                                          2.10E-03 mole/L
     Cl-                                           2.04E-04 mole/L
     CO3--                                         1.00E-03 mole/L
     Mg++                                          5.21E-04 mole/L
     SO4-- (inside silo or well)                   2.62E-04 mole/L
     SO4-- (outside silo or well)                  2.62E-04 mole/L
     O2                                            1.63E-04 mole/L

 Constituent solubilities:
     Ca(OH)2                                       2.00E-02 mole/L
     CO3--                                         1.20E-03 mole/L
     Mg++                                          1.20E-03 mole/L

 Concrete constituent concentrations:
     Calcium Concentration in C-S-H system         1.75E+00 mole/L
     Calcium concentration in pore fluid           2.00E-02 mole/L
     CaO content in cement                         2.11E+00 mole/L
     Free Cl-                                      1.00E-02 mole/L
     Silica concentration in C-S-H system          7.10E-01 mole/L

 Concrete design specifications:
     Compressive strength at 28 days               3.52E+02 kg/cm**2
     Poisson's ratio of concrete                   1.50E-01
     Modulus of elasticity of steel                2.04E+06 kg/cm**2
     Yield strength of steel                       4.22E+03 kg/cm**2
     Modulus of subgrade reaction                  2.11E+01 kg/cm**2
     Young's modulus of elasticity                 2.04E+05 kg/cm**2
     Concrete water/cement ratio                   4.00E-01
     Concrete density                              2.40E+00 g/cm**3
     Concrete porosity                             1.50E-01
     Cement content                                3.85E+02 kg/m**3
     Initial pH                                    1.26E+01

 Diffusion coefficients in concrete:
     NaOH, KOH                                     2.12E-11 m**2/s
     Ca(OH)2                                       1.82E-11 m**2/s
     Cl-                                           5.08E-11 m**2/s
     CO2                                           1.92E-10 m**2/s
     O2                                            2.10E-10 m**2/s
     SO4--                                         1.06E-11 m**2/s
```

```
Silo design specifications:

    Silo dimensions:
      Radius                                      1.30E+00 m
      Height                                      6.10E+00 m

    Concrete member thickness:
      Roof                                        3.05E+01 cm
      Walls                                       1.52E+01 cm
      Floor                                       3.05E+01 cm

    Steel reinforcement radius:
      Roof                                        4.76E-01 cm
      Walls                                       0.00E+00 cm
      Floor                                       4.76E-01 cm

    Spacing of steel reinforcement:
      Roof                                        1.52E+01 cm
      Walls                                       0.00E+00 cm
      Floor                                       1.52E+01 cm

    Corrugated steel thickness:
      Compression face                            1.52E-01 cm
      Tension face                                1.52E-01 cm

    Concrete cover thickness on tension face:
      Roof:
        X-direction                               1.48E+01 cm
        Y-direction                               1.48E+01 cm
      Walls:
        Horizontal direction                      0.00E+00 cm
        Vertical direction                        0.00E+00 cm
      Floor:
        X-direction                               1.48E+01 cm
        Y-direction                               1.48E+01 cm

    Static load                                   3.95E-01 kg/cm**2

Soil and waste properties:
    Earthen cover thickness                       1.83E+00 m
    Earthen cover density                         1.76E+00 g/cm**3
    Friction angle of waste backfill              4.00E+01 deg
    Friction angle of soil backfill               3.00E+01 deg
    Density of soil backfill                      1.76E+00 g/cm**3
    Waste density                                 1.76E+00 g/cm**3
    Relative saturation of waste                  3.50E-01

Concrete and steel failure rates:
    Epoxy coating:
      Start of failure                            0.00E+00 years
      Time to complete failure                    2.00E+01 years
    Steel liner:
      Start of failure                            0.00E+00 years
      Time to complete failure                    5.00E+01 years

Nuclide-specific parameters:

Nuclide    Half-life Solubility   Waste    Diffusion coefficient   Initial
                                   kd        Waste     Concrete   inventory
             (yr)     (mole/l)    (ml/g)    (m**2/s)   (m**2/s)      (g)
--------  ---------- ---------- ---------- ---------- ----------  ----------
 U-238     4.47E+09   1.46E-06   5.56E+01   3.11E-14   3.50E-15    1.00E+02

-----------------------------
Annual Summary for Year       1975
-----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                                     3.04E+01 cm
    Silo wall                                     1.52E+01 cm
```

```
    Silo floor                                       3.04E+01 cm

Concrete loss due to sulfate attack:
    Silo roof                                        5.10E-02 cm
    Silo wall                                        5.10E-02 cm
    Silo floor                                       5.10E-02 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                                        0.00E+00
    Silo wall                                        0.00E+00
    Silo floor                                       0.00E+00

Corrosion results:
    Time to onset of corrosion:
      Silo roof                                             0 years
      Silo floor                                            0 years

    Corrosion product layer thickness:
      Silo roof                                      0.00E+00 cm
      Silo floor                                     0.00E+00 cm

    Remaining steel reinforcement:
      Silo roof                                      4.76E-01 cm
      Silo floor                                     4.76E-01 cm

    Remaining corrugated steel:
      Compression face                               1.49E-01 cm
      Tension face                                   1.49E-01 cm

Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                          none
  Silo wall                                          none
  Silo floor                                         none

Cracking due to loading and shear:

  Silo roof                                          none
  Silo wall                                          none
  Silo floor                                         none


  Kd/diffusion controlled leach rates (g/yr)
----------------------------------------------
U-238     5.10E-07

-----------------------------
Annual Summary for Year      2175
-----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                                        2.03E+01 cm
    Silo wall                                        5.03E+00 cm
    Silo floor                                       2.03E+01 cm

Concrete loss due to sulfate attack:
    Silo roof                                        1.02E+01 cm
    Silo wall                                        1.02E+01 cm
    Silo floor                                       1.02E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                                        0.00E+00
    Silo wall                                        0.00E+00
    Silo floor                                       0.00E+00
```

```
Corrosion results:
    Time to onset of corrosion:
      Silo roof                                    0 years
      Silo floor                                   0 years

    Corrosion product layer thickness:
      Silo roof                             0.00E+00 cm
      Silo floor                            0.00E+00 cm

    Remaining steel reinforcement:
      Silo roof                             4.76E-01 cm
      Silo floor                            4.76E-01 cm

    Remaining corrugated steel:
      Compression face                      0.00E+00 cm
      Tension face                          0.00E+00 cm

Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                 none
  Silo wall                                 none
  Silo floor                                none

Cracking due to loading and shear:

  Silo roof                                 none
  Silo wall                                 none
  Silo floor                                none


  Kd/diffusion controlled leach rates (g/yr)
---------------------------------------------
U-238     1.10E-04

-----------------------------
Annual Summary for Year      2375
-----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                               1.01E+01 cm
    Silo wall                               0.00E+00 cm
    Silo floor                              1.01E+01 cm

Concrete loss due to sulfate attack:
    Silo roof                               2.04E+01 cm
    Silo wall                               1.52E+01 cm
    Silo floor                              2.04E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                               0.00E+00
    Silo wall                               0.00E+00
    Silo floor                              0.00E+00

Corrosion results:
    Time to onset of corrosion:
      Silo roof                                    0 years
      Silo floor                                   0 years

    Corrosion product layer thickness:
      Silo roof                             0.00E+00 cm
      Silo floor                            0.00E+00 cm

    Remaining steel reinforcement:
      Silo roof                             4.76E-01 cm
      Silo floor                            4.76E-01 cm
```

```
    Remaining corrugated steel:
      Compression face                            0.00E+00 cm
      Tension face                                0.00E+00 cm

Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                       none
  Silo wall                                       none
  Silo floor                                      none

Cracking due to loading and shear:

  Silo roof                                       none
  Silo wall                                       cracked
  Silo floor                                      none


Concrete crack characteristics:


 Silo wall
    Average crack width (cm)                      4.24E-01
    Fractional volume of cracks                   2.98E-09


  Kd/diffusion controlled leach rates (g/yr)
---------------------------------------------
U-238     1.01E+00

----------------------------
Annual Summary for Year      2575
----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                                     0.00E+00 cm
    Silo wall                                     0.00E+00 cm
    Silo floor                                    0.00E+00 cm

Concrete loss due to sulfate attack:
    Silo roof                                     3.05E+01 cm
    Silo wall                                     1.52E+01 cm
    Silo floor                                    3.05E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                                     0.00E+00
    Silo wall                                     1.00E+00
    Silo floor                                    0.00E+00

Corrosion results:
    Time to onset of corrosion:
      Silo roof                                        569 years
      Silo floor                                       569 years

    Corrosion product layer thickness:
      Silo roof                                   6.34E-01 cm
      Silo floor                                  6.34E-01 cm

    Remaining steel reinforcement:
      Silo roof                                   0.00E+00 cm
      Silo floor                                  0.00E+00 cm

    Remaining corrugated steel:
      Compression face                            0.00E+00 cm
      Tension face                                0.00E+00 cm
```

```
Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                       none
  Silo wall                                       none
  Silo floor                                      none

Cracking due to loading and shear:

  Silo roof                                       none
  Silo wall                                       cracked
  Silo floor                                      none


Concrete crack characteristics:


 Silo wall
    Average crack width (cm)                      4.24E-01
    Fractional volume of cracks                   2.98E-09


  Kd/diffusion controlled leach rates (g/yr)
---------------------------------------------
U-238     3.66E-01

-----------------------------
Annual Summary for Year       2775
-----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                                     0.00E+00 cm
    Silo wall                                     0.00E+00 cm
    Silo floor                                    0.00E+00 cm

Concrete loss due to sulfate attack:
    Silo roof                                     3.05E+01 cm
    Silo wall                                     1.52E+01 cm
    Silo floor                                    3.05E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                                     0.00E+00
    Silo wall                                     1.00E+00
    Silo floor                                    0.00E+00

Corrosion results:
    Time to onset of corrosion:
      Silo roof                                        569 years
      Silo floor                                       569 years

    Corrosion product layer thickness:
      Silo roof                                   6.34E-01 cm
      Silo floor                                  6.34E-01 cm

    Remaining steel reinforcement:
      Silo roof                                   0.00E+00 cm
      Silo floor                                  0.00E+00 cm

    Remaining corrugated steel:
      Compression face                            0.00E+00 cm
      Tension face                                0.00E+00 cm

Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                       none
```

```
  Silo wall                                       none
  Silo floor                                      none

Cracking due to loading and shear:

  Silo roof                                       none
  Silo wall                                       cracked
  Silo floor                                      none


Concrete crack characteristics:


 Silo wall
    Average crack width (cm)                      4.24E-01
    Fractional volume of cracks                   2.98E-09



  Kd/diffusion controlled leach rates (g/yr)
----------------------------------------------
U-238     1.33E-01

-----------------------------
Annual Summary for Year       2975
-----------------------------

Concrete Degradation Summary


Member thickness:
    Silo roof                                     0.00E+00 cm
    Silo wall                                     0.00E+00 cm
    Silo floor                                    0.00E+00 cm

Concrete loss due to sulfate attack:
    Silo roof                                     3.05E+01 cm
    Silo wall                                     1.52E+01 cm
    Silo floor                                    3.05E+01 cm

Fractional loss of yield strength
due to Ca(OH)2 leaching:
    Silo roof                                     0.00E+00
    Silo wall                                     1.00E+00
    Silo floor                                    0.00E+00

Corrosion results:
    Time to onset of corrosion:
      Silo roof                                        569 years
      Silo floor                                       569 years

    Corrosion product layer thickness:
      Silo roof                                   6.34E-01 cm
      Silo floor                                  6.34E-01 cm

    Remaining steel reinforcement:
      Silo roof                                   0.00E+00 cm
      Silo floor                                  0.00E+00 cm

    Remaining corrugated steel:
      Compression face                            0.00E+00 cm
      Tension face                                0.00E+00 cm

Concrete Cracking Analysis

Cracking due to corrosion of steel:

  Silo roof                                       none
  Silo wall                                       none
  Silo floor                                      none
```

```
Cracking due to loading and shear:

  Silo roof                                       none
  Silo wall                                       cracked
  Silo floor                                      none


Concrete crack characteristics:


 Silo wall
    Average crack width (cm)                      4.24E-01
    Fractional volume of cracks                   2.98E-09


  Kd/diffusion controlled leach rates (g/yr)
----------------------------------------------
U-238     4.83E-02
```

Exhibit C.14. Sample SOURCE2 output of water infiltration input data (*filename*.h2o)

```
 Summary of Infiltration Data


--------------------------------------------------------------------------------
Sample run of U-238 in a silo
--------------------------------------------------------------------------------

      Year1      Year2     Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec

          1         22    12.42 12.42 11.68 10.92 15.75  8.48 23.44  6.20  8.00  6.38  5.54 15.98
         23         32     0.03  0.38  1.12  1.12  0.77  0.34  0.16  0.08  0.04  0.02  0.01  0.01
         33         33     0.91  1.24  2.04  1.71  1.83  0.60  0.29  0.23  0.19  0.16  0.13  0.09
         34         34     1.80  2.10  2.96  2.30  2.89  0.86  0.42  0.38  0.33  0.30  0.26  0.16
         35         35     2.68  2.96  3.88  2.88  3.95  1.12  0.55  0.52  0.48  0.44  0.38  0.24
         36         36     3.56  3.82  4.80  3.47  5.01  1.38  0.68  0.67  0.63  0.58  0.50  0.32
         37         37     4.44  4.68  5.72  4.06  6.07  1.64  0.81  0.82  0.77  0.72  0.62  0.39
         38         38     5.33  5.55  6.65  4.65  7.14  1.91  0.93  0.97  0.92  0.85  0.75  0.47
         39         39     6.21  6.41  7.57  5.24  8.20  2.17  1.06  1.12  1.06  0.99  0.87  0.54
         40         40     7.09  7.27  8.49  5.83  9.26  2.43  1.19  1.27  1.21  1.13  0.99  0.62
         41         41     7.97  8.13  9.41  6.41 10.32  2.69  1.32  1.41  1.36  1.27  1.11  0.70
         42         42     8.86  8.99 10.33  7.00 11.38  2.95  1.45  1.56  1.50  1.41  1.24  0.77
         43    9999999     9.74  9.85 11.25  7.59 12.44  3.21  1.58  1.71  1.65  1.55  1.36  0.85
```

Exhibit C.15. Sample SOURCE2 output of recharge release component (*filename*.rch)

```
-------------------------------------------------------------------------------
Sample run of U-238 in a silo
-------------------------------------------------------------------------------

Water and total grams in recharge component

                                            U-238
      Year        Water infiltration (cm)       Recharge (g)
     1999             4.08000040E+00          5.73542311E-05
     2024             1.74531059E+01          6.45851978E-05
     2049             1.74531059E+01          6.45851978E-05
     2074             1.74531059E+01          6.45851978E-05
     2099             1.74531059E+01          6.45851978E-05
     2124             1.74531059E+01          6.45851978E-05
     2149             1.74531059E+01          6.45851978E-05
     2174             1.74531059E+01          6.45851978E-05
     2199             1.74531059E+01          6.45851978E-05
     2224             1.74531059E+01          5.97685635E-01
     2249             1.74531059E+01          5.26643217E-01
     2274             1.74531059E+01          4.64045286E-01
     2299             1.74531059E+01          4.08888131E-01
     2324             1.74531059E+01          3.60286891E-01
     2349             1.74531059E+01          3.17462385E-01
     2374             1.74531059E+01          2.79728115E-01
     2399             1.74531059E+01          2.46479124E-01
     2424             1.74531059E+01          2.17181981E-01
     2449             1.74531059E+01          1.91367343E-01
     2474             1.74531059E+01          1.68621048E-01
     2499             1.74531059E+01          1.48578435E-01
     2524             1.74531059E+01          1.30918041E-01
     2549             1.74531059E+01          1.15356930E-01
     2574             1.74531059E+01          1.01645432E-01
     2599             1.74531059E+01          8.95636678E-02
     2624             1.74531059E+01          7.89180100E-02
     2649             1.74531059E+01          6.95376843E-02
     2674             1.74531059E+01          6.12722896E-02
     2699             1.74531059E+01          5.39893135E-02
     2724             1.74531059E+01          4.75720204E-02
     2749             1.74531059E+01          4.19175364E-02
     2774             1.74531059E+01          3.69351283E-02
     2799             1.74531059E+01          3.25449593E-02
     2824             1.74531059E+01          2.86766011E-02
     2849             1.74531059E+01          2.52680480E-02
     2874             1.74531059E+01          2.22646333E-02
     2899             1.74531059E+01          1.96182150E-02
     2924             1.74531059E+01          1.72863565E-02
     2949             1.74531059E+01          1.52316606E-02
     2974             1.74531059E+01          1.34211946E-02
```

Exhibit C.16. Sample SOURCE2 output of lateral release component (*filename*.lat)

```
--------------------------------------------------------------------------------
Sample run of U-238 in a silo
--------------------------------------------------------------------------------

Water and total grams in lateral component

                                          U-238
       Year       Water infiltration (cm)        Lateral (g)
      1999            0.00000000E+00           0.00000000E+00
      2024            4.53268967E+01           4.50711632E-05
      2049            4.53268967E+01           4.50711632E-05
      2074            4.53268967E+01           4.50711632E-05
      2099            4.53268967E+01           4.50711632E-05
      2124            4.53268967E+01           4.50711632E-05
      2149            4.53268967E+01           4.50711632E-05
      2174            4.53268967E+01           4.50711632E-05
      2199            4.53268967E+01           4.50711632E-05
      2224            4.53268967E+01           1.55329180E+00
      2249            4.53268967E+01           1.36866379E+00
      2274            4.53268967E+01           1.20598149E+00
      2299            4.53268967E+01           1.06263661E+00
      2324            4.53268967E+01           9.36329484E-01
      2349            4.53268967E+01           8.25035334E-01
      2374            4.53268967E+01           7.26969719E-01
      2399            4.53268967E+01           6.40560806E-01
      2424            4.53268967E+01           5.64422131E-01
      2449            4.53268967E+01           4.97334003E-01
      2474            4.53268967E+01           4.38219875E-01
      2499            4.53268967E+01           3.86132210E-01
      2524            4.53268967E+01           3.40235621E-01
      2549            4.53268967E+01           2.99794614E-01
      2574            4.53268967E+01           2.64160603E-01
      2599            4.53268967E+01           2.32762009E-01
      2624            4.53268967E+01           2.05095589E-01
      2649            4.53268967E+01           1.80717632E-01
      2674            4.53268967E+01           1.59237131E-01
      2699            4.53268967E+01           1.40309811E-01
      2724            4.53268967E+01           1.23632275E-01
      2749            4.53268967E+01           1.08937152E-01
      2774            4.53268967E+01           9.59886461E-02
      2799            4.53268967E+01           8.45792666E-02
      2824            4.53268967E+01           7.45260045E-02
      2849            4.53268967E+01           6.56677261E-02
      2874            4.53268967E+01           5.78623153E-02
      2899            4.53268967E+01           5.09846881E-02
      2924            4.53268967E+01           4.49245498E-02
      2949            4.53268967E+01           3.95847112E-02
      2974            4.53268967E+01           3.48796062E-02
```

Exhibit C.17. Sample SOURCE2 output of leaching summary (*filename*.sum)

```
--------------------------------------------------------------------------------
Sample run of U-238 in a silo
--------------------------------------------------------------------------------


 Nuclide-specific parameters:

 Nuclide     Half-life  Solubility    Waste     Diffusion coefficient    Initial
                                       kd        Waste      Concrete    inventory
               (yr)      (mole/l)    (ml/g)    (m**2/s)    (m**2/s)        (g)
 --------  ---------- ---------- ---------- ---------- ----------  ----------
  U-238      4.47E+09   1.46E-06   5.56E+01   3.11E-14   3.50E-15    1.00E+02

Output summary:
---------------
                                          U-238
             ---------------------------------------------------------------------------
      Year  Inventory remaining (g)  Annual leach rate (g/yr)  Cumulative leached (g/yr)
      1999          4.5000E+02              5.7354E-05               7.1527E-04
      2024          4.5000E+02              1.0966E-04               2.8865E-03
      2049          4.5000E+02              1.0966E-04               5.6279E-03
      2074          4.5000E+02              1.0966E-04               8.3693E-03
      2099          4.5000E+02              1.0966E-04               1.1111E-02
      2124          4.5000E+02              1.0966E-04               1.3852E-02
      2149          4.5000E+02              1.0966E-04               1.6594E-02
      2174          4.5000E+02              1.0966E-04               1.9335E-02
      2199          4.5000E+02              1.0966E-04               2.2076E-02
      2224          4.2388E+02              2.1510E+00               2.6142E+01
      2249          3.7350E+02              1.8953E+00               7.6525E+01
      2274          3.2910E+02              1.6700E+00               1.2092E+02
      2299          2.8999E+02              1.4715E+00               1.6004E+02
      2324          2.5552E+02              1.2966E+00               1.9451E+02
      2349          2.2515E+02              1.1425E+00               2.2488E+02
      2374          1.9838E+02              1.0067E+00               2.5164E+02
      2399          1.7480E+02              8.8704E-01               2.7522E+02
      2424          1.5403E+02              7.8160E-01               2.9600E+02
      2449          1.3572E+02              6.8870E-01               3.1430E+02
      2474          1.1959E+02              6.0684E-01               3.3044E+02
      2499          1.0537E+02              5.3471E-01               3.4465E+02
      2524          9.2848E+01              4.7115E-01               3.5718E+02
      2549          8.1812E+01              4.1515E-01               3.6821E+02
      2574          7.2088E+01              3.6581E-01               3.7794E+02
      2599          6.3519E+01              3.2233E-01               3.8650E+02
      2624          5.5969E+01              2.8401E-01               3.9405E+02
      2649          4.9317E+01              2.5026E-01               4.0071E+02
      2674          4.3455E+01              2.2051E-01               4.0657E+02
      2699          3.8290E+01              1.9430E-01               4.1173E+02
      2724          3.3738E+01              1.7120E-01               4.1628E+02
      2749          2.9728E+01              1.5085E-01               4.2030E+02
      2774          2.6195E+01              1.3292E-01               4.2383E+02
      2799          2.3081E+01              1.1712E-01               4.2694E+02
      2824          2.0338E+01              1.0320E-01               4.2969E+02
      2849          1.7920E+01              9.0936E-02               4.3210E+02
      2874          1.5790E+01              8.0127E-02               4.3423E+02
      2899          1.3913E+01              7.0603E-02               4.3611E+02
      2924          1.2260E+01              6.2211E-02               4.3776E+02
      2949          1.0802E+01              5.4816E-02               4.3922E+02
      2974          9.5184E+00              4.8301E-02               4.4050E+02


 The solubility constraints were exceeded for U-238
```

Exhibit C.18. Sample SOURCE2 output of advective and diffusive release rates (*filename*.lch)

```
--------------------------------------------------------------------------------
Sample run of U-238 in a silo
--------------------------------------------------------------------------------


 Nuclide-specific parameters:

 Nuclide    Half-life Solubility   Waste    Diffusion coefficient   Initial
                                     kd       Waste     Concrete   inventory
              (yr)     (mole/l)    (ml/g)   (m**2/s)   (m**2/s)      (g)
 -------- ---------- ---------- ---------- ---------- ----------  ----------
  U-238    4.47E+09   1.46E-06   5.56E+01   3.11E-14   3.50E-15    1.00E+02

Output summary:
---------------
                                               U-238
            ---------------------------------------------------------------------
      Year  Adv leach rate (g/yr)  Dif leach rate (g/yr)  Annual leach rate (g/yr)
     1999        5.7354E-05             0.0000E+00              5.7354E-05
     2024        1.0966E-04             0.0000E+00              1.0966E-04
     2049        1.0966E-04             0.0000E+00              1.0966E-04
     2074        1.0966E-04             0.0000E+00              1.0966E-04
     2099        1.0966E-04             0.0000E+00              1.0966E-04
     2124        1.0966E-04             0.0000E+00              1.0966E-04
     2149        1.0966E-04             0.0000E+00              1.0966E-04
     2174        1.0966E-04             0.0000E+00              1.0966E-04
     2199        1.0966E-04             0.0000E+00              1.0966E-04
     2224        2.1510E+00             0.0000E+00              2.1510E+00
     2249        1.8953E+00             0.0000E+00              1.8953E+00
     2274        1.6700E+00             0.0000E+00              1.6700E+00
     2299        1.4715E+00             0.0000E+00              1.4715E+00
     2324        1.2966E+00             0.0000E+00              1.2966E+00
     2349        1.1425E+00             0.0000E+00              1.1425E+00
     2374        1.0067E+00             0.0000E+00              1.0067E+00
     2399        8.8704E-01             0.0000E+00              8.8704E-01
     2424        7.8160E-01             0.0000E+00              7.8160E-01
     2449        6.8870E-01             0.0000E+00              6.8870E-01
     2474        6.0684E-01             0.0000E+00              6.0684E-01
     2499        5.3471E-01             0.0000E+00              5.3471E-01
     2524        4.7115E-01             0.0000E+00              4.7115E-01
     2549        4.1515E-01             0.0000E+00              4.1515E-01
     2574        3.6581E-01             0.0000E+00              3.6581E-01
     2599        3.2233E-01             0.0000E+00              3.2233E-01
     2624        2.8401E-01             0.0000E+00              2.8401E-01
     2649        2.5026E-01             0.0000E+00              2.5026E-01
     2674        2.2051E-01             8.7380E-37              2.2051E-01
     2699        1.9430E-01             1.0117E-35              1.9430E-01
     2724        1.7120E-01             9.8592E-35              1.7120E-01
     2749        1.5085E-01             8.2226E-34              1.5085E-01
     2774        1.3292E-01             5.9554E-33              1.3292E-01
     2799        1.1712E-01             3.7943E-32              1.1712E-01
     2824        1.0320E-01             2.1507E-31              1.0320E-01
     2849        9.0936E-02             1.0956E-30              9.0936E-02
     2874        8.0127E-02             5.0601E-30              8.0127E-02
     2899        7.0603E-02             2.1361E-29              7.0603E-02
     2924        6.2211E-02             8.3000E-29              6.2211E-02
     2949        5.4816E-02             2.9877E-28              5.4816E-02
     2974        4.8301E-02             1.0020E-27              4.8301E-02


 The solubility constraints were exceeded for U-238
```

Exhibit C.19. Sample SOURCE2 input file for ^{137}Cs in a well-type facility (*filename*.inp)

```
Sample run of Cs-137 in a well
      1000    2 0 0    1 0    1 0   50 0    1 0    1
 1.920E+02 1.538E+01
 1.200E+01 7.500E-01 1.200E+01
 5.810E+00 5.810E+00 0.000E+00 0.000E+00 0.000E+00 0.000E+00
 3.600E+04 3.000E-01 7.800E+00
 3.000E+02 4.000E+01 1.760E+00 3.000E+01
 0.000E+00 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.260E+01 3.850E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 2.900E+07 6.000E+04 2.000E+10
 1.750E+00 2.000E-02 7.100E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 1.630E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 7.500E+01
 1.000E+00 5.800E-07 3.500E-03 1.000E-10
water.dat
    1
Cs-137   1.370E+02 3.000E+01 1.600E+01 1.990E+01 1.000E+00 6.800E-12 5.120E-13
```

Exhibit C.20. Sample SOURCE2 input file for ^{137}Cs in a well-in-silo-type facility (*filename*.inp)

```
Sample run of Cs-137 in wells within a silo
      1000    3 0 0   50 0   50 0  200 0   50 0   50
 1.920E+02 5.400E+01
 0.000E+00 1.088E+01 1.200E+01
 0.000E+00 0.000E+00 0.000E+00 0.000E+00 5.810E+00 5.810E+00
 0.000E+00 6.250E-02
 0.000E+00 0.000E+00 0.000E+00 0.000E+00 1.875E-01 6.000E+00
 1.920E+02 1.038E+01
 1.200E+01 7.500E-01 1.200E+01
 5.810E+00 5.810E+00 0.000E+00 0.000E+00 0.000E+00 0.000E+00
 3.600E+04 3.000E-01 7.800E+00
 3.000E+02 4.000E+01 1.760E+00 3.000E+01
 0.000E+00 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.260E+01 3.850E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 2.900E+07 6.000E+04 2.000E+10
 1.750E+00 2.000E-02 7.100E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 1.630E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 2.000E+01 0.000E+00 5.000E+01
 0.000E+00 7.500E+01
 1.000E+01 5.800E-07 3.500E-03 1.000E-10
water.dat
    1
Cs-137   1.370E+02 3.000E+01 1.600E+01 1.990E+01 1.000E+00 6.800E-12 5.120E-13
```

Exhibit C.21. Sample SOURCE2 input file for ^{137}Cs in a trench-type facility (*filename*.inp)

```
Sample run of Cs-137 in a trench
      1000  1 0 0      1 0    1 0   50 0    1 0    1
 2.049E+02 1.959E+02
 1.200E+01 6.000E+00 1.200E+01
 5.810E+00 5.810E+00 0.000E+00 0.000E+00 5.810E+00 5.810E+00
 0.000E+00 0.000E+00
 0.000E+00 0.000E+00 0.000E+00 0.000E+00 0.000E+00 0.000E+00
 3.000E+02 4.000E+01 1.760E+00 3.000E+01
 7.200E+01 1.760E+00 1.760E+00 3.500E-01
 2.400E+00 1.500E-01 1.500E-01 5.000E+03 4.000E-01 1.260E+01 3.850E+02
 1.000E-02 2.110E+00 8.000E+00 6.700E-01
 0.000E+00 0.000E+00 0.000E+00
 1.750E+00 2.000E-02 7.100E-01
 2.100E-03 2.040E-04 8.810E-06 1.000E-03 5.210E-04 1.630E-04 2.620E-04 2.620E-04
 2.120E-11 1.820E-11 5.080E-11 1.920E-10 2.100E-10 1.060E-11
 6.750E+00 3.490E+02 1.500E+01
 2.000E-02 1.200E-03 1.200E-03
 0.000E+00 2.000E+01 0.000E+00 0.000E+00
 4.500E+01 5.800E-07 3.500E-03 3.500E-03
water.dat
    1
Cs-137   1.370E+02 3.000E+01 1.600E+01 1.990E+01 1.000E+00 6.800E-12 5.120E-13
```

APPENDIX D

COMPUTER CODE LISTINGS FOR SOURCE1 AND SOURCE2

D. COMPUTER CODE LISTINGS FOR SOURCE1 AND SOURCE2

Computer code listings for Version 2.0 of SOURCE1 and SOURCE2 are provided in Exhibits D.1 and D.2, respectively. An illustration of the code hierarchy for the SOURCE codes is presented in Figs. 2.5 and 2.6 of Sect. 2.

Exhibit D.1. Computer code listing for SOURCE1

```
      program source1

c.....SOURCE1 Version 2.0.

c.....Reference: Icenhour, A. S. and M. L. Tharp, "User's Manual for
c.....the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating
c.....Low-Level Radioactive Waste Disposal Facility Source Terms
c.....(Version 2.0)," ORNL/TM-13035, Oak Ridge National Laboratory,
c.....Oak Ridge, TN, 1996.

c-----------------------------------------------------------------------
c     Program driver
c
c     Calls: input, output, sarl, concrete, roof, walls, floor,
c            pad, fcrack, leach
c-----------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,w1aper(3,2),w1frac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      data fcask/0./

c.....Read input data and perform preliminary calculations for simulation.

      call input(0,nyears)

c.....Opens the files which have been selected to provide summaries
c.....of the simulation results.

      call output(fcask,0,nyears)

c.....Start annual loop.

      do 100 iyear=1,nyears

c.....Read years when inventories were disposed and associated inventories
c.....and updates water infiltration values.

         call input(iyear,nyears)

c.....   if (mod(iyear,10) .eq. 0) write(*,*) 'year: ',iyear

c.....Calculate time-dependent properties of reinforced concrete.

         time = iyear*365.
         csstrn = amax1(((time-28.)/(time+7.))*6.7e-4,0.0)
         crpcof = 5.83e-1*(time**0.6/(10.+time**0.6))

c.....Perform structural analysis for model simulation.

         if (iyear .eq. 1) call sarl

c.....Perform concrete deterioration analysis. Analysis results are
c.....used in cracking analysis until all casks have at least one
c.....cracked or failed member. Degradation analysis is continued for
c.....the entire simulation.

         call concrete(iyear)
```

```
c.....Perform analysis of cracking due to shear, compression, corrosion,
c.....and loading for roof, walls, and floor in succession. Cracking
c.....analysis is peformed until all casks have at least one cracked
c.....member.

         if (fcask .lt. 1.) then

            if (cmthk(1) .gt. 0.) call roof(iyear)
            if (cmthk(2) .gt. 0.) call walls(iyear)
            if (cmthk(3) .gt. 0.) call floor(iyear)

         endif

         if(nmember .eq. 4 .and. padcrk .eq. 0.) call pad(iyear)

c.....Determine proportion of casks that have cracked; update advective and
c.....diffusive leach rates; provide output at user-specified edit frequency.

         if (fcask .lt. 1.) call fcrack(fcask)

         call leach(fcask,iyear,nyears)

         call output(fcask,iyear,nyears)

  100 continue

      stop
      end

      block data sorbd
      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/miscel/acoef,bcoef,dpm(12)

      data cmthk/4*0./
      data attk,icrack/4*1.,3*0/
      data icl,ico2,icrflg/9*0/
      data ispl/3*0/
      data dpm/31.,28.25,31.,30.,31.,30.,31.,31.,30.,31.,30.,31./
      data annprc,noclx,nocly,ovrhng/0.,1,1,0./

      end

      subroutine caoh(iyear)

c-----------------------------------------------------------------------------
c     Called by concrete
c
c     Calculates loss of concrete strength and reduction in pH of concrete
c     due to leaching of Ca(OH)2.
c
c     Calls: derf, derfc
c-----------------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
```

```
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      dimension tlch1(4),tlch2(4)

      real*8 az1,az2,derf

      data tlch1,tlch2/4*0.,4*0./

c.....Calculate retardation factor for Ca(OH)2 leaching by diffusion and
c.....initialize Ca content in C-S-H system.

      if (iyear .eq. 1) then

         rf = 1.+ccdns*cacon/amin1(casol,cap)/ccpor

c.....Calculate Langelier or calcium carbonate saturation index and ionic
c.....concentrations of Ca(OH)2, Mg++, and CO3-.

         xmg2 = amin1(xmg2,xmgsol)
         co3 = amin1(co3,crbsol)
         cap = amin1(cap,casol)
         casum = xmg2+co3+cap
         pk = 2.268712-1.122e-2*temp+3.91e-5*temp**2+1.007e-3*tds -
     #        6.3e-7*tds**2
         phs = pk+alog10(1/cagw)+alog10(1/co3)
         xli = phgw - phs

      endif

c.....If initial pH is greater than 12.5, calculate rate of loss of NaOH
c.....and KOH and consequent decline in pH. Pore liquid and solid con-
c.....centrations are assumed to be equal.

      do 300 l=1,nmember

         if(ph(l) .gt. 12.5) then

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by advection.

            tlch1(l) = amin1(1.,tlch1(l)+annprc/2.54/ommthk(l))
            xlch1 = 1.-tlch1(l)

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by diffusion.

            xlch2 = 0.
            xthk = ommthk(l)/39.37

            do 100 i=0,10

               az1 = (i*xthk/21.+xthk/2.)/(2.*sqrt(dfalk*iyear*3.15e7))
               az2 = (i*xthk/21.-xthk/2.)/(2.*sqrt(dfalk*iyear*3.15e7))
               ft = 0.5*(derf(az1)-derf(az2))

               if (i .eq. 0) then

                  xlch2 = xlch2+ft/21.

               else

                  xlch2 = xlch2+2.*ft/21.

               endif

  100       continue
```

```
c.....Determine total amount of alkalis lost from concrete.

            alklch = 2.-xlch1-xlch2
            ph(1) = amax1(12.5,phbeg-(phbeg-12.5)*alklch)

         else

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by advection. Groundwater leaching is assumed only if
c.....the Langelier index is negative, indicating the water is capable
c.....of dissolving Ca(OH)2.

            if (xli .lt. 0.) then

               tlch2(1) = amin1(1.,tlch2(1)+annprc/2.54*casum/
     #                    (ommthk(1)*cacon))
               xlch3 = 1.-tlch2(1)

            else

               xlch3 = 1.

            endif

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by diffusion.

            xlch4 = 0.
            xthk = ommthk(1)/39.37

            do 200 i=0,10

               az1 = (i*xthk/21.+xthk/2.)/(2.*
     #               sqrt(dfcaoh/rf*iyear*3.15e7))
               az2 = (i*xthk/21.-xthk/2.)/(2.*
     #               sqrt(dfcaoh/rf*iyear*3.15e7))
               ft = 0.5*(derf(az1)-derf(az2))

               if (i .eq. 0) then

                  xlch4 = xlch4+ft/21.

               else

                  xlch4 = xlch4+2.*ft/21.

               endif

  200       continue

c.....Determine total amount of Ca lost from concrete.

            calch = 2.-xlch3-xlch4

c.....Adjust Ca concentration and recalculate Ca:Si ratio.

            ca = cacon*(1.-calch)
            ca_si = ca/si

c.....Calculate average pH for concrete as a function of Ca:Si following
c.....the loss of NaOH and KOH.

            ph(1) = amin1(12.5,8.83533+3.143848*ca_si-0.6617*ca_si**2)

c.....Calculate equivalent depth of Ca(OH)2 loss and loss in strength for
c.....roof, floor, and internal and external walls.

            x = calch

            if (cmthk(1) .gt. 0.) then

               attk(1) = amax1(0.,1.-0.015*x/0.01)
               if (attk(1) .eq. 0.) cmthk(1) = 0.
```

```
           else

              attk(1) = 0.

           endif

        endif

  300 continue

      return
      end

      subroutine ccrack(i,iyear)

c-----------------------------------------------------------------------
c     Called by: sourcel
c
c     Calculates cracking due to corrosion of reinforcing steel.
c
c     Calls: none
c-----------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod

      data stlpsn/.30/

      time = 365.*iyear
      comstr = amin1(time/(cfa+cfb*time)*com28d*attk(i),constr*attk(i))
      cdtstr = 4.*sqrt(comstr)
      crpcof = 5.83e-1*(time**0.6/(10.+time**0.6))
      csstrn = amax1(((time-28.)/(time+7.))*6.7e-4,0.0)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      tmp = amin1(tencvx(i),tencvy(i),comcvx(i),comcvy(i))+ostlrd(i)
      if (tmp .le. stlrad(i)+stlcor(i)) return

c.....Calculate internal pressure due to corrosion.

      pstl = (stlrad(i)+stlcor(i)-ostlrd(i))/ostlrd(i)*1./
     #       ((1.-stlpsn)/stlmod+((1.-conpsn)*ostlrd(i)**2+
     #       (1.+conpsn)*tmp**2)/(conmod*(tmp**2-ostlrd(i)**2)))

c.....Calculate maximum stress.

      conrad = stlrad(i)+stlcor(i)-pstl*ostlrd(i)*(1.-stlpsn)/stlmod
      ctstrs = 4.*pstl*conrad**2/(tmp**2-conrad**2)
      ststrs = pstl*(tmp**2+conrad**2)/(tmp**2-conrad**2)
      xzero = amax1(0.5*stlspc(i),sqrt(tmp**2-ostlrd(i)**2))

c.....Determine whether spalling has occurred. if it has, all of the
c.....concrete cover is assumed to be destroyed and all steel is exposed.

      if (ctstrs.gt.cdtstr .and. ststrs.gt.cdtstr .and.
     #    ctstrs.gt.ststrs) then

         ispl(i) = 1
         cmthk(i) = 0.
         tencvx(i) = 0.
         tencvy(i) = 0.
         comcvx(i) = 0.
         comcvy(i) = 0.
```

```
c.....Cracking extends through concrete cover to steel along steel members.

      elseif (ctstrs .gt. cdtstr) then

         crfrcd(i) = cmthk(i)-2.*stlrad(i)
         crfrcw(i) = 2.*xzero*(cdtstr/conmod+csstrn)
         crfrcs(i) = stlspc(i)
         crfrac(i) = crfrcw(i)*crfrcd(i)*(clwid+cllth)*(clwid+
     #               cllth)/crfrcs(i)

c.....Calculate potential for cracking through concrete cover versus
c.....internal cracking only (i.e., not through the entire concrete cover).

      elseif (ststrs .gt. cdtstr) then

         rrr = sqrt((conrad**2*tmp**2*pstl)/(cdtstr*(tmp**2-
     #         conrad**2)-conrad**2*pstl))

         if (rrr-conrad .ge. 0.5*(tmp-conrad)) then

            crfrcd(i) = cmthk(i)-2.*stlrad(i)
            crfrcw(i) = 2.*xzero*(cdtstr/conmod+csstrn)
            crfrcs(i) = stlspc(i)
            crfrac(i) = crfrcw(i)*crfrcd(i)*(clwid+cllth)*(clwid+
     #                  cllth)/crfrcs(i)

         endif

      endif

      return
      end

      subroutine concrete(iyear)

c-----------------------------------------------------------------------
c     Called by: sourcel
c
c     Calculates degradation of concrete with time.
c
c     Calls: caoh, corrode, sulfate
c-----------------------------------------------------------------------

      common/failure/cftl,dcft,eftl,deft

c.....Calculate loss of concrete strength and changes in pH of concrete
c.....structure due to leaching of Ca(OH)2.

      call caoh(iyear)

c.....Calculate loss of concrete due to sulfate attack.

      call sulfate(iyear)

c.....Calculate corrosion of steel reinforcement.

      call corrode(iyear)

      return
      end

      subroutine corrode(iyear)

c-----------------------------------------------------------------------
c     Called by concrete
c
c     Calculates rate of corrosion of steel reinforcement in concrete
c
c     Calls: none
c-----------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
```

```
     #      otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #      stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #      dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #      crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #      icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #      slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),w1frac(3,2),
     #      w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #      co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #      yngmod
      common/failure/cftl,dcft,eftl,deft
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      real*8 az,derf

      dimension corvol(3)

      data crbcof,eff,pi/0.,0.,3.141592653589793/

c.....Calculate failure function of epoxy coating on steel reinforcement.
c.....If no failure has occurred, corrosion does not occur.

      if (deft .gt. 0) then

         aux = float(iyear)

         if(aux .gt. eftl) then

            eff = aminl(1.,(aux-eftl)/deft)

         else

            return

         endif

      else

         eff = 1.

      endif

c.....Determine depth of concrete carbonation based on Crank formulation.

      if(crbcof .eq. 0.) then

         dhydr = 0.4 + 0.5*wcr
         ccon = ccon*dhydr
         c1 = co2 + ccon
         crbcof = sqrt((c1/ccon - 1.)*4.*dfco2*3.15e7/sqrt(pi))

      endif

      dpcrb = crbcof*sqrt(float(iyear))

c.....Check carbonation depth and concrete cover thickness for structural
c.....components.

      do 100 i=1,nmember

         tmp = (aminl(comcvx(i),tencvx(i),comcvy(i),tencvy(i))+
     #         ostlrd(i))/39.37

         if(dpcrb.ge.tmp .and. icrflg(i).eq.0 .and. eff.gt.0.) then

            ico2(i) = iyear
            icrflg(i) = 1

         endif

c.....Calculate [OH-] in pore solution based on concrete pH.
```

```
         poh = 14.-ph(i)
         oh = 1./10.**poh

c.....Calculate chloride ion concentration at steel reinforcement.  If
c.....ratio of chloride concentration to hydroxide ion concentration
c.....exceeds 0.61 corrosion is initiated (Hausman).

         if(icrflg(i) .eq. 0) then

            tmp = (amin1(comcvx(i),tencvx(i),comcvy(i),tencvy(i))+
     #            ostlrd(i))/39.37
            az = tmp/(2.*dsqrt(dble(dfcl*iyear*3.15e7)))
            clstl = cl-(cl-clcon)*derf(az)

            if(clstl/oh.ge.0.61 .and. eff.gt.0.) then

               icl(i) = iyear
               icrflg(i) = 1

            endif

         endif

c.....Upon de-passivation, the rate of corrosion is determined by the
c.....rate of O2 diffusion to steel. If concrete cover is gone, corrosion
c.....is assumed to proceed at a rate of .025 cm/yr. densities of steel
c.....and corrosion product (FeO) are 7.86 and 5.70 g/cm**3, respectively.

         if (icrflg(i).eq.1 .and. stlrad(i).gt.0.) then

            tmp = amin1(comcvx(i),tencvx(i),comcvy(i),
     #            tencvy(i))+ostlrd(i)

            if(tmp .le. ostlrd(i)) then

               if (stlrad(i) .gt. .01) then

                  volfe = 3.142*(.02*stlrad(i)-.0001)
                  stlrad(i) = stlrad(i)-0.01

               else
                  volfe = 3.142*stlrad(i)**2
                  stlrad(i) = 0.

               endif

            elseif (stlrad(i) .gt. 0.) then

c.....Calculate oxygen flux at the steel reinforcement.

               stlara = eff*stlspc(i)*6.45e-4          !1./39.39**2. (m2)
               o2grd = 39.37e3*o2/tmp                  !1000.*39.37 (mole/m4)
               o2flux = dfo2*stlara*o2grd*3.15e7       !(mole/yr)
               xmole = 7.245*stlrad(i)**2       !pi*7.86*2.54**3/55.85 (mole)
               o2flux = amin1(o2flux,0.5*xmole)
               volfe = .867*o2flux                     !2*55.85/(7.86*2.54**3)
               stlrad(i) = sqrt(amax1(0.,
     #                     stlrad(i)**2-volfe/3.142))
            endif

            corvol(i) = corvol(i)+1.77*volfe        !(7.86/5.7)*(71.85/55.85)
            stlcor(i) = sqrt(stlrad(i)**2+
     #                  corvol(i)/3.142)-stlrad(i)

         endif

  100 continue

      return
      end

      function derf(z)
```

```
c-----------------------------------------------------------------------------
c     Called by: caoh
c
c     Function used in KOH, NaOH, and Ca(OH)2 diffusion calculations. series
c     expansion for erf(z) found in Abramowitz & Stegun #7.1.5.
c
c     Calls: derfc
c-----------------------------------------------------------------------------

      implicit real*8 (a-h,o-z)

      if(dabs(z).lt.2.) go to 10
      derf=1.-derfc(z)
      go to 50

   10 sum=0.

      if(z.eq.0.) go to 40
      z2=z*z
      zpowr=z
      fac=1.
      n2p1=1
      term=z

      do 20 i=1,30

         sum=sum+term
         zpowr=-zpowr*z2
         fac=fac*i
         n2p1=n2p1+2
         term=zpowr/(fac*n2p1)
         if(dabs(term/sum).lt.1.e-15) go to 30

   20 continue

   30 sum=sum+term

   40 derf=1.128379167095513*sum

   50 return
      end

      function derfc(z)

c-----------------------------------------------------------------------------
c     Called by: derf
c
c     Function used in KOH, NaOH, and Ca(OH)2 diffusion calculations. based
c     on continued fraction from Abramowitz & Stegun #7.1.14.
c
c     Calls: none
c-----------------------------------------------------------------------------

      implicit real*8 (a-h,o-z)

      if(dabs(z).ge.2.) go to 10
      derfc=1.-derf(z)
      go to 30

   10 xnum=20.
      zab=dabs(z)
      frac=zab

      do 20 i=1,40

         frac=zab+xnum/frac
         xnum=xnum-0.5

   20 continue

      derfc=0.
      if(zab.le.9.3) derfc=dexp(-zab*zab)/(frac*1.772453850905516)
      if(z.lt.0.) derfc=2.-derfc
```

```
   30 return
      end

      subroutine fcrack(fcask)

c-----------------------------------------------------------------------------
c     Called by: sourcel
c
c     Determines the number of casks with one or more cracked members.
c
c     Calls: none
c-----------------------------------------------------------------------------

      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

c.....Determine the number of casks which have at least one cracked
c.....structural member.

      nccask = 0

      do 100 i=1,lyr

         if(rffrac(i).gt.0. .or. flfrac(i).gt.0.) then

            nccask = nccask+numcsk/lyr

         elseif(wlfrac(i,2).gt.0. .or. w2frac(i,2).gt.0.) then

            nccask = nccask+numcsk/lyr

         else

            if(wlfrac(i,1) .gt. 0.) nccask = nccask+numwid*2
            if(w2frac(i,1) .gt. 0.) nccask = nccask+numlth*2
            nccask = min0(nccask,2*(numwid+numlth)-4)

         endif

  100 continue

      fcask = float(nccask)/float(numcsk)

      return
      end

      subroutine floor(iyear)

c-----------------------------------------------------------------------------
c     Called by: sourcel
c
c     Performs cracking analysis for cask floor.
c
c     Calls: ccrack
c-----------------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/flfrac/flfdpx(3,11,11),flfdpy(3,11,11),flfspx(3,11,11),
```

```
     #       flfspy(3,11,11)
      common/moment/rfxmnt(3,11,11),rfymnt(3,11,11),flxmnt(3,11,11),
     #       flymnt(3,11,11),w1xmnt(3,11,11),w2xmnt(3,11,11),
     #       w1ymnt(3,11,11),w2ymnt(3,11,11)
      common/shear/rfxshr(3,11,11),rfyshr(3,11,11),flxshr(3,11,11),
     #       flyshr(3,11,11),w1xshr(3,11,11),w2xshr(3,11,11),
     #       w1yshr(3,11,11),w2yshr(3,11,11)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      data pi/3.141592653589793/
      data strred/.9/

c.....Calculate time-dependent parameters used in cracking analysis. Steel
c.....running parallel to the cask width is x-direction steel; steel
c.....running perpendicular to cask width is y-direction steel.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attk(3),constr*attk(3))
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      fldstx = cmthk(3)-tencvx(3)
      fldsty = cmthk(3)-tencvy(3)
      starcm = 0.
      startn = stlrad(3)**2*pi/stlspc(3)
      cnmnti = cmthk(3)**3./12.
      crkmtf = cnmnti/(0.5*cmthk(3))*rupmod

c.....Calculate ultimate strength for floor.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c1 = .003*stlmod*starcm*comcvx(3)
      c2 = .003*stlmod*starcm*comcvy(3)
      axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
      axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

      if(axisn1 .le. comcvx(3)) then

         cmblk = startn*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,strred*stlyld*startn*(fldstx-cmblk/2.))

      else

         csstrs = (axisn1-comcvx(3))/axisn1*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,strred*(as1*stlyld*(fldstx-cmblk/2.)+
     #            starcm*csstrs*(fldstx-comcvx(3))))

      endif

      if(axisn2 .le. comcvy(3)) then

         cmblk = startn*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,strred*stlyld*startn*(fldsty-cmblk/2.))

      else

         csstrs = (axisn2-comcvy(3))/axisn2*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,strred*(as1*stlyld*(fldsty-cmblk/2.)+
     #            starcm*csstrs*(fldsty-comcvy(3))))

      endif

c.....Calculate cracking moment of inertia for floor for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
```

```
      ccx = comcvx(3)*starcm*(ratmod-1.)-fldstx*ratmod*startn
      ccy = comcvy(3)*starcm*(ratmod-1.)-fldsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3.+starcm*(ratmod-1.)*(axneux-comcvx(3))
     #         **2+ratmod*startn*(fldstx-axneux)**2
      crmtiy = 0.333*axneuy**3.+starcm*(ratmod-1.)*(axneuy-comcvy(3))
     #         **2+ratmod*startn*(fldsty-axneuy)**2

c.....Calculate cracking due to shear for floor for all layers of casks.

      xk = (1.6+2.4*(clwid/cllth-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*fldstx
      shrsty = 1.7*sqrt(comstr)*fldsty

      do 400 j=1,lyr

         krkx = 0
         krky = 0
         flfrac(j) = 0.
         flaper(j) = 0.
         ii = 0

         do 300 k=1,6

            do 200 l=1,6

               frcwdx = 0.
               frcwdy = 0.

               if(flxshr(j,k,l) .ge. shrstx) then

                  if (flxmnt(j,k,l) .gt. 0.) then

                     tmp = amin1(flxshr(j,k,l)/flxmnt(j,k,l)*fldstx,1.)
                     vcr = amin1((1.9*sqrt(comstr)+2500.*startn/fldstx*
     #                     tmp)*fldstx,3.5*sqrt(comstr)*fldstx)

                  else

                     vcr = 3.5*sqrt(comstr)*fldstx

                  endif

                  if(flxshr(j,k,l) .ge. vcr) then

                     sfrcdx = cmthk(3)
                     sfrcwx = 0.013
                     sfrcsx = clwid/10.

                  endif

               else

                  sfrcdx = 0.
                  sfrcwx = 0.
                  sfrcsx = 0.

               endif

c.....Calculate fracture characteristics for floor due to bending.

               if (flxmnt(j,k,l) .ge. crkmtf) then

                  if (tencvx(3).eq.0. .and. flfspx(j,k,l).eq.0.) then

                     q = stlrad(3)**2*1.571/(stlspc(3)*otncvx(3))
                    if (stlrad(3) .lt. 1.e-15) q = ostlrd(3)**2*1.571/
     #                 (stlspc(3)*otncvx(3))
```

```
                  elseif (stlrad(3).lt.1.e-15 .and.
     #                   flfspx(j,k,l).eq.0.) then

                     q = ostlrd(3)**2*1.571/(stlspc(3)*tencvx(3))

                  elseif (tencvx(3) .gt. 0. .and.
     #                   stlrad(3) .ge. 1.e-15) then

                     q = stlrad(3)**2*1.571/(stlspc(3)*tencvx(3))

                  endif

                  if (stlrad(3) .ge. 1.e-15) then

                     frspce = 0.5*xk*sqrt(2.*stlrad(3)*stlspc(3)/q)

                  elseif (stlrad(3).lt.1.e-15 .and.
     #                   flfspx(j,k,l).eq.0.) then

                     frspce = 0.5*xk*sqrt(2.*ostlrd(3)*stlspc(3)/q)

                  endif

                  if (flfspx(j,k,l).eq.0. .or. flfspx(j,k,l) .ge. 2.
     #               *frspce)flfspx(j,k,l) = frspce

               endif

c.....X-moments exceed cracking moment but not ultimate strength of floor.

               if (flxmnt(j,k,l).ge.crkmtf .and.
     #            flxmnt(j,k,l).lt.flustx) then

                  efmntx = (crkmtf/flxmnt(j,k,l))**3.*cnmnti+
     #                     (1.-(crkmtf/flxmnt(j,k,l))**3.)*crmtix
                  strsmx = flxmnt(j,k,l)*axneux/efmntx
                  stltnx = ratmod*flxmnt(j,k,l)*(fldstx-axneux)/efmntx
                  axsnex = fldstx/(stltnx/stlmod+strsmx/conmod)*
     #                     (stltnx/stlmod+csstrn)+tencvx(3)
                  betax = axsnex/(axsnex-tencvx(3))
                  flfdpx(j,k,l) = axsnex
                  frcwdx = flfspx(j,k,l)*(stltnx/stlmod*betax+csstrn)

               endif

c.....X-moments exceed ultimate strength of floor.

               if (flxmnt(j,k,l).ge.flustx.and.
     #            flfdpx(j,k,l).lt.cmthk(3)) then

                  flfdpx(j,k,l) = cmthk(3)
                  frcwdx = amin1((stlyld/stlmod+csstrn)*flfspx(j,k,l),
     #                     3.e-3*flfspx(j,k,l))

               endif

c.....Perform calculations for y (length) direction of floor. Start
c.....with shear cracking calculations.

               if(flyshr(j,k,l) .ge. shrsty) then

                  if (flymnt(j,k,l) .gt. 0.) then

                     tmp = amin1(flyshr(j,k,l)/flymnt(j,k,l)*fldsty,1.)
                     vcr = amin1((1.9*sqrt(comstr)+2500.*startn/
     #                     fldsty*tmp)*fldsty,3.5*sqrt(comstr)*fldsty)

                  else

                     vcr = 3.5*sqrt(comstr)*fldsty

                  endif
```

```
                  if(flyshr(j,k,l) .ge. vcr) then

                     sfrcdy = cmthk(3)
                     sfrcwy = 0.013
                     sfrcsy = cllth/10.

                  endif

               else

                  sfrcdy = 0.
                  sfrcwy = 0.
                  sfrcsy = 0.

               endif

c.....Calculate fracture characteristics for y (length) direction.

               if (flymnt(j,k,l) .ge. crkmtf) then

                  if (tencvy(3).eq.0. .and. flfspy(j,k,l).eq.0.) then

                     q = stlrad(3)**2*1.571/(stlspc(3)*otncvy(3))
                     if (stlrad(3) .lt. 1.e-15) q = ostlrd(3)**2*1.571/
     #                  (stlspc(3)*otncvy(3))

                  elseif (stlrad(3).lt.1.e-15 .and.
     #                   flfspy(j,k,l).eq.0.) then

                     q = ostlrd(3)**2*1.571/(stlspc(3)*tencvy(3))

                  elseif (tencvy(3).gt.0. .and.
     #                   stlrad(3).ge.1.e-15) then

                     q = stlrad(3)**2*1.571/(stlspc(3)*tencvy(3))

                  endif

                  if (stlrad(3) .ge. 1.e-15) then

                     frspce = 0.5*xk*sqrt(2.*stlrad(3)*stlspc(3)/q)

                  elseif (stlrad(3).lt.1.e-15 .and.
     #                   flfspy(j,k,l).eq.0.) then

                     frspce = 0.5*xk*sqrt(2.*ostlrd(3)*stlspc(3)/q)

                  endif

                  if (flfspy(j,k,l).eq.0. .or.
     #               flfspy(j,k,l).ge.2.*frspce)
     #               flfspy(j,k,l) = frspce

               endif

c.....Y-moments exceed cracking moment but not ultimate strength of floor.

               if (flymnt(j,k,l).ge.crkmtf .and.
     #            flymnt(j,k,l).lt.flusty) then

                  efmnty = (crkmtf/flymnt(j,k,l))**3.*cnmnti+(1.-
     #                     (crkmtf/flymnt(j,k,l))**3.)*crmtiy
                  strsmy = flymnt(j,k,l)*axneuy/efmnty
                  stltny = ratmod*flymnt(j,k,l)*(fldsty-axneuy)/efmnty
                  axsney = fldsty/(stltny/stlmod+strsmy/conmod)*
     #                     (stltny/stlmod+csstrn)+tencvy(3)
                  betay = axsney/(axsney-tencvy(3))
                          flfdpy(j,k,l) = axsney
                  frcwdy = flfspy(j,k,l)*(stltny/stlmod*betay+csstrn)

               endif

c.....Y-moments exceed ultimate strength of floor.
```

```
               if (flymnt(j,k,l).ge.flusty.and.
     #            flfdpy(j,k,l).lt.cmthk(3)) then

                  flfdpy(j,k,l) = cmthk(3)
                  frcwdy = amin1((stlyld/stlmod+csstrn)*flfspy(j,k,l),
     #                     3.e-3*flfspy(j,k,l))

               endif

c.....Calculate cracking due to corrosion once it begins.

               if (icrflg(3).eq.1 .and. (j+k+l).eq.3)
     #            call ccrack(3,iyear)

c.....Calculate average crack characteristics for floor.

               if (cmthk(3) .eq. 0.) then

                  do 100 m=1,lyr

                     flaper(m) = 0.
                     flfrac(m) = 0.

  100             continue
                  return

               else

                  fmax = .75*cmthk(3)
                  depth = amax1(flfdpx(j,k,l),sfrcdx,crfrcd(3))

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     krkx = krkx+1
                     if (flfspx(j,k,l) .gt. 0.)
     #                  tmp1 = clwid/10./flfspx(j,k,l)
                     if (crfrcs(3) .gt. 0.) tmp2 = cllth/10./crfrcs(3)
                     if (sfrcsx .gt. 0.) tmp3 = clwid/10./sfrcsx
                     tmp = tmp1+tmp2+tmp3
                     flaper(j) = flaper(j)+(frcwdx*tmp1+crfrcw(3)*tmp2+
     #                           sfrcwx*tmp3)/tmp
                     flfrac(j) = flfrac(j)+2.*cmthk(3)*cllth/10.*
     #                           (frcwdx*tmp1+sfrcwx*tmp3)

                  endif

                  depth = amax1(flfdpy(j,k,l),sfrcdy,crfrcd(3))

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     krky = krky+1
                     if (flfspy(j,k,l) .gt. 0.)
     #                  tmp1 = cllth/10./flfspy(j,k,l)
                     if (crfrcs(3) .gt. 0.) tmp2 = clwid/10./crfrcs(3)
                     if (sfrcsy .gt. 0.) tmp3 = cllth/10./sfrcsy
                     tmp = tmp1+tmp2+tmp3
                     flaper(j) = flaper(j)+(frcwdy*tmp1+crfrcw(3)*tmp2+
     #                           sfrcwy*tmp3)/tmp
                     flfrac(j) = flfrac(j)+2.*cmthk(3)*clwid/10.*
     #                           (frcwdy*tmp1+sfrcwy*tmp3)

                  endif

               endif

  200       continue
```

```
  300    continue

         if (flfrac(j) .gt. 0.) icrack(3) = 1
         flfrac(j) = flfrac(j)+crfrac(3)

  400 continue

      do 500 j=1,lyr

         if (flfrac(j) .gt. 0.) then

            flfrac(j) = flfrac(j)/(cmthk(3)*clwid*cllth)
            flaper(j) = flaper(j)/(krkx+krky)*2.54

         endif

  500 continue

      return
      end

      subroutine flothru(d1,d2,flam,iyear,m,qzero,rel)

c-------------------------------------------------------------------------------
c
c        computes the monthly release r(t) of a radioactive
c        contaminant released from a slab of half-thickness a,
c        through a layer of thickness b - a, remaining at time t,
c        given an initial concentration of zero in the
c        outer layer and an initial amount of q0 in the inner
c        layer. subscripts 1 and 2 refer to the inner and outer layers.
c
c        matl   - label for material (20 characters maximum)
c        d1, d2 - diffusion coefficients (cm**2/sec)
c        a      - inner layer half-thickness (cm)
c        thk    - outer layer thickness (cm)
c        t      - time (sec)
c        flam   - decay constant (sec**-1)
c        qzero  - initial amount in inner layer (gm)
c        rel    - monthly release (gm)
c        v      - contains f (x), n = -1, 0, ..., 81
c                           n
c                 such that
c                  n
c                 i erfc  (x) = (2/sqrt(pi))*(exp(-x**2))*v(n)/v(-1)
c
c     Reference: Icenhour, A. S. and M. L. Tharp, "User's Manual for
c     the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating
c     Low-Level Radioactive Waste Disposal Facility Source Terms
c     (Version 2.0)," ORNL/TM-13035, Oak Ridge National Laboratory,
c     Oak Ridge, TN, 1996.
c-------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      parameter (maxnuc = 10)
      parameter (r12=1.d0/12.d0)

      common/miscel/acoef,bcoef,dpm(12)

      dimension v(-1:81), ff(30), fx2(30), save(3), rel(maxnuc,12)

      real*4 acoef,bcoef,dpm,d1,d2,flam,rel

      data tosrpi/1.128379167095513d0/
      data pi/3.141592653589793d0/
      data secpyr/3.15576d7/

      a = acoef
      thk = bcoef
      xkap = dsqrt (dble(d2/d1))
      tk = xkap + xkap
      alfa = (thk)/(xkap*a)
```

```
      apk = alfa + xkap
      b = a + thk
      boa = b/a

c.....Compute first 30 roots of transcendental equation.

      c1 = ((alfa*xkap + 3.d0)*alfa + (xkap*3.d0))*alfa + 1.d0
      c2 = ((alfa + (xkap*3.d0))*alfa + 3.d0)*alfa + xkap
      gam = dsqrt(((alfa**2 + 1.d0)*xkap + alfa + alfa)/xkap)
      x1 = 0.5d0*pi/gam
      x = x1

      do 100 i=1,30

         n = 0

   50    continue

         call fxcal (x, alfa, xkap, f, fp)
         x2 = x - f/fp

         if (abs ((x2 - x)/x2) .gt. 5.d-9) then

            n = n + 1
            x = x2
            if (n .le. 20) go to 50
            write (*, '(1x, a)') 'not converged after 20 iterations'

         endif

         fx2(i) = d1*(x/a)**2
         ax = alfa*x
         c = cos(x)
         s = sin(x)
         ca = cos(ax)
         sa = sin(ax)
         ff(i) = s/((apk*ca*s + boa*sa*c)*x**2)
         f3 = c1*c*sa + c2*s*ca
         gam = -f3/fp

         if (gam .lt. 0.d0) then

            write(*,*) 'fp and f3 have same sign'
            return

         endif

         x = x + pi/dsqrt(gam)

  100 continue

c.....Set a few constants.

      ropk = 1.d0/(1.d0 + xkap)
      c3 = ropk + ropk
      fac1 = (c3 + c3)/a
      fxk = (xkap - 1.d0)*ropk
      resold = 0.d0

c.....Compute monthly releases for current year.

      do 500 n=iyear,iyear

         yr = float (n - 1)

         do 400 mo=1,12

            decay = 0.d0
            t = (yr + r12*float(mo))*secpyr
            t0 = (r12*float(mo))*secpyr
            arg = log(2.d0)/flam*t0
            decay = exp(-arg)
            arg1 = a/dsqrt(d1*t)
```

```
           if (arg1 .lt. 4.d0) then

c.....Sum the series.

              sum = 0.d0

              do 200 k=1,30

                 exx = 0.d0
                 arg = fx2(k)*t
                 if (arg .le. 80.d0) exx = exp (-arg)
                 trm = ff(k)*exx
                 sum = sum + trm

                 if (abs(ff(k)/sum) .gt. 5.d-9) then

                    if (abs(trm/sum) .lt. 5.d-9) go to 210

                 endif

  200         continue

              write(*,*) 'series not converged'

          return

  210         continue

              res = 1.d0 - tk*sum

           else

c.....Use the ierfc series.

              arg2 = 0.5d0*alfa*arg1
              sum = 0.d0
              extold = 0.d0
              sign = 1.d0
              odd = 1.d0
              fxz = 1.d0
              l = 0

              do 300 k=1,200

                 arg = odd*arg2

                 if (arg .le. 1.d0) then

                    res2 = sxierfc (arg)

                 else

                    res2 = 0.d0

                    if (arg .le. 10.d0) then

                       call ierfc (arg, v, 1, 5.d-9)
                       res2 = tosrpi*v(1)*exp(-arg**2)/v(-1)

                    endif

                 endif

                 if (res2 .eq. 0.d0) go to 310

                 trm = fxz*res2
                 sum = sum + sign*trm
                 l = l + 1
                 save(l) = sum

                 if (l .eq. 3) then

c.....Aitken Delta-Squared extrapolation:.
```

```
                     d21 = save(2) - save(1)
                     d32 = save(3) - save(2)
                     ext = save(3) + d32**2/(d21 - d32)
                     1 = 0

                     if (extold .ne. 0.d0) then

                        if (abs(1.d0 - extold/ext) .lt. 5.d-9) then

                           sum = ext
                           go to 310

                        endif

                     endif

                    extold = ext

                  endif

                  if (sum .ne. 0.d0) then

                     if (abs(trm/sum) .lt. 5.d-9) go to 310

                  endif

                  odd = odd + 2.d0
                  fxz = fxz*fxk
                  sign = -sign

  300          continue

               write (*, '(1x, a, 1p, 2e13.5)') 'trm, sum: ', trm, sum
               write(*,*) 'ierfc series not converged'

               return

  310       continue

            res = facl*dsqrt(d2*t)*sum

            endif

            rel(m,mo) = qzero*decay*(res - resold)
            resold = res

  400    continue

  500 continue

      return
      end

      subroutine fxcal (x,alfa,xkap,f,fp)

c------------------------------------------------------------------------------
c     Called by: flothru
c     Calls: none
c------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      ax = alfa*x
      c = cos (x)
      s = sin (x)
      ca = cos (ax)
      sa = sin (ax)
      f = xkap*c*ca - s*sa
      fp = -(xkap + alfa)*s*ca - (1.d0 + xkap*alfa)*c*sa

      return
      end
```

```
      subroutine ierfc (x,v,n,tol)

c-------------------------------------------------------------------------------
c     Called by: flothru
c
c     Subroutine used in diffusion leaching calculations (2 december 1991).
c
c     Calls: none
c-------------------------------------------------------------------------------

c     Compute the repeated integrals of the complementary error
c               n
c     function i erfc (x) by backward recurrence and normalization.
c     input parameters:
c       x   - argument
c       n   - maximum value of n
c                                    n
c       tol - relative error in i erfc (x)
c       v   - double precision aray, dimensioned (-1:81) in the
c               calling program
c     output parameters:
c       v   - contains f (x), n = -1, 0, ..., 81
c                         n
c               such that
c                n
c               i erfc(x) = (2/sqrt(pi))*exp(-x**2))*v(n)/v(1)
c     see W. Gautschi, Recursive computation of the repeated
c     integrals of the error function, Mathematics of Computation
c     15, 227-232(1961)
c-------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      common/numb/mmax

      dimension v(-1:81), xt(21)

      xsq = x**2
      l = 0

      do 200 m=21,81,5

         v(m) = 0.d0
         v(m-1) = 10.d0**(-20)
         x2 = x + x
         a = float (m + m)

         do 100 k=m,1,-1

            v(k-2) = a*v(k) + x2*v(k-1)

c.....Watch growth in backward recurrence.  Scale down if needed.
c.....this works for n=1 only.

            if(v(k-2) .gt. 1.d20) then

               if(k.gt.1) then

                  v(k-1)=v(k-1)/v(k-2)
                  v(k-2)=1.d0

               endif

            endif

            a = a - 2.d0

  100    continue

         l = l + 1
         xt(l) = v(n)/v(-1)

         if (l .gt. 1) then
```

```
            if (abs(xt(l)/xt(l-1) - 1.d0) .lt. tol) go to 210

         endif

  200 continue

      write (*, '(1x, a, i2, 1x, a, 1p, e11.3)')
     1      'm = 81 not enough for n =', n, ' x =', x
      m = 81
      write (*, '(1x, 1p, 4e15.7)') (xt(n), n=1,l)

  210 continue

      mmax = m

      return
      end

      subroutine input(iyear,nyears)

c-----------------------------------------------------------------------
c     Called by source1
c
c     Reads and checks input data, prints summary, and performs initial
c     calculations.
c
c     Calls: none
c-----------------------------------------------------------------------

      parameter (maxnuc = 10, maxyr = 9999999)

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,w1aper(3,2),w1frac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/dump/ndump,refyear
      common/failure/cft1,dcft,eft1,deft
      common/files/iprint,fname,iprn1,ifrq1,iprn2,ifrq2,iprn3,ifrq3,
     #       iprn4,ifrq4,iprn5,ifrq5,iprn6,ifrq6,iprn7,ifrq7,
     #       filenam(9)
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/miscel/acoef,bcoef,dpm(12)
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),x1lch(maxnuc),
     #       qcask(maxnuc),rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc)
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/pleach/cumlch(maxnuc),xleach(maxnuc),sladv(maxnuc),
     #       sldif(maxnuc),qcask1(maxnuc),saladv1(maxnuc),
     #       sldif1(maxnuc),qcask2(maxnuc),saladv2(maxnuc),
     #       sldif2(maxnuc)
      common/runid/title
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      dimension ext(9)

      real*8 qcask1,qcask2,xleach
      integer refyear, bgndump
      character*16 fname, filenam*20,ext*4,wat_inp*60
      character*8 nuclid
```

```
      character*80 title

      data iyr2/0/, ndump/1/, refyear/1/,
     #     bgndump/0/
      data ext/'.inp','.con','.lch','.rch','.lat','.sum',
     #         '.h2o','.vt1','.vt2'/

      if(iyear .eq. 0) then

c.....Input filename of input file for the simulation.

         write(*,*) ' Enter first extension of filename for opening'
         write(*,*) ' the input file and output files '
         write(*,*) ' associated with this run. '
         read(*,'(a16)') fname

         do il=1,16

            ilen = il

            if(fname(il:il) .eq. ' ')then

               ilen = ilen-1
               go to 10

            endif

         enddo

   10    continue

c.....Create filenames with extensions for all input and output
c.....files.

         do ifile=1,9

            filenam(ifile) = fname(1:ilen)//ext(ifile)

         enddo

c.....Open input file with name "fname".inp.
c.....Open input data set; read title of simulation and simulation options.

         open(unit=1,file=filenam(1),status='old')

         read(1,'(a80)') title
         read(1,'(2i10,i2,7(i2,i5))') nyears,intctrl,iprint,
     #         iprn1,ifrq1,iprn2,ifrq2,
     #         iprn3,ifrq3,iprn4,ifrq4,iprn5,ifrq5,
     #         iprn6,ifrq6,iprn7,ifrq7

c.....Set default for output files to print every year.

         if(iprn1 .eq. 0 .and. ifrq1 .eq. 0)ifrq1 = 1
         if(iprn2 .eq. 0 .and. ifrq2 .eq. 0)ifrq2 = 1
         if(iprn3 .eq. 0 .and. ifrq3 .eq. 0)ifrq3 = 1
         if(iprn4 .eq. 0 .and. ifrq4 .eq. 0)ifrq4 = 1
         if(iprn5 .eq. 0 .and. ifrq5 .eq. 0)ifrq5 = 1
         if(iprn6 .eq. 0 .and. ifrq6 .eq. 0)ifrq6 = 1
         if(iprn7 .eq. 0 .and. ifrq7 .eq. 0)ifrq7 = 1

c.....Cask and tumulus dimensions and design specifications.

         read(1,'(4i5)') lyr,numwid,numlth,nmember
         read(1,'(3e10.3)') clwid,cllth,clhght
         read(1,'(3e10.3)') (cmthk(i),i=1,3)
         read(1,'(6e10.3)') (tencvx(i),tencvy(i),i=1,3)
         read(1,'(6e10.3)') (stlrad(i),stlspc(i),i=1,3)
         read(1,'(4e10.3)') submod,flangl,sldns,slangl
         read(1,'(4e10.3)') cvrthk,cvrdns,wstdns,wstht

c.....Concrete and steel specifications for vault.
```

```
         read(1,'(7e10.3)') ccdns,ccpor,conpsn,com28d,wcr,phbeg,wtcmnt
         read(1,'(4e10.3)') clcon,ccon,cfa,cfb
         read(1,'(3e10.3)') stlmod,stlyld,yngmod
         read(1,'(3e10.3)') cacon,cap,si

c.....Pad dimensions and concrete and steel specifications.

         if (nmember .eq. 4)
     #       read(1,'(f8.4,8e9.2)')pstlrad,cmthk(4),pstlmod,
     #       pstlyld,pconstr,pstlspc,pbotcov,pwtcmnt,piff


c.....Chemical concentrations, diffusion coefficients, groundwater
c.....properties, and solubilities.

         read(1,'(8e10.3)') cagw,cl,co2,co3,xmg2,o2,so4i,so4o
         read(1,'(6e10.3)') dfalk,dfcaoh,dfcl,dfco2,dfo2,dfso4
         read(1,'(3e10.3)') phgw,tds,temp
         read(1,'(3e10.3)') casol,crbsol,xmgsol

c.....Failure function data for metal containers and epoxy
c.....covering on rebar.

         read(1,'(4e10.3)') cftl,dcft,eftl,deft

c.....Hydrogeological parameters.

         read(1,'(4e10.3)') sitara,slkr,slk,cck

c.....Input name of file containing water seepage values.

         read (1,'(a)') wat_inp
         open(unit=4,file= wat_inp, status = 'old')

c.....Radionuclide-specific data.

         read(1,'(i5)') noncld

         if(noncld .gt. maxnuc)then

            write(*,
     #         '('' The value of the variable noncld is greater'')')
            write(*,
     #         '('' than the value specified for maxnuc on the'')')
            write(*,
     #         '('' parameter statements.  Increase the value of'')')
            write(*,'('' maxnuc.'')')
            stop

         endif

         do 100 k=1,noncld

            read(1,'(a8,7e10.3)') nuclid(k),am(k),hlife(k),sol(k),
     #           xkd(k),qcask(k),dfwst(k),dfcon(k)

            if(qcask(k) .le. 0.)ndump=0

  100    continue

         if(ndump .ne. 1) then

            read(1,'(i10)') refyear
            read(1,'(i10,10e10.3)')
     #           bgndump,(qcask(k),k=1,noncld)
            ndump = bgndump

         endif

c.....Calculate or initialize various parameters for tumulus. Start with
c.....variables dealing with casks, cask dimensions, and concrete pH.

         acoef = 0.5*(clwid-cmthk(2))/39.37
```

```
         bcoef = cmthk(2)/39.37
         constr = com28d/cfb
         numcsk = lyr*numwid*numlth
         wsthk = (clhght-0.5*(cmthk(1)+cmthk(3)))*2.54

         do 200 i=1,3

            comcvx(i) = cmthk(i)-tencvx(i)
            comcvy(i) = cmthk(i)-tencvy(i)
            otncvx(i) = tencvx(i)
            otncvy(i) = tencvy(i)
            ostlrd(i) = stlrad(i)

  200    continue

         do i = 1,nmember

            ommthk(i) = cmthk(i)
            ph(i) = phbeg

         enddo

c.....Calculate uniform loads on roof.

         do 300 i=1,lyr

            xload(i) = 3.61e-2*(cvrthk*cvrdns+cmthk(1)*ccdns)+
     #                 (i-1)*3.61e-2*
     #                 ((clhght-0.5*(cmthk(1)+cmthk(3)))*
     #                 wstdns+(cmthk(1)+ cmthk(3)) * ccdns)

  300    continue

c.....Convert half-life from years to seconds.

         do 600 n=1,noncld

            hlife(n) = hlife(n)*3.15576e7

  600    continue

      endif

c.....Update water values.

      if (iyear .gt. iyr2) then

         read(4,'(2i10,12f5.2)')iyr1, iyr2, (water(i),i=1,12)

c.....Calculate annual percolation rate through intact concrete.

         annprc = 0.

         do mo=1,12

            annprc = annprc+amin1(cck*8.64e4*dpm(mo),water(mo))

         enddo

      endif

      return
      end

      subroutine leach(fcask,iyear,nyears)

c------------------------------------------------------------------------------
c     Called by: source1
c
c     Calculates annual radionuclide releases due to advection and diffusion.
c
c     Calls: flothru, maxlch
c------------------------------------------------------------------------------
```

```
      parameter (maxnuc = 10)

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/failure/cftl,dcft,eftl,deft
      common/flagg/iflagsl(maxnuc),iflags2(maxnuc),iflags(maxnuc)
      common/flows/rflow,sflow
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyrl,iyr2
      common/leachnew/tleachl(maxnuc),tleach2(maxnuc)
      common/miscel/acoef,bcoef,dpm(12)
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qcask(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc)
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/pleach/cumlch(maxnuc),xleach(maxnuc),sladv(maxnuc),
     #       sldif(maxnuc),qcaskl(maxnuc),saladvl(maxnuc),
     #       sldifl(maxnuc),qcask2(maxnuc),saladv2(maxnuc),
     #       sldif2(maxnuc)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      dimension difusl(maxnuc,12),difus2(maxnuc,12)
      dimension rel(maxnuc,12),q(maxnuc)
      dimension iyrcut(maxnuc),dkayyr(maxnuc),dkaymo(maxnuc,12)
      dimension qkl(maxnuc),qk2(maxnuc)
      dimension dkaymon(maxnuc,12)
      dimension xleachl(maxnuc),xleach2(maxnuc)

      real*8 q,qkl,qk2,qcaskl,qcask2,difusl,difus2
      real*8 xleach,xleachl,xleach2,tleachl,tleach2
      real*8 tl,t2,lmbdal(maxnuc),lmbda2(maxnuc),lmbdad(maxnuc)
      character*8 nuclid

      data cff/0./

      save iyrcut,dkayyr,dkaymo
      save fcskold

c.....Define the local vectors iyrcut, dkayyr, dkaymo on first call.
c.....dkaymo(n,mo) - decay constant for first mo months of a year.
c.....dkaymon(n,mo) - monthly decay constant

      if(iyear.eq.1) then

         nyd5pl = nyears/5 + 1
         dcon = -log(2.)*3.15576e7
         ccon = -80./dcon

         do 100 n=1,noncld

            arg=dcon/hlife(n)
            dkayyr(n)=exp(arg)
            arg=arg/365.25
            lmbdad(n)=(-arg)

            do 50 mo=1,12

               dkaymon(n,mo)=exp(arg*dpm(mo))
               sum=0.

               if(mo.eq.12) then

                  dkaymo(n,mo)=1.
```

```
              else

                 do imo=mo+1,12

                    sum=sum+dpm(imo)

                 enddo

                 dkaymo(n,mo)=exp(arg*sum)

              endif

   50       continue

  100    continue

         do n=1,noncld

            qcask1(n) = qcask(n)
            qcask2(n) = 0.

         enddo

         fcskold = 0.

      endif

c.....Calculate inventory in intact (qcask1(n))
c.....and cracked (qcask2(n)) vaults.

      do n=1,noncld

         if(fcask .ne. 0) qcask2(n) = qcask2(n) * (fcskold/fcask) +
     #      qcask1(n) * ((fcask-fcskold)/fcask)

         if(fcask .eq. 1) qcask1(n) = 0.

      enddo

      fcskold = fcask

c.....Calculate failure fraction for steel boxes inside casks to determine
c.....the inventory subject to leaching by diffusion and advection.

      if (dcft .gt. 0.) then

         aux = float(iyear)

         if (aux.gt.cftl) then

            cff = amin1(1.,(aux-cftl)/dcft)

         else

            do 130 n=1,noncld

               qcask1(n)=dkayyr(n)*qcask1(n)
               qcask2(n)=dkayyr(n)*qcask2(n)

  130       continue

            return

         endif

      else

         cff = 1.

      endif

c.....Initialize variables to zero.
```

```
       do 150 n=1,noncld

          sladv(n)=0.
          sldif(n)=0.
          saladv1(n) = 0.

      saladv2(n) = 0.
      sldif1(n) = 0.
      sldif2(n) = 0.

          do mo = 1,12

             difus1(n,mo) = 0.d0
             difus2(n,mo) = 0.d0

          enddo

   150 continue

       rflow = 0.
       sflow = 0.

c.....Set very small inventory values to zero to prevent
c.....numerical problems in the leaching calculations.

       do n = 1,noncld

          if(qcask1(n) .lt. 1.d-25) qcask1(n) = 0.d0
          if(qcask2(n) .lt. 1.d-25) qcask2(n) = 0.d0

       enddo

       do mo = 1,12

          do l = 1,noncld

             rel(l,mo) = 0.

          enddo

       enddo

c.....Begin monthly loop.

       t1=1.

       do 400 mo=1,12

          t2=t1+dpm(mo)-1.
          ttlwat = water(mo)/100.*sitara
          xperc(1) = amin1(cck*8.64e4*dpm(mo),water(mo))
          xperc(2) = amin1(slk*8.64e4*dpm(mo),water(mo))

c.....Calculate lateral and recharge flow components.

          if(water(mo).ne.0)then

             tmp = amin1(1.,slkr*8.64e4*dpm(mo)/water(mo))
             rflow = rflow+tmp*water(mo)
             sflow = sflow+(1.-tmp)*water(mo)

          endif

c         begin nuclide loop

          do 200 l=1,noncld

c.....Calculate inventory available for leaching from intact and cracked casks.

             qk1(l) = qcask1(l)*cff*numcsk*(1.-fcask)
             qk2(l) = qcask2(l)*cff*numcsk*(fcask)

c.....Calculate total inventory available for leaching.
```

```
            q(l) = qkl(l)+ qk2(l)

c.....Calculate leach rate constant for intact and cracked casks.

            lmbdal(l)=xperc(l)/(wsthk*(wstht+wstdns*xkd(l)))
            lmbda2(l)=xperc(2)/(wsthk*(wstht+wstdns*xkd(l)))

c.....Calculate advective releases based on percolation rates through
c.....intact casks.

            tleachl(l) = lmbdal(l)*qkl(l)*exp(-(lmbdal(l)))*
     #                   dkaymon(l,mo)

            if(mo .eq. 1) then

c.....Calculate monthly leach rates due to diffusion for entire year
c.....using the flothru computer code and initialize leach fractions
c.....for recharge and lateral flow components.

               if(q(l) .ne. 0.)
     #            call flothru(dfwst(l),dfcon(l),hlife(l),
     #                 iyear,l,q(l),rel)
               rlch(l) = 0.
               xllch(l) = 0.

            endif

            if(q(l) .ne. 0.d0)
     #         difusl(l,mo) = dble(rel(l,mo)) * qkl(l)/q(l)
            if(q(l) .ne. 0.d0)
     #         difus2(l,mo) = dble(rel(l,mo)) * qk2(l)/q(l)

c.....Sum diffusive and advective releases for intact casks to obtain total
c.....release rate.

            tleachl(l) = tleachl(l)+difusl(l,mo)

c.....Calculate advective releases for cracked casks.

            tleach2(l) = lmbda2(l)*qk2(l)*dexp(-(lmbda2(l)))*
     #                   dkaymon(l,mo)
c.....Sum diffusive and advective releases for cracked casks to obtain total
c.....release rate.

            tleach2(l) = tleach2(l)+difus2(l,mo)

c.....Sum diffusive releases for intact and cracked casks.

  200    continue

c.....Check calculated releases against solubility limits using the total
c.....amount of water passing through intact casks.

      do l = 1,noncld

         q(l) = qkl(l)

      enddo

          call maxlch(q,xperc(l)/100.*sitara,l)

      do l = 1,noncld

c.....Sum advective and diffusive components for intact and cracked casks.

            tleach2(l) = tleach2(l)+tleachl(l)

c.....Sum inventory for intact and cracked casks.

         q(l) = qkl(l)+qk2(l)

      enddo
```

```
c.....Check calculated releases against solubility limits using the total
c.....amount of water passing through casks.

         call maxlch(q,xperc(2)/100.*sitara,2)

     do 300 l = 1,noncld

c.....Calculate total tumulus release and monthly update of inventory.

            if(iflags1(l).eq.0)then

               lmbda1(l)=lmbda1(l)/dpm(mo)
               xleach1(l)=(lmbda1(l)*qk1(l))/(lmbda1(l)+lmbdad(l))
     #                     *(dexp(-t1*(lmbda1(l)+lmbdad(l)))-
     #                    dexp(-t2*(lmbda1(l)+lmbdad(l))))+
     #                    difus1(l,mo)*1.
            else

               xleach1(l)=tleach1(l)*1.

            endif

            if(fcask .ne. 1)
     #         qcask1(l) = dmax1(0.d0,(dkaymon(l,mo)*qcask1(l)-
     #                    xleach1(l)/(numcsk*(1.-fcask))))

            if(iflags2(l).eq.0)then

               lmbda2(l)=lmbda2(l)/dpm(mo)
               xleach2(l)=(lmbda2(l)*qk2(l))
     #                    /(lmbda2(l)+lmbdad(l))
     #                    *(dexp(-t1*(lmbda2(l)+lmbdad(l)))
     #                    -dexp(-t2*(lmbda2(l)+lmbdad(l))))+
     #                    difus2(l,mo)*1.
            else

               xleach2(l)=(tleach2(l)-tleach1(l))*1.

            endif

            if(fcask .ne. 0)
     #         qcask2(l) = dmax1(0.d0,(dkaymon(l,mo)*qcask2(l)-
     #                    xleach2(l)/(numcsk*fcask)))
            xleach(l) = xleach1(l)+xleach2(l)
            tleach2(l) = tleach2(l)-tleach1(l)
            saladv1(l) = saladv1(l) + dkaymo(l,mo) *
     #                   dmax1(0.d0,xleach1(l) - difus1(l,mo))
            saladv2(l) = saladv2(l) + dkaymo(l,mo) *
     #                   dmax1(0.d0,xleach2(l) - difus2(l,mo))
            sldif1(l) = sldif1(l) +
     #                  dkaymo(l,mo)*dmin1(tleach1(l),
     #                  dble(difus1(l,mo)))
            sldif2(l) = sldif2(l) +
     #                  dkaymo(l,mo)*dmin1(tleach2(l),
     #                  dble(difus2(l,mo)))

            if(mo .eq. 12) then

               sladv(l) = saladv1(l) + saladv2(l)
               sldif(l) = sldif1(l) + sldif2(l)

            endif

c.....Partition release into lateral flow and recharge components assuming
c.....same contaminant concentration in each component.  Decay partitioned
c.....releases to end of current year.

            rlch(l) = rlch(l)+xleach(l)*tmp*dkaymo(l,mo)
            if(tmp .lt. 1.) xllch(l) = xllch(l)+xleach(l)*
     #         (1.-tmp)*dkaymo(l,mo)

  300    continue
```

```
         t1=t2+1.
  400 continue
      do 310 l=1,noncld
c.....Calculate adjusted total inventory per cask.
         qcask(l)=qcask1(l) * (1.-fcask) +
     #            qcask2(l) * fcask
c.....Calculate total annual release.
         xleach(l) = rlch(l)+xllch(l)
c.....Determine cumulative amount leached.
         if(iyear.eq.1)then
            cumlch(l)=xleach(l)
         else
            cumlch(l)=cumlch(l)+xleach(l)
         endif
  310 continue
c.....Reset negative diffusion values.
      do n=1,noncld
         if (sldif(n) .lt. 0) sldif(n) = - sldif(n)
         if (sldif1(n) .lt. 0) sldif1(n) = - sldif1(n)
         if (sldif2(n) .lt. 0) sldif2(n) = - sldif2(n)
      enddo
      do n = 1,noncld
         if(fcask .ne. 1.) then
c.....Advection and diffusion for intact vaults (per vault).
            saladv1(n) = saladv1(n)/(numcsk*(1.-fcask))
            sldif1(n) = sldif1(n)/(numcsk*(1.-fcask))
         endif
         if(fcask .ne. 0.) then
c.....Advection and diffusion for cracked vaults (per vault).
            saladv2(n) = saladv2(n)/(numcsk*fcask)
            sldif2(n) = sldif2(n)/(numcsk*fcask)
         endif
      enddo
      if(nmember .eq. 4) then
         if(padcrk .ne. 1. .and. iyear .lt. intctrl) then
            do n = 1,noncld
c.....Attenuate the releases to the environment based on the
c.....functionality of the concrete pad and collection system.
               rlch(n)  = rlch(n)  * (1.-(piff-(piff/intctrl)*iyear))
               xllch(n) = xllch(n) * (1.-(piff-(piff/intctrl)*iyear))
```

```
               enddo

            endif

         endif

         return
         end

         subroutine maxlch(q,ttlwat,icask)
c-------------------------------------------------------------------------------
c        Called by leach
c
c        Calculates solubility limitations on leach rate.
c
c        Calls: none
c-------------------------------------------------------------------------------

         parameter (maxnuc = 10)

         common/flagg/iflags1(maxnuc),iflags2(maxnuc),iflags(maxnuc)
         common/leachnew/tleach1(maxnuc),tleach2(maxnuc)
         common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
        #       dfcon(maxnuc),dfwst(maxnuc),
        #       hlife(maxnuc),xllch(maxnuc),qcask(maxnuc),
        #       rlch(maxnuc),sol(maxnuc),xkd(maxnuc)

         dimension match(maxnuc),q(maxnuc)

         real*8 q,tleach1,tleach2
         character*2 xn(maxnuc)
         character*8 nuclid

         data iflags/maxnuc*0/
         data ifl/0/

c.....Initialize solubility flags to zero.

         do i=1,noncld

            if(icask.eq.1)iflags1(i)=0
            if(icask.eq.2)iflags2(i)=0

         enddo

c.....Find occurrences of multiple isotopes  of the same element.

         if (ifl .eq. 0) then

            do 100 i=1,noncld

               match(i) = 0
               xn(i) = nuclid(i)

  100       continue

            do 300 i=1,noncld

               do 200 j=i,noncld

                  if (match(j).eq.0 .and. xn(j).eq.xn(i)) match(j)=i

  200          continue

  300       continue

            ifl=1

         endif

c.....Calculate maximum leach fraction allowed by solubility.
```

```
      do 600 i=1,noncld

         if(sol(i).eq.0. .or. match(i).lt.i) go to 600
         emole=0.

         do 400 j=1,noncld

            if(match(j) .eq. i) emole=emole+q(j)/am(j)

 400     continue

         if (emole .eq. 0.) go to 600
         xlmax = 1000. * sol(i) * ttlwat / emole

         do 500 j=1,noncld

            if (match(j).eq.i) then

               if(icask .eq. 1)then

                  if(tleachl(j) .gt. (q(j)*xlmax))iflagsl(j)=1
                  tleachl(j) = dminl(dble(q(j)*xlmax),tleachl(j))

               endif

               if(icask .eq. 2)then

                  if(tleach2(j) .gt. (q(j)*xlmax))iflags2(j)=1
              tleach2(j) = dminl(dble(q(j)*xlmax),tleach2(j))

               endif

            endif

            if(iflagsl(j) .eq. 1 .or. iflags2(j) .eq. 1)
     #         iflags(j) = iflags(j) + 1

 500     continue

 600  continue

      return
      end

      subroutine output(fcask,iyear,nyears)

c---------------------------------------------------------------------------
c     Called by main
c
c     Prints results of concrete cracking analyses and leach calculations.
c
c     Calls: none
c---------------------------------------------------------------------------

      parameter (maxnuc = 10)

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/dump/ndump,refyear
      common/failure/cftl,dcft,eftl,deft
      common/files/iprint,fname,iprnl,ifrql,iprn2,ifrq2,iprn3,ifrq3,
```

```
     #       iprn4,ifrq4,iprn5,ifrq5,iprn6,ifrq6,iprn7,ifrq7,
     #       filenam(9)
      common/flagg/iflagsl(maxnuc),iflags2(maxnuc),iflags(maxnuc)
      common/flows/rflow,sflow
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyrl,iyr2
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qcask(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc)
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/pleach/cumlch(maxnuc),xleach(maxnuc),sladv(maxnuc),
     #       sldif(maxnuc),qcaskl(maxnuc),saladvl(maxnuc),
     #       sldifl(maxnuc),qcask2(maxnuc),saladv2(maxnuc),
     #       sldif2(maxnuc)
      common/runid/title
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      dimension name(3)

      real*8 qcaskl,qcask2,xleach
      integer refyear
      character*8 nuclid
      character*10 name
      character*12 fname, filenam*20
      character*80 title
      character*23 labell(maxnuc)/10*'Inventory remaining (g)'/,
     #             label2(maxnuc)*24/10*'Annual leach rate (g/yr)'/,
     #             label3(maxnuc)*25/10*'Cumulative leached (g)'/
      character*21 label4(maxnuc)/10*'Adv leach rate (g/yr)'/,
     #             label5(maxnuc)/10*'Dif leach rate (g/yr)'/
      character*12 label6(maxnuc)/10*'Recharge (g)'/,
     #             label7(maxnuc)/10*'Lateral (g)'/
      character*1  dashs(76)/76*'-'/

      data name/'Cask roof','Cask walls','Cask floor'/
      data ictrl/0/,ipcrk/0/
      data zero/0./

      if(iyear .eq. 0) then

c.....Open file for input data summary and concrete analysis
c.....with name "fname".con.

         if(iprint .eq. 0 .or. iprn3 .eq. 0)
     #      open(unit=7,file=filenam(2),status='new')

         if (iprint .eq. 0) then

            write(7,1000) title
            write(7,1025)
            write(7,1050) nyears
            write(7,1060) ifrq3
            write(7,1125) sitara,tds,temp,phgw,slkr,slk,cck
            write(7,1150) cagw,cl,co3,xmg2,so4o,o2
            write(7,1175) casol,crbsol,xmgsol
            write(7,1200) cacon,cap,ccon,clcon,si
            write(7,1225) com28d*7.04e-2,conpsn,stlmod*7.04e-2,stlyld*
     #           7.04e-2,submod*7.04e-2,yngmod*1.02e-5,wcr,
     #           ccdns,ccpor,wtcmnt,phbeg

            if (nmember .eq. 4)
     #         write(7,1240)pstlrad*2.54,cmthk(4)*2.54,
     #              pstlmod*7.04e-2,pstlyld*7.04e-2,pconstr*7.04e-2,
     #              pstlspc*2.54,pbotcov*2.54,
     #              pwtcmnt,piff
            write(7,1250) dfalk,dfcaoh,dfcl,dfco2,dfo2,dfso4
            write(7,1275) lyr,numwid,numlth,clwid/39.37,
     #           cllth/39.37,clhght/
     #           39.37,(cmthk(i)*2.54,i=1,4),(stlrad(i)*2.54,
     #           i=1,3),(stlspc(i)*2.54,i=1,3)
```

```
          write(7,1285) (tencvx(i)*2.54,tencvy(i)*2.54,i=1,3),
     #          (i,xload(i)* 7.04e-2,i=1,lyr)
          write(7,1300) cvrthk/39.37,cvrdns,flangl,slangl,sldns,
     #          wstdns,wstht
          write(7,1325) cftl,dcft,eftl,deft
          write(7,1350)
          write(7,1375) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #          dfwst(i),dfcon(i),qcask(i),i=1,noncld)
          write(7,1400)

        endif

        if(iprn1 .eq. 0) then

c.....Open file for recharge components with name "fname".rch.

          open(unit=2,file=filenam(4),recl=246,status='new')
          write(2,1000) title
          write(2,'(t1,''Water and total grams in '',
     #          ''recharge component''/)')
          write(2,'(t41,10(a8,12x))')
     #          (nuclid(n),n=1,noncld)
          write(2,'(t8,''Year'',
     #          t19,''Water infiltration (cm)'',t48,10(a,8x)/)')
     #          (label6(n),n=1,noncld)

          if(ndump .gt. refyear) then

            do iz = 1, ndump-refyear

              write(2,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #              (refyear+iz-1),zero,(zero,n=1,noncld)

            enddo

          endif

        endif

        if(iprn2 .eq. 0) then

c.....Open file for lateral flow with name "fname".lat.

          open(unit=3,file=filenam(5),recl=246,status='new')
          write(3,1000) title
          write(3,'(t1,''Water and total grams in '',
     #          ''lateral component''/)')
          write(3,'(t41,10(a8,12x))')
     #          (nuclid(n),n=1,noncld)
          write(3,'(t8,''Year'',
     #          t19,''Water infiltration (cm)'',
     #          t48,10(a,8x)/)')
     #          (label7(n),n=1,noncld)

          if(ndump .gt. refyear) then

            do iz = 1, ndump-refyear

              write(3,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #              (refyear+iz-1),zero,(zero,n=1,noncld)

            enddo

          endif

        endif

        if(iprn4 .eq. 0) then

c.....Open file for output summary information
c.....with name "fname".sum.

          open(unit=10,file=filenam(6),recl=829,status='new')
```

```
          write(10,1000) title
          write(10,2360)
          write(10,2385) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #         dfwst(i),dfcon(i),
     #         qcask(i),i=1,noncld)
          write(10,'(/ '' Output summary total '',
     #         ''for all vaults:''/ )')
          write(10,'(t44,10(a8,70x))')
     #         (nuclid(n),n=1,noncld)
          write(10,'(t14,10(76(a),2x))')(dashs,n=1,noncld)
          write(10,'(t8,''Year'',t14,10(3(a,2x))/)')
     #         (label1(n),label2(n),label3(n),n=1,noncld)

       endif

       if(iprn5 .eq. 0) then

c.....Open file for annual advective loss, diffusive loss, and total loss
c.....with name "fname".lch.

          open(unit=11,file=filenam(3),recl=766,status='new')
          write(11,1000) title
          write(11,2360)
          write(11,2385) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #         dfwst(i),dfcon(i),qcask(i),i=1,noncld)
          write(11,'(/ '' Output summary per total '',
     #         ''number of vaults:''/ )')
          write(11,'(t41,10(a8,64x))')
     #         (nuclid(n),n=1,noncld)
          write(11,'(t14,10(70(a),2x))')
     #         ((dashs(id),id=1,70),n=1,noncld)
          write(11,'(t8,''Year'',
     #         t14,10(3(a,2x))/)')
     #         (label4(n),label5(n),label2(n),n=1,noncld)

       endif

       if(iprint .eq. 0) then

c.....Open file for water infiltration summary information
c.....with name "fname".h2o.

          open(unit=12,file=filenam(7),status='new')

          write(12,'('' Summary of Infiltration Data ''/)')
          write(12,1000) title
          write(12,'(''      Year1      Year2      Jan    Feb'',
     #         ''   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct'',
     #         ''   Nov   Dec'' /)')

       endif

       if(iprn6 .eq. 0) then

c.....Open file for intact vault information
c.....with name "fname".vtl.

          open(unit=14,file=filenam(8),recl=766,status='new')

          write(14,1000) title
          write(14,2360)
          write(14,2385) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #         dfwst(i),dfcon(i),qcask(i),i=1,noncld)
          write(14,'(/ '' Output summary per number of '',
     #         ''intact vaults:''/ )')
          write(14,'(t41,10(a8,64x))')
     #         (nuclid(n),n=1,noncld)
          write(14,'(t14,10(70(a),2x))')
     #         ((dashs(id),id=1,70),n=1,noncld)
          write(14,'(t8,''Year'',
     #         t14,10(3(a,2x))/)')
     #         (label1(n),label4(n),label5(n),n=1,noncld)
```

```
         endif

         if(iprn7 .eq. 0) then

c.....Open file for cracked vault information
c.....with name "fname".vt2.

            open(unit=15,file=filenam(9),recl=766,status='new')

            write(15,1000) title
            write(15,2360)
            write(15,2385) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #           dfwst(i),dfcon(i),qcask(i),i=1,noncld)
            write(15,'(/ '' Output summary per number of '',
     #           ''cracked vaults:''/ )')
            write(15,'(t41,10(a8,64x))')
     #           (nuclid(n),n=1,noncld)
            write(15,'(t14,10(70(a),2x))')
     #           ((dashs(id),id=1,70),n=1,noncld)
            write(15,'(t8,''Year'',
     #           t14,10(3(a,2x))/)')
     #           (label1(n),label4(n),label5(n),n=1,noncld)

         endif

         return

      endif

      if((iprn3 .eq. 0 .and. iyear .eq. 1) .or.
     #   (iprn3 .eq. 0 .and. mod(iyear,ifrq3) .eq. 0)) then

         if(iyear .eq. 1) then

            write(7,2000) ndump

         else

            if(ndump .gt. 1)then

               write(7,2000) ndump+iyear

            else

               write(7,2000)iyear

            endif

         endif

c.....Print concrete degradation.

         write(7,2025)

         write(7,2050) (cmthk(i)*2.54,i=1,4),(amin1(ommthk(i),iyear*
     #        (slfi+slfo))*2.54,i=1,4),(1.-attk(i),i=1,4)

         write(7,2075) (max0(ic1(i),ico2(i)),i=1,3),(stlcor(i)*2.54,
     #        i=1,3),(stlrad(i)*2.54,i=1,3)

c.....Print results of cracking analyses.

         write(7,2100)
         write(7,2125)

         do 100 i=1,3

            if (ispl(i) .eq. 1) then

               write(7,2150) name(i)

            elseif (crfrcd(i) .gt. .75*cmthk(i)) then
```

```
            write(7,2175) name(i)

         else

            write(7,2200) name(i)

         endif

100   continue

      write(7,2225)

      do 200 i=1,3

         if (icrack(i).eq.1) then

            write(7,2175) name(i)

         else

            write(7,2200) name(i)

         endif

200   continue

      if (fcask .gt. 0) then

         if (lyr .eq. 2) write(7,2290)
         if (lyr .eq. 3) write(7,2300)

      endif

      if (rfaper(1)+rfaper(2)+rfaper(3) .gt. 0.) then

         write(7,2325) (rfaper(j),j=1,lyr)
         write(7,2350) (rffrac(j),j=1,lyr)

      endif

      if (w1aper(1,1)+w1aper(2,1)+w1aper(3,1) .gt. 0.) then

         write(7,2375) (w1aper(j,1),j=1,lyr)
         write(7,2350) (w1frac(j,1),j=1,lyr)

      endif

      if (w1aper(1,2)+w1aper(2,2)+w1aper(3,2) .gt. 0.) then

         write(7,2400) (w1aper(j,2),j=1,lyr)
         write(7,2350) (w1frac(j,2),j=1,lyr)

      endif

      if (w2aper(1,1)+w2aper(2,1)+w2aper(3,1) .gt. 0.) then

         write(7,2425) (w2aper(j,1),j=1,lyr)
         write(7,2350) (w2frac(j,1),j=1,lyr)

      endif

      if (w2aper(1,2)+w2aper(2,2)+w2aper(3,2) .gt. 0.) then

         write(7,2450) (w2aper(j,2),j=1,lyr)
         write(7,2350) (w2frac(j,2),j=1,lyr)

      endif

      if (flaper(1)+flaper(2)+flaper(3) .gt. 0.) then

         write(7,2475) (flaper(j),j=1,lyr)
         write(7,2350) (flfrac(j),j=1,lyr)
```

```
         endif

      endif

c.....Output summary values for inventory and leaching.

      if(iprn4 .eq. 0 .and. mod(iyear,ifrq4) .eq. 0)

     #   write(10,'(i10,t18,10(1pe12.4,14x,1pe12.4,14x,
     #        1pe12.4,14x))')
     #        ndump+iyear-1,(qcask(n)*numcsk,xleach(n),
     #        cumlch(n),n=1,noncld)

c.....Output values for leaching.

      if(iprn5 .eq. 0 .and. mod(iyear,ifrq5) .eq. 0)

     #   write(11,'(i10,t17,10(1pe12.4,11x,1pe12.4,13x,
     #        1pe12.4,12x))')
     #        ndump+iyear-1,(sladv(n)/numcsk,sldif(n)/numcsk,
     #        xleach(n)/numcsk,n=1,noncld)

c.....Output values for vault 1 (intact).

      if(iprn6 .eq. 0 .and. mod(iyear,ifrq6) .eq. 0)

     #   write(14,'(i10,t17,10(1pe12.4,11x,1pe12.4,13x,
     #        1pe12.4,12x))')
     #        ndump+iyear-1,(qcask1(n),saladv1(n),
     #        sldif1(n),n=1,noncld)

c.....Output values for vault 2 (cracked).

      if(iprn7 .eq. 0 .and. mod(iyear,ifrq7) .eq. 0)

     #   write(15,'(i10,t17,10(1pe12.4,11x,1pe12.4,13x,
     #        1pe12.4,12x))')
     #        ndump+iyear-1,(qcask2(n),saladv2(n),
     #        sldif2(n),n=1,noncld)

c.....Check to see if solubility constraints have been exceeded.

      if(iyear .ge. nyears)then

         do n = 1,noncld

            if(iflags(n) .ne. 0)then

               if(iprn4 .eq. 0) write(10,702) nuclid(n)
               if(iprn5 .eq. 0) write(11,702) nuclid(n)
  702          format(///' The solubility constraints were exceeded ',
     #               'for ',a)
            else

               if(iprn4 .eq. 0) write(10,703) nuclid(n)
               if(iprn5 .eq. 0) write(11,703) nuclid(n)
  703          format(///' The solubility constraints ',
     #               'were not exceeded for ',a)

            endif

         enddo

      endif

c.....Check for end of institutional control.

      if(iyear .ge. intctrl .and. ictrl .eq. 0) then

         if(iprn4 .eq. 0)
     #      write(10,'('' ***** End of institutional control'',
     #            '' at year '',i10,'' *****'')') ndump+iyear-1
         if(iprn5 .eq. 0)
```

```
     #     write(11,'('' ***** End of institutional control'',
     #          '' at year '',i10,'' *****'')') ndump+iyear-1

         ictrl = 1

      endif

c.....Check if concrete pad has cracked.

      if (padcrk .eq. 1. .and. ipcrk .eq. 0) then

         if(iprn4 .eq. 0)
     #      write(10,'('' ***** Pad has cracked'',
     #          '' at year '',i10,'' *****'')') ndump+iyear-1
         if(iprn5 .eq. 0)
     #      write(11,'('' ***** Pad has cracked'',
     #          '' at year '',i10,'' *****'')') ndump+iyear-1

         ipcrk = 1

      endif

c.....Write annual releases to lateral and recharge component files.

      if(iprn1 .eq. 0 .and. mod(iyear,ifrq1) .eq. 0)
     #   write(2,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #        ndump+iyear-1,rflow,(rlch(n),n=1,noncld)

      if(iprn2 .eq. 0 .and. mod(iyear,ifrq2) .eq. 0)
     #   write(3,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #        ndump+iyear-1,sflow,(xllch(n),n=1,noncld)

      if (iyear .eq. iyr2 .or. iyear .eq. nyears) then

         if(iprint .eq. 0)
     #      write(12,'(1h ,i10,1x,i10,3x,12f6.2)') iyr1, iyr2,
     #           (water(i),i=1,12)

      endif

      return

c------------------------------------------------------------------------------
c     format statements
c------------------------------------------------------------------------------
 1000 format(/80('-')/a80/80('-')/)
 1025 format(/t1,'Input Data Summary:'/t1,19('-'))
 1050 format(/' Simulation length',t50,i10,t61,'years')
 1060 format(' Output edit frequency',t50,i10,t61,'years')
 1125 format(/t6,'Disposal unit area',t50,1pe10.2,' m**2'/t6,'Total ',
     #      'dissolved solids',t50,e10.2,' ppm'/t6,'Groundwater ',
     #      'temperature',t50,e10.2,' deg C'/t6,'Groundwater pH',t50,
     #      e10.2//t6,'Saturated hydraulic conductivity:'/t8,'Recharge',
     #      t50,e10.2,' cm/s'/
     #      t8,'Soil ',
     #      'backfill',t50,e10.2,' cm/s'/t8,'Concrete',t50,e10.2,
     #      ' cm/s')
 1150 format(/' Groundwater constituent concentrations:'/t6,'Ca++',t50,
     #      1pe10.2,' mole/L'/t6,'Cl-',t50,e10.2,' mole/L'/t6,'CO3--',
     #      t50,e10.2,' mole/L'/t6,'Mg++',t50,e10.2,' mole/L'/t6,
     #      'SO4--',t50,e10.2,' mole/L'/t6,'O2',t50,e10.2,' mole/L')
 1175 format(/' Constituent solubilities:'/t6,'Ca(OH)2',t50,1pe10.2,
     #      ' mole/L'/t6,'CO3--',t50,e10.2,' mole/L'/t6,'Mg++',t50,
     #      e10.2,' mole/L')
 1200 format(/' Concrete constituent concentrations:'/t6,'Calcium ',
     #      'concentration in C-S-H system',t50,1pe10.2,' mole/L'/t6,
     #      'Calcium concentration in pore fluid',t50,e10.2,' mole/L'/
     #      t6,'CaO content in cement',t50,e10.2,' mole/L'/t6,'Free ',
     #      'Cl-',t50,e10.2,' mole/L'/t6,'Silica concentration in ',
     #      'C-S-H system',t50,e10.2,' mole/L')
 1225 format(/,' Concrete design specifications:'/t6,'Compressive ',
     #      'strength at 28 days',t50,1pe10.2,' kg/cm**2'/t6,
     #      'Poisson''s ratio of concrete',t50,e10.2/t6,'Modulus of ',
```

```
     #      'elasticity of steel',t50,e10.2,' kg/cm**2'/t6,'Yield ',
     #      'strength of steel',t50,e10.2,' kg/cm**2'/t6,'Modulus of ',
     #      'subgrade reaction',t50,e10.2,' kg/cm**2'/t6,'Young''s ',
     #      'modulus of elasticity',t50,e10.2,' kg/cm**2'/t6,
     #      'Concrete water/cement ratio',t50,e10.2/t6,'Concrete ',
     #      'density',t50,e10.2,' g/cm**3'/t6,'Concrete porosity',t50,
     #      e10.2/t6,'Cement content',t50,e10.2,' kg/m**3'/t6,
     #      'Initial pH',t50,e10.2)
 1240 format(/' Concrete pad failure model parameters:'/
     #       t6,'Radius of pad steel reinforcement',t50,1pe10.2,' cm'/
     #       t6,'Concrete pad thickness',t50,1pe10.2,' cm'/
     #       t6,'Modulus of elasticity of steel reinforcement',
     #       t50,1pe10.2,' kg/cm**2'/
     #       t6,'Yield strength of steel reinforcement',
     #       t50,1pe10.2,' kg/cm**2'/
     #       t6,'Compressive strength of pad concrete',
     #       t50,1pe10.2,' kg/cm**2'/
     #       t6,'Spacing between steel reinforcing rods',
     #       t50,1pe10.2,' cm'/t6,
     #       'Concrete cover thickness from the center '
     #       t50,1pe10.2,' cm'/
     #       t6,'of the bottom ',
     #       'row of steel reinforcing rods '/
     #       t6,'to the bottom of the pad'/
     #       t6,'Weight of pad cement per unit',
     #       t50,1pe10.2,' kg/m**3'/
     #       t6,'volume concrete'/
     #       t6,'Pad initial functionality fraction',
     #       t50,1pe10.2)
 1250 format(/,' Diffusion coefficients in concrete:',/,t6,'NaOH, KOH',
     #      t50,1pe10.2,' m**2/s',/,t6,'Ca(OH)2',t50,e10.2,' m**2/s',
     #      /,t6,'Cl-',t50,e10.2,' m**2/s',/,t6,'CO2',t50,e10.2,
     #      ' m**2/s',/,t6,'O2',t50,e10.2,' m**2/s',/,t6,'SO4--',t50,
     #      e10.2,' m**2/s')
 1275 format(/' Tumulus design specifications:'/t6,'Layers of vaults',
     #      t50,i4/t6,'Number of vaults wide',t50,i4/t6,'Number of ',
     #      'vaults long',t50,i4//t6,'Vault dimensions:'/t8,'Width',
     #      t50,1pe10.2,' m'/t8,'Length',t50,e10.2,' m'/t8,'Height',
     #      t50,e10.2,' m'//t6'Concrete member thickness:'/
     #      t8,'Roof',t50,e10.2,' cm'/t8,'Walls',t50,e10.2,' cm'/t8,
     #      'Floor',t50,e10.2,' cm'/t8,'Pad',t50,e10.2,' cm'//
     #      t6,'Steel reinforcement radius:'/
     #      t8,'Roof',t50,e10.2,' cm'/t8,'Walls',t50,e10.2,' cm'/t8,
     #      'Floor',t50,e10.2,' cm'//t6,'Spacing of steel ',
     #      'reinforcement:'/t8,'Roof',t50,e10.2,' cm'/t8,'Walls',t50,
     #      e10.2,' cm'/t8,'Floor',t50,e10.2,' cm')
 1285 format(/t6'Concrete cover thickness on tension face:'/t8,'Roof:'/
     #      t10,'X-direction',t50,1pe10.2,' cm'/t10,'Y-direction',t50,
     #      e10.2,' cm'/t8,'Walls:'/t10,'Horizontal direction',t50,
     #      e10.2,' cm'/t10,'Vertical direction',t50,e10.2,' cm'/t8,
     #      'Floor:'/t10,'X-direction',t50,e10.2,' cm'/t10,
     #      'Y-direction',t50,e10.2,' cm'//t6,'Static load:',3(/t8,
     #      'Vault layer',i2,t50,e10.2,' kg/cm**2'))
 1300 format(/' Soil and waste properties:'/t6,'Earthen cover ',
     #      'thickness',t50,1pe10.2,' m'/t6,'Earthen cover density',
     #      t50,e10.2,' g/cm**3'/t6,'Friction angle of waste backfill',
     #      t50,e10.2,' deg'/t6,'Friction angle of soil backfill',t50,
     #      e10.2,' deg'/
     #      t6,'Density of soil backfill',t50,e10.2,
     #      ' g/cm**3'/t6,'Waste density',t50,e10.2,' g/cm**3'/t6,
     #      'Relative saturation of waste',t50,e10.2)
 1325 format(/' Concrete and waste package failure rates:'/t6,'Waste',
     #      ' container:'/t8,'Start of failure',t50,1pe10.2,' years'/
     #      t8,'Time to complete failure',t50,e10.2,' years'/t6,
     #      'Epoxy coating:'/t8,'Start of failure',t50,e10.2,' years'/
     #      t8,'Time to complete failure',t50,e10.2,' years')
 1350 format(/,' Nuclide-specific parameters:'//t45,'Diffusion ',
     #      'coefficient'/t2,'Nuclide',t13,'Half-life',t23,'Solubility',
     #      t36,'Waste',t45,'---------------------',t69,'Initial'/t38,
     #      'Kd',t47,'Waste',t57,'Concrete',t68,'Inventory'/t15,'(yr)',
     #      t24,'(mole/L)',t36,'(ml/g)',t46,'(m**2/s)',t57,'(m**2/s)',
     #      t71,'(g)'/t2,'--------',t12,'----------',t23,'----------',
     #      t34,'----------',t45,'----------',t56,'----------',t68,
```

```
     #      '----------')
 1375 format(t3,a8,t11,1pe10.2,t22,e10.2,t33,e10.2,t44,e10.2,t55,e10.2,
     #      t67,e10.2)
 1400 format(//t1,'Output summary:'/t1,15('-'))
 2000 format(/' -----------------------------'/' Annual summary for ',
     #      'year ',i10/' -----------------------------')
 2025 format(/' Concrete Degradation Summary'/)
 2050 format(' Concrete Member Thickness:'/t6,'Roof',t50,1pe10.2,' cm'/
     #      t6,'Walls',t50,e10.2,' cm'/t6,'Floor',t50,e10.2,' cm'/
     #      t6,'Pad',t50,e10.2,' cm'//
     #      ' Concrete loss due to sulfate attack:'/t6,'Roof',t50,
     #      e10.2,' cm'/t6,'Walls',t50,e10.2,' cm'/t6,'Floor',t50,
     #      e10.2,' cm'/t6,'Pad',t50,e10.2,' cm'//
     #      ' Fractional loss of yield strength'/' due ',
     #      'to Ca(OH)2 leaching:'/t6,'Roof',t50,e10.2/t6,'Walls',t50,
     #      e10.2/t6,'Floor',t50,e10.2/t6,'Pad',t50,e10.2)
 2075 format(//' Corrosion results:'//t6,'Time to onset of corrosion:'/
     #      t8,'Roof',t55,i5,' years'/t8,'Walls',t55,i5,' years'/t8,
     #      'Floor',t55,i5,' years'//t6,'Corrosion product layer',
     #      ' thickness:'/t8,'Roof',t50,1pe10.2,' cm'/t8,'Walls',t50,
     #      e10.2,' cm'/t8,'Floor',t50,e10.2,' cm'//t6,'Remaining ',
     #      'steel reinforcement:'/t8,'Roof',t50,e10.2,' cm'/t8,
     #      'Walls',t50,e10.2,' cm'/t8,'Floor',t50,e10.2,' cm')
 2100 format(//' Concrete Cracking Analysis'/)
 2125 format(' Cracking due to corrosion of steel:')
 2150 format(t4,a10,t52,'Spalled out')
 2175 format(t4,a10,t52,'Cracked')
 2200 format(t4,a10,t52,'None')
 2225 format(//' Cracking due to loading and shear:')
 2290 format(//' Concrete crack characteristics:'//t40,'Layer of casks'/
     #      t35,'-------------------------'/t35,'Upper layer   lower ',
     #      'layer'/t35,'-----------   -----------')
 2300 format(//' Concrete crack characteristics:'//t47,'Layer of casks'/
     #      t35,'--------------------------------------'/t35,
     #      'Upper layer  middle layer  lower layer'/t35,'-----------',
     #      '  ------------  -----------')
 2325 format(/t3,'Cask roof'/5x,'Average crack width (cm)',t35,1pe10.2,
     #      2(4x,e10.2))
 2350 format(t6,'Fractional volume of cracks',t35,1pe10.2,2(4x,e10.2))
 2360 format(/,' Nuclide-specific parameters:'//t45,'Diffusion ',
     #      'coefficient'/t2,'Nuclide',t13,'Half-life',t23,'Solubility',
     #      t36,'Waste',t45,'---------------------',t69,'Initial'/t38,
     #      'Kd',t47,'Waste',t57,'Concrete',t68,'Inventory'/t15,'(yr)',
     #      t24,'(mole/L)',t36,'(ml/g)',t46,'(m**2/s)',t57,'(m**2/s)',
     #      t71,'(g)'/t2,'--------',t12,'----------',t23,'----------',
     #      t34,'----------',t45,'----------',t56,'----------',t68,
     #      '----------')
 2375 format(/t3,'Exterior cask wall (X)'/5x,'Average crack width (cm)',
     #      t35,1pe10.2,2(4x,e10.2))
 2385 format(t3,a8,t11,1pe10.2,t22,e10.2,t33,e10.2,t44,e10.2,t55,e10.2,
     #      t67,e10.2)
 2400 format(/t3,'Interior cask wall (X)'/5x,'Average crack width (cm)',
     #      t35,1pe10.2,2(4x,e10.2))
 2425 format(/t3,'Exterior cask wall (Y)'/5x,'Average crack width (cm)',
     #      t35,1pe10.2,2(4x,e10.2))
 2450 format(/t3,'Interior cask wall (Y)'/5x,'Average crack width (cm)',
     #      t35,1pe10.2,2(4x,e10.2))
 2475 format(/t3,'Cask floor'/5x,'Average crack width (cm)',t35,
     #      1pe10.2,2(4x,e10.2))
 2500 format(//,' Radionuclide release rates (g/yr)'/)
 2525 format(3(1x,a8,1pe10.3,8x))

      end

      subroutine pad(iyear)

c-----------------------------------------------------------------------
c    Called by: SOURCE1
c
c    Calculates degradation and cracking of tumulus concrete pad.
c
c    Calls: none
c-----------------------------------------------------------------------
```

```
      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #       pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      data pi/3.1415927/, padcrk/0./

      if(iyear .eq. 1)then

         pstlmod = pstlmod * 6.895e-3
         pstlyld = pstlyld * 6.895e-3
         pconstr = pconstr * 6.895e-3

c.....Calculate reinforcement cross-sectional area per unit length.

         starpd = 2. * pi * (pstlrad)**2 / (pstlspc + 2. * pstlrad)

c.....Calculate yield strain of reinforcement.

         epsy = pstlyld / pstlmod

      endif

      if(pconstr .gt. 30.)then

         betal = 0.85 - 0.08 * (pconstr * attk(4) - 30.) / 10.

      else

         betal = 0.85

      endif

c.....Calculate limiting reinforcement ratio for compressive failure.

      rratiol = .003 / (.003 + epsy) * ((0.85 * betal *
     #          (pconstr * attk(4)) / pstlyld))

c.....Calculate cover thickness at compressive failure (d).

      d = starpd / rratiol

      if((cmthk(4) - pbotcov) .le. d) padcrk = 1

      return
      end

      subroutine roof(iyear)

c-------------------------------------------------------------------------------
c     Called by: sourcel
c
c     Performs cracking analysis for cask roof.
c
c     Calls: ccrack
c-------------------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
```

```
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(3,11,11),rfymnt(3,11,11),flxmnt(3,11,11),
     #       flymnt(3,11,11),w1xmnt(3,11,11),w2xmnt(3,11,11),
     #       w1ymnt(3,11,11),w2ymnt(3,11,11)
      common/rffrac/rffdpx(3,11,11),rffdpy(3,11,11),rffspx(3,11,11),
     #       rffspy(3,11,11)
      common/shear/rfxshr(3,11,11),rfyshr(3,11,11),flxshr(3,11,11),
     #       flyshr(3,11,11),w1xshr(3,11,11),w2xshr(3,11,11),
     #       w1yshr(3,11,11),w2yshr(3,11,11)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

      data pi/3.141592653589793/
      data strred/.9/

c.....Calculate time-dependent parameters used in cracking analysis. In
c.....roof steel running parallel to the cask width is x-direction steel;
c.....steel running perpendicular to cask width is y-direction steel.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attk(1),constr*attk(1))
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      rfdstx = cmthk(1)-tencvx(1)
      rfdsty = cmthk(1)-tencvy(1)
      starcm = 0.
      startn = stlrad(1)**2*pi/stlspc(1)
      cnmnti = cmthk(1)**3./12.
      crkmtr = cnmnti/(0.5*cmthk(1))*rupmod

c.....Calculate ultimate strength for roof.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c1 = .003*stlmod*starcm*comcvx(1)
      c2 = .003*stlmod*starcm*comcvy(1)
      axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
      axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

      if(axisn1 .le. comcvx(1)) then

         cmblk = startn*stlyld/(0.85*comstr)
         rfustx = amax1(crkmtr,strred*stlyld*startn*(rfdstx-cmblk/2.))

      else

         csstrs = (axisn1-comcvx(1))/axisn1*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         rfustx = amax1(crkmtr,strred*(as1*stlyld*(rfdstx-cmblk/2.)+
     #            starcm*csstrs*(rfdstx-comcvx(1))))

      endif

      if(axisn2 .le. comcvy(1)) then

         cmblk = startn*stlyld/(0.85*comstr)
         rfusty = amax1(crkmtr,strred*stlyld*startn*(rfdsty-cmblk/2.))

      else

         csstrs = (axisn2-comcvy(1))/axisn2*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         rfusty = amax1(crkmtr,strred*(as1*stlyld*(rfdsty-cmblk/2.)+
     #            starcm*csstrs*(rfdsty-comcvy(1))))

      endif
```

```
c.....Calculate cracking moment of inertia for roof for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      ccx = comcvx(1)*starcm*(ratmod-1.)-rfdstx*ratmod*startn
      ccy = comcvy(1)*starcm*(ratmod-1.)-rfdsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3.+starcm*(ratmod-1.)*(axneux-comcvx(1))
     #         **2+ratmod*startn*(rfdstx-axneux)**2
      crmtiy = 0.333*axneuy**3.+starcm*(ratmod-1.)*(axneuy-comcvy(1))
     #         **2+ratmod*startn*(rfdsty-axneuy)**2

c.....Calculate cracking due to shear for roof for all layers of casks.

      xk = (1.6+2.4*(clwid/cllth-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*rfdstx
      shrsty = 1.7*sqrt(comstr)*rfdsty

      do 400 j=1,lyr

         krkx = 0
         krky = 0
         rffrac(j) = 0.
         rfaper(j) = 0.

         do 300 k=1,6

            do 200 l=1,6

               frcwdx = 0.
               frcwdy = 0.

               if(rfxshr(j,k,l) .ge. shrstx) then

                  if (rfxmnt(j,k,l) .gt. 0.) then

                     tmp = amin1(rfxshr(j,k,l)/rfxmnt(j,k,l)*rfdstx,1.)
                     vcr = amin1((1.9*sqrt(comstr)+2500.*
     #                     startn/rfdstx*tmp)*
     #                     rfdstx,3.5*sqrt(comstr)*rfdstx)
                  else

                     vcr = 3.5*sqrt(comstr)*rfdstx

                  endif

                  if(rfxshr(j,k,l) .ge. vcr) then

                     sfrcdx = cmthk(1)
                     sfrcwx = 0.013
                     sfrcsx = clwid/10.

                  endif

               else

                  sfrcdx = 0.
                  sfrcwx = 0.
                  sfrcsx = 0.

               endif

c.....Calculate fracture characteristics for roof due to bending.

               if (rfxmnt(j,k,l) .ge. crkmtr) then

                  if (tencvx(1).eq.0. .and. rffspx(j,k,l).eq.0.) then
```

```
                  if (stlrad(1) .lt. 1.e-15) then

                     q = ostlrd(1)**2*1.571/(stlspc(1)*otncvx(1))

                  else

                     q = stlrad(1)**2*1.571/(stlspc(1)*otncvx(1))

                  endif

               elseif (stlrad(1).lt.1.e-15 .and.
     #                 rffspx(j,k,l).eq.0.) then

                  q = ostlrd(1)**2*1.571/(stlspc(1)*tencvx(1))

               elseif (tencvx(1).gt.0. .and.
     #                 stlrad(1).ge.1.e-15) then

                  q = stlrad(1)**2*1.571/(stlspc(1)*tencvx(1))

               endif

               if (stlrad(1) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(1)*stlspc(1)/q)

               elseif (stlrad(1).lt.1.e-15 .and.
     #                 rffspx(j,k,l).eq.0.) then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(1)*stlspc(1)/q)

               endif

               if (rffspx(j,k,l).eq.0. .or.
     #            rffspx(j,k,l).ge.2.*frspce)
     #            rffspx(j,k,l) = frspce

            endif

c.....X-moments exceed cracking moment but not ultimate strength of roof.

            if (rfxmnt(j,k,l).ge.crkmtr .and.
     #         rfxmnt(j,k,l).lt.rfustx) then

               efmntx = (crkmtr/rfxmnt(j,k,l))**3.
     #                  *cnmnti+(1.-(crkmtr/
     #                  rfxmnt(j,k,l))**3.)*crmtix
               strsmx = rfxmnt(j,k,l)*axneux/efmntx
               stltnx = ratmod*rfxmnt(j,k,l)*(rfdstx-axneux)/efmntx
               axsnex = rfdstx/(stltnx/stlmod+strsmx/conmod)*
     #                  (stltnx/stlmod+csstrn)+tencvx(1)
               betax = axsnex/(axsnex-tencvx(1))
               rffdpx(j,k,l) = axsnex
               frcwdx = rffspx(j,k,l)*(stltnx/stlmod*betax+csstrn)

            endif

c.....X-moments exceed ultimate strength of roof.

            if (rfxmnt(j,k,l).ge.rfustx.and.
     #         rffdpx(j,k,l).lt.cmthk(1)) then

               rffdpx(j,k,l) = cmthk(1)
               frcwdx = amin1((stlyld/stlmod+csstrn)*
     #                  rffspx(j,k,l),3.e-3*rffspx(j,k,l))

            endif

c.....Perform calculations for y (length) direction of roof. Start
c.....with shear cracking calculations.

            if(rfyshr(j,k,l) .ge. shrsty) then
```

```
               if (rfymnt(j,k,l) .gt. 0.) then

                  tmp = aminl(rfyshr(j,k,l)/rfymnt(j,k,l)*rfdsty,1.)
                  vcr = aminl((1.9*sqrt(comstr)+2500.*
     #                  startn/rfdsty*tmp)*
     #                  rfdsty,3.5*sqrt(comstr)*rfdsty)

               else

                  vcr = 3.5*sqrt(comstr)*rfdsty

               endif

               if (rfyshr(j,k,l) .ge. vcr) then

                  sfrcdy = cmthk(l)
                  sfrcwy = 0.013
                  sfrcsy = cllth/10.

               endif

            else

               sfrcdy = 0.
               sfrcwy = 0.
               sfrcsy = 0.

            endif

c.....Calculate fracture characteristics for y (length) direction.

            if (rfymnt(j,k,l) .ge. crkmtr) then

               if (tencvy(l) .eq. 0. .and. rffspy(j,k,l).eq.0.) then

                  q = stlrad(l)**2*1.571/(stlspc(l)*otncvy(l))
                  if (stlrad(l) .lt. 1.e-15) q = ostlrd(l)**2*1.571/
     #               (stlspc(l)*otncvy(l))

               elseif (stlrad(l).lt.1.e-15 .and.
     #              rffspy(j,k,l).eq.0.) then

                  q = ostlrd(l)**2*1.571/(stlspc(l)*tencvy(l))

               elseif (tencvy(l).gt.0. .and.
     #              stlrad(l).ge.1.e-15) then

                  q = stlrad(l)**2*1.571/(stlspc(l)*tencvy(l))

               endif

               if (stlrad(l) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(l)*stlspc(l)/q)

               elseif (stlrad(l).lt.1.e-15 .and.
     #              rffspy(j,k,l).eq.0.) then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(l)*stlspc(l)/q)

               endif

               if (rffspy(j,k,l).eq.0. .or.
     #            rffspy(j,k,l).ge.2.*frspce)
     #            rffspy(j,k,l) = frspce

            endif

c.....Y-moments exceed cracking moment but not ultimate strength of roof.

            if (rfymnt(j,k,l).ge.crkmtr .and.
     #         rfymnt(j,k,l).lt.rfusty) then
```

```
                  efmnty = (crkmtr/rfymnt(j,k,l))**3.*cnmnti+
     #                     (1.-(crkmtr/rfymnt(j,k,l))**3.)*crmtiy
                  strsmy = rfymnt(j,k,l)*axneuy/efmnty
                  stltny = ratmod*rfymnt(j,k,l)*(rfdsty-axneuy)/efmnty
                  axsney = rfdsty/(stltny/stlmod+strsmy/conmod)*
     #                     (stltny/stlmod+csstrn)+tencvy(1)
                  betay = axsney/(axsney-tencvy(1))
                  rffdpy(j,k,l) = axsney
                  frcwdy = rffspy(j,k,l)*(stltny/stlmod*betay+csstrn)

               endif

c.....Y-moments exceed ultimate strength of roof.

               if (rfymnt(j,k,l).ge.rfusty.and.
     #             rffdpy(j,k,l).lt.cmthk(1)) then

                  rffdpy(j,k,l) = cmthk(1)
                  frcwdy = amin1((stlyld/stlmod+csstrn)*
     #                     rffspy(j,k,l),3.e-3*rffspy(j,k,l))

               endif

c.....Calculate cracking due to corrosion once it begins.

               if (icrflg(1).eq.1 .and. (j+k+l).eq.3)
     #            call ccrack(1,iyear)

c.....Calculate average crack characteristics for roof.

               if (cmthk(1) .eq. 0.) then

                  do 100 m=1,lyr

                     rfaper(m) = 0.
                     rffrac(m) = 0.

  100             continue

                  return

               else

                  fmax = .75*cmthk(1)
                  depth = amax1(rffdpx(j,k,l),sfrcdx,crfrcd(1))

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     krkx = krkx+1
                     if (rffspx(j,k,l) .gt. 0.)
     #                  tmp1 = clwid/10./rffspx(j,k,l)
                     if (crfrcs(1) .gt. 0.) tmp2 = cllth/10./crfrcs(1)
                     if (sfrcsx .gt. 0.) tmp3 = clwid/10./sfrcsx
                     tmp = tmp1+tmp2+tmp3
                     rfaper(j) = rfaper(j)+(frcwdx*tmp1+crfrcw(1)*tmp2+
     #                           sfrcwx*tmp3)/tmp
                     rffrac(j) = rffrac(j)+2.*cmthk(1)*cllth/10.*
     #                           (frcwdx*tmp1+sfrcwx*tmp3)

                  endif

                  depth = amax1(rffdpy(j,k,l),sfrcdy,crfrcd(1))

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     krky = krky+1
                     if (rffspy(j,k,l) .gt. 0.)
```

```
     #                  tmp1 = cllth/10./rffspy(j,k,1)
                     if (crfrcs(1) .gt. 0.) tmp2 = clwid/10./crfrcs(1)
                     if (sfrcsy .gt. 0.) tmp3 = cllth/10./sfrcsy
                     tmp = tmp1+tmp2+tmp3
                     rfaper(j) = rfaper(j)+(frcwdy*tmp1+crfrcw(1)*tmp2+
     #                           sfrcwy*tmp3)/tmp
                     rffrac(j) = rffrac(j)+2.*cmthk(1)*clwid/10.*
     #                            (frcwdy*tmp1+sfrcwy*tmp3)

                  endif

               endif

  200       continue

  300    continue

         if (rffrac(j) .gt. 0.) icrack(1) = 1
         rffrac(j) = rffrac(j)+crfrac(1)

  400 continue

      do 500 j=1,lyr

         if (rffrac(j) .gt. 0.) then

            rffrac(j) = rffrac(j)/(cmthk(1)*clwid*cllth)
            rfaper(j) = rfaper(j)/(krkx+krky)*2.54

         endif

  500 continue

      return
      end

      subroutine sar1

c-----------------------------------------------------------------------
c     Called by source1
c
c     Performs structural analysis of casks.
c
c     Calls: none
c-----------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,w1aper(3,2),w1frac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(3,11,11),rfymnt(3,11,11),flxmnt(3,11,11),
     #       flymnt(3,11,11),w1xmnt(3,11,11),w2xmnt(3,11,11),
     #       w1ymnt(3,11,11),w2ymnt(3,11,11)
      common/reactn/rfxrxn(3,11,11),rfyrxn(3,11,11)
      common/shear/rfxshr(3,11,11),rfyshr(3,11,11),flxshr(3,11,11),
     #       flyshr(3,11,11),w1xshr(3,11,11),w2xshr(3,11,11),
     #       w1yshr(3,11,11),w2yshr(3,11,11)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember
      common/w1forc/w1cmfy(3,11,11),w2cmfy(3,11,11)

      dimension cncfrx(3,11,11),cncfry(3,11,11)
      dimension rxnc(3),unfld(3)
      dimension w1mntx(11,11),w1mnty(11,11)
      dimension xmnnt(11,11),ymnnt(11,11),xshrt(20,20),yshrt(20,20)
      dimension xim(2,12),xiim(2,12),xip(2,12),xiip(2,12),yim(2,12),
```

```
     #          yiim(2,12),yip(2,12),yiip(2,12)

      real*4 num1,num2,num3,num4

      data pi/3.141592653589793/

c.....Calculate modulus of elasticity of concrete for use in structural
c.....analysis of floor.

      time = 365.
      comstr = amin1(time/(cfa+cfb*time)*com28d,constr)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)

c.....Begin roof structural analysis.

c.....Calculate x- and y-direction moments for roof of concrete casks.
c.....moment curves are discretized into eleven segments, over any one of
c.....which the moment is constant.

      jk = 12

      do 400 k=0,5

         jk = jk-2
         jl = 12

         do 300 l=0,5

            jl =  jl-2
            coef1 = 0.0
            coef2 = 0.0
            coef3 = 0.0
            coef4 = 0.0
            coef5 = 0.0
            coef6 = 0.0
            coef7 = 0.0

            do 100 m=1,15,2

               num1 = sin(pi*m*(l*clwid/10.)/clwid)
               num3 = cos(pi*m*(l*clwid/10.)/clwid)

               do 50 n=1,15,2

                  num2 = sin(pi*n*(k*cllth/10.)/cllth)
                  num4 = cos(pi*n*(k*cllth/10.)/cllth)
                  den1 = (m**2+n**2/(cllth/clwid)**2)**2
                  den2 = (n**2+m**2/(clwid/cllth)**2)**2
                  den3 = (m**2+n**2*(clwid/cllth)**2)**2
                  den4 = (m**2*cllth/clwid+n**2*(clwid/cllth))**2
                  den5 = (m**2*(cllth/clwid)**2+n**2)**2
                  den6 = (m**2*cllth/clwid+n**2*clwid/cllth)**2
                  coef1 = coef1+num1*num2/(n*den1/m) +
     #                    conpsn*(num1*num2/
     #                    ((cllth/clwid)**2*m*den1/n))
                  coef2 = coef2+num1*num2/(m*den2/n) +
     #                    conpsn*(num1*num2/
     #                    ((clwid/cllth)**2*n*den2/m))
                  coef3 = coef3+num2*num3*(m**2/(n*den3)+n/den4)
                  coef4 = coef4+num1*num4*(m/den4+n**2/(m*den5))
                  coef5 = coef5+num2*(m**2/(n*den3)+n*(2.-conpsn)/den4)
                  coef6 = coef6+num1*(n**2/(m*den5)+m*(2.-conpsn)/den4)
                  coef7 = coef7+1./den6

   50          continue

  100       continue

            do 200 j=1,lyr

c.....Calculate bending moments and shears for roof in x-direction.

               rfxmnt(j,k+1,l+1) = abs(16./pi**4.*coef1*
```

```
     #                             xload(j)*clwid**2)
               rfxshr(j,k+1,l+1) = abs(-16./pi**3.*coef3*xload(j)*clwid)
               rfxmnt(j,k+1,l+1+jl) = rfxmnt(j,k+1,l+1)
               rfxshr(j,k+1,l+1+jl) = abs(-rfxshr(j,k+1,l+1))
               rfxmnt(j,k+1+jk,l+1) = rfxmnt(j,k+1,l+1)
               rfxshr(j,k+1+jk,l+1) = abs(rfxshr(j,k+1,l+1))
               rfxmnt(j,k+1+jk,l+1+jl) = rfxmnt(j,k+1,l+1)
               rfxshr(j,k+1+jk,l+1+jl) = abs(-rfxshr(j,k+1,l+1))
               rxnc(j) = -32.*(1.-conpsn)/pi**4.*coef7*xload(j)*
     #                   clwid * cllth/(0.5*cmthk(2)+clwid/10.)

               if(k.eq.0 .or. l.eq.0) then

c.....Calculate roof reaction in x-direction.

                  rfxrxn(j,k+1,l+1) = 16./pi**3.*coef5*xload(j)*clwid
                  if(k.eq.0.and.l.eq.0) rfxrxn(j,k+1,l+1) =
     #               rfxrxn(j,k+1,l+1)+rxnc(j)/2.
                  rfxrxn(j,k+1,l+1+jl) = rfxrxn(j,k+1,l+1)
                  rfxrxn(j,k+1+jk,l+1) = rfxrxn(j,k+1,l+1)
                  rfxrxn(j,k+1+jk,l+1+jl) = rfxrxn(j,k+1,l+1)

               endif

c.....Calculate bending moments and shears for roof in y-direction.

               rfymnt(j,k+1,l+1) = abs(16./pi**4.*coef2*
     #                             xload(j)*cllth**2)
               rfyshr(j,k+1,l+1) = abs(-16./pi**3.*coef4*xload(j)*cllth)
               rfymnt(j,k+1,l+1+jl) = rfymnt(j,k+1,l+1)
               rfyshr(j,k+1,l+1+jl) = abs(rfyshr(j,k+1,l+1))
               rfymnt(j,k+1+jk,l+1) = rfymnt(j,k+1,l+1)
               rfyshr(j,k+1+jk,l+1) = abs(-rfyshr(j,k+1,l+1))
               rfymnt(j,k+1+jk,l+1+jl) = rfymnt(j,k+1,l+1)
               rfyshr(j,k+1+jk,l+1+jl) = abs(-rfyshr(j,k+1,l+1))

               if(k.eq.0 .or. (k.gt.0.and.l.eq.0)) then

c.....Calculate roof reaction in y-direction.

                  rfyrxn(j,k+1,l+1) = 16./pi**3.*coef6*xload(j)*cllth
                  if(k.eq.0.and.l.eq.0) rfyrxn(j,k+1,l+1) =
     #               rfyrxn(j,k+1,l+1)+rxnc(j)/2.
                  rfyrxn(j,k+1,l+1+jl) = rfyrxn(j,k+1,l+1)
                  rfyrxn(j,k+1+jk,l+1) = rfyrxn(j,k+1,l+1)
                  rfyrxn(j,k+1+jk,l+1+jl) = rfyrxn(j,k+1,l+1)

               endif

  200       continue

  300    continue

  400 continue

c.....Begin wall structural analysis.

c.....Calculate moments and shears due to uniform load for walls of
c.....concrete casks.  Calculations are discretized into eleven
c.....segments, over any one of which the moment or shear is assumed
c.....to be constant.

      flarg = 1.-sin(flangl*pi/180.)
      slarg = 1.-sin(slangl*pi/180.)

      do 500 i=1,lyr

c.....Calculate the uniform load on the wall.

         unfld(i) = sldns*3.61e-2*slarg*((cvrthk+0.5*cmthk(1))+(i-1)*
     #              (0.5*(cmthk(1)+cmthk(3))+clhght))

  500 continue
```

```
c.....Calculate the maximum hydrostatic pressure.

      hydrld = sldns*3.61e-2*clhght*slarg-wstdns*3.61e-2*clhght*flarg
      jk = 12

      do 1100 k=0,5

         jk = jk-2
         jl = 12

         do 1000 l=0,5

            jl =  jl-2
            coef1 = 0.0
            coef2 = 0.0
            coef3 = 0.0
            coef4 = 0.0

            do 800 m=1,15,2

               num1 = sin(pi*m*(l*clwid/10.)/clwid)
               num3 = cos(pi*m*(l*clwid/10.)/clwid)

               do 700 n=1,15,2

                  num2 = sin(pi*n*(k*clhght/10.)/clhght)
                  num4 = cos(pi*n*(k*clhght/10.)/clhght)
                  den1 = (m**2+n**2/(clhght/clwid)**2)**2
                  den2 = (n**2+m**2/(clwid/clhght)**2)**2
                  den3 = (m**2+n**2*(clwid/clhght)**2)**2
                  den4 = (m**2*clhght/clwid+n**2*(clwid/clhght))**2
                  den5 = (m**2*(clhght/clwid)**2+n**2)**2
                  coef1 = coef1+num1*num2/(n*den1/m) +
     #                    conpsn*(num1*num2/
     #                    ((clhght/clwid)**2*m*den1/n))
                  coef2 = coef2+num1*num2/(m*den2/n) +
     #                    conpsn*(num1*num2/
     #                    ((clwid/clhght)**2*n*den2/m))
                  coef3 = coef3+num2*num3*(m**2/(n*den3)+n/den4)
                  coef4 = coef4+num1*num4*(m/den4+n**2/(m*den5))

  700          continue

  800       continue

            do 900 m=1,lyr

c.....Calculate the bending moments and shear forces for wall 1.

               w1xmnt(m,k+1,l+1) = 16./pi**4.*coef1*unfld(m)*clwid**2
               w1xshr(m,k+1,l+1) = 16./pi**3.*coef3*unfld(m)*clwid
               w1xmnt(m,k+1,l+1+jl) = w1xmnt(m,k+1,l+1)
               w1xshr(m,k+1,l+1+jl) = w1xshr(m,k+1,l+1)
               w1xmnt(m,k+1+jk,l+1) = w1xmnt(m,k+1,l+1)
               w1xshr(m,k+1+jk,l+1) = -w1xshr(m,k+1,l+1)
               w1xmnt(m,k+1+jk,l+1+jl) = w1xmnt(m,k+1,l+1)
               w1xshr(m,k+1+jk,l+1+jl) = -w1xshr(m,k+1,l+1)
               w1ymnt(m,k+1,l+1) = 16./pi**4.*coef2*unfld(m)*clhght**2
               w1yshr(m,k+1,l+1) = 16./pi**3.*coef4*unfld(m)*clhght
               w1ymnt(m,k+1,l+1+jl) = w1ymnt(m,k+1,l+1)
               w1yshr(m,k+1,l+1+jl) = w1yshr(m,k+1,l+1)
               w1ymnt(m,k+1+jk,l+1) = w1ymnt(m,k+1,l+1)
               w1yshr(m,k+1+jk,l+1) = -w1yshr(m,k+1,l+1)
               w1ymnt(m,k+1+jk,l+1+jl) = w1ymnt(m,k+1,l+1)
               w1yshr(m,k+1+jk,l+1+jl) = -w1yshr(m,k+1,l+1)

  900       continue

 1000    continue

 1100 continue

c.....Perform calculations for second wall of cask.
```

```
      jk = 12

      do 1600 k=0,5

         jk = jk-2
         jl = 12

         do 1500 l=0,5

            jl =  jl-2
            coef1 = 0.0
            coef2 = 0.0
            coef3 = 0.0
            coef4 = 0.0

            do 1300 m=1,15,2

               num1 = sin(pi*m*(l*cllth/10.)/cllth)
               num3 = cos(pi*m*(l*cllth/10.)/cllth)

               do 1200 n=1,15,2

                  num2 = sin(pi*n*(k*clhght/10.)/clhght)
                  num4 = cos(pi*n*(k*clhght/10.)/clhght)
                  den1 = (m**2+n**2/(clhght/cllth)**2)**2
                  den2 = (n**2+m**2/(cllth/clhght)**2)**2
                  den3 = (m**2+n**2*(cllth/clhght)**2)**2
                  den4 = (m**2*clhght/cllth+n**2*(cllth/clhght))**2
                  den5 = (m**2*(clhght/cllth)**2+n**2)**2
                  coef1 = coef1+num1*num2/(n*den1/m) +
     #                    conpsn*(num1*num2/
     #                    ((clhght/cllth)**2*m*den1/n))
                  coef2 = coef2+num1*num2/(m*den2/n) +
     #                    conpsn*(num1*num2/
     #                    ((cllth/clhght)**2*n*den2/m))
                  coef3 = coef3+num2*num3*(m**2/(n*den3)+n/den4)
                  coef4 = coef4+num1*num4*(m/den4+n**2/(m*den5))

 1200          continue

 1300       continue

            do 1400 m=1,lyr

c.....Calculate the bending moments and shear forces for wall 2.

               w2xmnt(m,k+1,l+1) = 16./pi**4.*coef1*unfld(m)*cllth**2
               w2xshr(m,k+1,l+1) = 16./pi**3.*coef3*unfld(m)*cllth
               w2xmnt(m,k+1,l+1+jl) = w2xmnt(m,k+1,l+1)
               w2xshr(m,k+1,l+1+jl) = w2xshr(m,k+1,l+1)
               w2xmnt(m,k+1+jk,l+1) = w2xmnt(m,k+1,l+1)
               w2xshr(m,k+1+jk,l+1) = -w2xshr(m,k+1,l+1)
               w2xmnt(m,k+1+jk,l+1+jl) = w2xmnt(m,k+1,l+1)
               w2xshr(m,k+1+jk,l+1+jl) = -w2xshr(m,k+1,l+1)
               w2ymnt(m,k+1,l+1) = 16./pi**4.*coef2*unfld(m)*clhght**2
               w2yshr(m,k+1,l+1) = 16./pi**3.*coef4*unfld(m)*clhght
               w2ymnt(m,k+1,l+1+jl) = w2ymnt(m,k+1,l+1)
               w2yshr(m,k+1,l+1+jl) = w2yshr(m,k+1,l+1)
               w2ymnt(m,k+1+jk,l+1) = w2ymnt(m,k+1,l+1)
               w2yshr(m,k+1+jk,l+1) = -w2yshr(m,k+1,l+1)
               w2ymnt(m,k+1+jk,l+1+jl) = w2ymnt(m,k+1,l+1)
               w2yshr(m,k+1+jk,l+1+jl) = -w2yshr(m,k+1,l+1)

 1400       continue

 1500    continue

 1600 continue

c.....Calculate moments and shears due to hydrostatic load for walls of
c.....concrete casks.  Calculations are discretized into eleven
c.....segments, over any one of which the moment or shear is assumed
c.....to be constant.
```

```
      do 2000 k=0,10

         jl = 12

         do 1900 l=0,5

            jl =  jl-2
            coef1 = 0.0
            coef2 = 0.0
            coef3 = 0.0
            coef4 = 0.0

            do 1700 m=1,12

c.....Calculate quantities needed to determine the bending moments
c.....due to hydrostatic pressures.

               tmpam = m*pi*clwid/(2.*clhght)
               am = -(2+tmpam*tanh(tmpam))*(-1.)**(m+1)/(pi**5.*m**5.*
     #             cosh(tmpam))
               bm = (-1.)**(m+1)/(pi**5.*m**5.*cosh(tmpam))
               num1 = cosh(pi*m*(clwid/2.-l*clwid/10.)/clhght)
               num2 = sinh(pi*m*(clwid/2.-l*clwid/10.)/clhght)
               num3 = sin(pi*m*(k*clhght/10.)/clhght)
               aacf1 = 2.*(-1.)**(m+1)/(pi**5.*m**5.)*conpsn
               bbcf1 = ((conpsn-1.)*am-2.*bm)*num1
               cccf1 = (conpsn-1.)*bm*
     #                 (m*pi/clhght*(clwid/2.-l*clwid/10.)*num2)
               coef1 = coef1+(m*pi)**2*(aacf1 + bbcf1 + cccf1)*num3
           aacf3 = 2.*(-1.)**(m+1)/(pi**5.*m**5.)
               bbcf3 = ((1.-conpsn)*am-2.*conpsn*bm)*num1
               cccf3 = (1.-conpsn)*bm*
     #                 (m*pi/clhght*(clwid/2.-l*clwid/10.)*num2)
               coef3 = coef3+(m*pi)**2*(aacf3 + bbcf3 + cccf3)*num3
               coef2 = coef2+(m*pi)**3.*bm*num2*num3
               coef4 = coef4+(m*pi)**3.*((-1.)**(m+1)/(pi**5.*m**5.)-
     #                 bm*num1)*cos(pi*m*(k*clhght/10.)/clhght)

 1700       continue

c.....Combine moments and shears for uniform and hydrostatic loads on
c.....walls while calculating moments and shears for hydrostatic load.

            tmp1 = hydrld*clhght**2*coef1
            tmp2 = -2.*hydrld*clhght*coef2
            tmp3 = hydrld*clhght**2*coef3
            tmp4 = 2.*hydrld*clhght*coef4

            do 1800 m=1,lyr

               wlxmnt(m,k+1,l+1) = abs(wlxmnt(m,k+1,l+1)+tmp1)
               wlxmnt(m,k+1,l+1+jl) = abs(wlxmnt(m,k+1,l+1+jl)+tmp1)
               wlxshr(m,k+1,l+1) = abs(wlxshr(m,k+1,l+1)+tmp2)
               wlxshr(m,k+1,l+1+jl) = abs(wlxshr(m,k+1,l+1+jl)-tmp2)
               wlymnt(m,k+1,l+1) = abs(wlymnt(m,k+1,l+1)+tmp3)
               wlymnt(m,k+1,l+1+jl) = abs(wlymnt(m,k+1,l+1+jl)+tmp3)
               wlyshr(m,k+1,l+1) = abs(wlyshr(m,k+1,l+1)+tmp4)
               wlyshr(m,k+1,l+1+jl) = abs(wlyshr(m,k+1,l+1+jl)+tmp4)

 1800       continue

 1900    continue

 2000 continue

c.....Perform calculations for hydrostatic pressure for second wall.

      do 2400 k=0,10

         jl = 12

         do 2300 l=0,5
```

```
            j1 =  j1-2
            coef1 = 0.0
            coef2 = 0.0
            coef3 = 0.0
            coef4 = 0.0

            do 2100 m=1,12

c.....Calculate quantities needed to determine the bending moments
c.....due to hydrostatic pressures.

               tmpam = m*pi*cllth/(2.*clhght)
               am = -(2+tmpam*tanh(tmpam))*(-1.)**(m+1)/(pi**5.*m**5.*
     #              cosh(tmpam))
               bm = (-1.)**(m+1)/(pi**5.*m**5.*cosh(tmpam))
               num1 = cosh(pi*m*(cllth/2.-l*cllth/10.)/clhght)
               num2 = sinh(pi*m*(cllth/2.-l*cllth/10.)/clhght)
               num3 = sin(pi*m*(k*clhght/10.)/clhght)

               aacf1 = 2.*(-1.)**(m+1)/(pi**5.*m**5.)*conpsn
               bbcf1 = ((conpsn-1.)*am-2.*bm)*num1
               cccf1 = (conpsn-1.)*bm*
     #                 (m*pi/clhght*(clwid/2.-l*clwid/10.)*num2)
               coef1 = coef1+(m*pi)**2*(aacf1 + bbcf1 + cccf1)*num3
           aacf3 = 2.*(-1.)**(m+1)/(pi**5.*m**5.)
               bbcf3 = ((1.-conpsn)*am-2.*conpsn*bm)*num1
               cccf3 = (1.-conpsn)*bm*
     #                 (m*pi/clhght*(clwid/2.-l*clwid/10.)*num2)
               coef3 = coef3+(m*pi)**2*(aacf3 + bbcf3 + cccf3)*num3
               coef2 = coef2+(m*pi)**3.*bm*num2*num3
               coef4 = coef4+(m*pi)**3.*((-1.)**(m+1)/(pi**5.*m**5.)-
     #                bm*num1)*cos(pi*m*(k*clhght/10.)/clhght)

 2100       continue

c.....Combine moments and shears for uniform and hydrostatic loads on
c.....walls while calculating moments and shears for hydrostatic load.

            tmp1 = hydrld*clhght**2*coef1
            tmp2 = -2.*hydrld*clhght*coef2
            tmp3 = hydrld*clhght**2*coef3
            tmp4 = 2.*hydrld*clhght*coef4

            do 2200 m=1,lyr

               w2xmnt(m,k+1,l+1) = abs(w2xmnt(m,k+1,l+1)+tmp1)
               w2xmnt(m,k+1,l+1+j1) = abs(w2xmnt(m,k+1,l+1+j1)+tmp1)
               w2xshr(m,k+1,l+1) = abs(w2xshr(m,k+1,l+1)+tmp2)
               w2xshr(m,k+1,l+1+j1) = abs(w2xshr(m,k+1,l+1+j1)-tmp2)
               w2ymnt(m,k+1,l+1) = abs(w2ymnt(m,k+1,l+1)+tmp3)
               w2ymnt(m,k+1,l+1+j1) = abs(w2ymnt(m,k+1,l+1+j1)+tmp3)
               w2yshr(m,k+1,l+1) = abs(w2yshr(m,k+1,l+1)+tmp4)
               w2yshr(m,k+1,l+1+j1) = abs(w2yshr(m,k+1,l+1+j1)+tmp4)

 2200       continue

 2300    continue

 2400 continue

c.....Calculate compressive forces on walls.

      do 2700 l=1,lyr

         do 2600 m=0,10

            do 2500 n=1,11

c.....Calculate compressive force on wall 1.

               w1cmfy(l,m+1,n) = amax1(0.,rfyrxn(l,1,n)+cmthk(2)*m*
     #                          clhght/10.*ccdns*3.61e-2)
```

```
 2500       continue

 2600    continue

 2700 continue

      do 3000 l=1,lyr

         do 2900 m=0,10

            do 2800 n=1,11

c.....Calculate compressive force on wall 2.

               w2cmfy(l,m+1,n) = amaxl(0.,rfxrxn(l,n,l)+cmthk(2)*m*
     #                           clhght/10.*ccdns*3.61e-2)

 2800       continue

 2900    continue

 3000 continue

c.....Begin floor structural analysis.

c.....Calculate concentrated forces and x- and y-direction moments for
c.....floor of concrete casks. moment curves are discretized into eleven
c.....segments, over any one of which the moment is constant.

c.....Calculate moment of inertia of concrete section.

      cnmntf = cmthk(3)**3./12.

      do 3300 l=1,lyr

         do 3200 m=1,11

            do 3100 n=1,11,10

c.....Calculate concentrated load on the floor in the x-direction.

               cncfrx(l,m,n) = amaxl(0.,rfxrxn(l,m,n)+cmthk(2)*clhght*
     #                         (ccdns-wstdns)*3.61e-2)

 3100       continue

 3200    continue

 3300 continue

      do 3600 l=1,lyr

         do 3500 m=1,11,10

            do 3400 n=1,11

c.....Calculate concentrated load on the floor in the y-direction.

               cncfry(l,m,n) = amaxl(0.,rfyrxn(l,m,n)+cmthk(2)*clhght*
     #                         (ccdns-wstdns)*3.61e-2)

 3400       continue

 3500    continue

 3600 continue

      xlamda = (submod/(4.*conmod*cnmntf)*(1.-conpsn**2))**.25
      axx = ovrhng+0.5*cmthk(2)
      x1 = xlamda*(clwid*noclx+2.*axx)
      y1 = xlamda*(cllth*nocly+2.*axx)
      xchi = sinh(x1)**2-sin(x1)**2
      ychi = sinh(y1)**2-sin(y1)**2
```

```
      do 3800 m=1,noclx+1

         ax = axx+(m-1)*clwid
         cx = axx+(noclx-m+1)*clwid
         x3 = xlamda*ax
         x4 = xlamda*cx

         do 3700 k=1,10*m-9

c.....Calculate geometry of the floor for the bending moments
c.....and shears in the x-direction.

            x2 = xlamda*(axx+(k-1)*clwid/10.)
            txip1 = 2.*sinh(x2)*sin(x2)*(sinh(x1)*cos(x3)*cosh(x4)-
     #              sin(x1)*cosh(x3)*cos(x4))
            txip2 =    (sinh(x2)*cos(x2)-cosh(x2)*sin(x2))
            txip3 = (sinh(x1)*(sin(x3)*cosh(x4)-cos(x3)*sinh(x4))+
     #              sin(x1)*(sinh(x3)*cos(x4)-cosh(x3)*sin(x4)))
            xip(m,k) = txip1 -txip2 * txip3
            txiip1 = (cosh(x2)*sin(x2)+sinh(x2)*cos(x2))*(sinh(x1)*
     #               cos(x3)*cosh(x4)-sin(x1)*cosh(x3)*cos(x4))
            txiip2 = (sinh(x1)*(sin(x3)*cosh(x4)-cos(x3)*sinh(x4))
     #               +sin(x1)*(sinh(x3)*cos(x4)-cosh(x3)*sin(x4)))
            xiip(m,k) = txiip1 + sinh(x2) * sin(x2) * txiip2
            txim1 = (sinh(x1)*(cos(x1)*sinh(x4)*cos(x4)+sin(x3)*
     #              cosh(x4))+sin(x1)*(cosh(x1)*cosh(x4)*sin(x4)
     #              +sinh(x3)*cos(x4)))
            txim2 = (sinh(x1)*cos(x3)*cosh(x4)+sin(x1)*
     #              cosh(x3)*cos(x4))
            xim(m,k)  = sinh(x2)*sin(x2)*txim1+(sinh(x2)*cos(x2)-
     #                 cosh(x2)*sin(x2))*txim2
            txiim1 = (cosh(x2)*sin(x2)+sinh(x2)*cos(x2))
            txiim2 = (cos(x1)*sinh(x4)*cos(x4)+sin(x3)*cosh(x4))
            txiim3 = (cosh(x1)*cosh(x4)*sin(x4)+sinh(x3)*cos(x4))
            txiim4 = (sinh(x1)*cos(x3)*cosh(x4)+sin(x1)*
     #               cosh(x3)*cos(x4))
            xiim(m,k)  = txiim1*(sinh(x1)*txiim2+sin(x1)*
     #                  txiim3)-2.*sinh(x2)*sin(x2)*txiim4

 3700    continue

 3800 continue

      do 4000 j=1,nocly+1

         ay = axx+(j-1)*cllth
         cy = axx+(nocly-j+1)*cllth
         y3 = xlamda*ay
         y4 = xlamda*cy

         do 3900 k=1,10*j-9

c.....Calculate geometry of the floor for the bending moments
c.....and shears in the y-direction.

            y2 = xlamda*(axx+(k-1)*cllth/10.)
            tyip1 = (sinh(y1)*cos(y3)*cosh(y4)-sin(y1)*
     #              cosh(y3)*cos(y4))
            tyip2 = (sinh(y1)*(sin(y3)*cosh(y4)-cos(y3)*sinh(y4))+
     #               sin(y1)*(sinh(y3)*cos(y4)-cosh(y3)*sin(y4)))
            yip(j,k)  = 2*sinh(y2)*sin(y2)*tyip1-(sinh(y2)*cos(y2)-
     #                 cosh(y2)*sin(y2))*tyip2
            tyiip1 = (sinh(y1)*cos(y3)*cosh(y4)-sin(y1)*
     #               cosh(y3)*cos(y4))
            tyiip2 = (sinh(y1)*(sin(y3)*cosh(y4)-cos(y3)*sinh(y4))
     #               +sin(y1)*(sinh(y3)*cos(y4)-cosh(y3)*sin(y4)))
            yiip(j,k)  = (cosh(y2)*sin(y2)+sinh(y2)*cos(y2))*tyiip1+
     #                  sinh(y2)*sin(y2)*tyiip2
            tyim1 = (cos(y1)*sinh(y4)*cos(y4)+sin(y3)*cosh(y4))
            tyim2 = (cosh(y1)*cosh(y4)*sin(y4)+sinh(y3)*cos(y4))
            tyim3 = (sinh(y2)*cos(y2)-cosh(y2)*sin(y2))
            tyim4 = (sinh(y1)*cos(y3)*cosh(y4)+sin(y1)*cosh(y3)*
     #              cos(y4))
```

```
            yim(j,k) = sinh(y2)*sin(y2)*(sinh(y1)*tyim1+sin(y1)*tyim2)
     #                  +tyim3*tyim4
            tyiim1 = (sinh(y1)*(cos(y1)*sinh(y4)*cos(y4)+sin(y3)*
     #                cosh(y4))+sin(y1)*(cosh(y1)*cosh(y4)*sin(y4)+
     #                sinh(y3)*cos(y4)))
            tyiim2 = (sinh(y1)*cos(y3)*cosh(y4)+sin(y1)*
     #                cosh(y3)*cos(y4))
            yiim(j,k) = (cosh(y2)*sin(y2)+sinh(y2)*cos(y2))*tyiim1-2.*
     #                   sinh(y2)*sin(y2)*tyiim2

 3900    continue

 4000 continue

      do 4200 k=1,noclx+1

         do 4100 l=1,10*k-9

            wlmntx(k,l) = 0.
            wlmnty(k,l) = 0.
            xmnnt(k,l) = xip(k,l)/(2.*xlamda*xchi)+
     #                   wlmntx(k,l)/xchi*xim(k,l)
            xshrt(k,l) = 1./xchi*xiip(k,l)+wlmntx(k,l)*
     #                   xlamda/xchi*xiim(k,l)
            ymnnt(k,l) = yip(k,l)/(2.*xlamda*ychi)+
     #                   wlmnty(k,l)/ychi*yim(k,l)
            yshrt(k,l) = 1./ychi*yiip(k,l)+wlmnty(k,l)*
     #                   xlamda/ychi*yiim(k,l)
 4100    continue

 4200 continue

c.....Calculate bending moments and shear forces in the x and y-direction.

      xmnnt(1,1) = xmnnt(1,1)+xmnnt(2,1)
      xshrt(1,1) = xshrt(2,1)-xshrt(2,11)
      ymnnt(1,1) = ymnnt(1,1)+ymnnt(2,1)
      yshrt(1,1) = yshrt(2,1)-yshrt(2,11)
      jk = 10

      do 4300 k=2,6

         jk = jk-2
         xmnnt(1,k) = xmnnt(2,k)+xmnnt(2,k+jk)
         xshrt(1,k) = xshrt(2,k)-xshrt(2,k+jk)
         ymnnt(1,k) = ymnnt(2,k)+ymnnt(2,k+jk)
         yshrt(1,k) = yshrt(2,k)-yshrt(2,k+jk)

 4300 continue

      jk = 0

      do 4400 k=7,11

         jk = jk+2
         xmnnt(1,k) = xmnnt(1,k-jk)
         xshrt(1,k) = xshrt(2,k)-xshrt(2,k-jk)
         ymnnt(1,k) = ymnnt(1,k-jk)
         yshrt(1,k) = yshrt(2,k)-yshrt(2,k-jk)

 4400 continue

      do 4700 m=1,lyr

         do 4600 l=1,11

            do 4500 k=1,11

               flxmnt(m,l,k) = abs(xmnnt(1,k)*cncfrx(m,l,1))
               flxshr(m,l,k) = abs(xshrt(1,k)*(-1.)*cncfrx(m,l,1))
               flymnt(m,k,l) = abs(ymnnt(1,k)*cncfry(m,l,1))
               flyshr(m,k,l) = abs(yshrt(1,k)*(-1.)*cncfry(m,l,1))
```

```
 4500        continue

 4600     continue

 4700 continue

      end

      subroutine sulfate(iyear)

c-----------------------------------------------------------------------------
c     Called by concrete
c
c     Calculates loss of concrete thickness due to sulfate attack.
c
c     Calls: none
c
c-----------------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
     #       slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #       w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/padc/pstlrad,pstlmod,pstlyld,pconstr,
     #            pbotcov,pwtcmnt,pstlspc,padcrk,piff,intctrl
      common/tumulus/lyr,numwid,numlth,numcsk,nmember

c.....Rate of degradation calculated as per Atkinson and Hearne.

      if (iyear .eq. 1) then

c.....Begin outside disposal facility calculations.

         tmp=1.24                                   !Atkinson & Hearne
         so4o = so4o*1000.

  100    continue

c.....Estimate ettringite concentration.

         ce = wtcmnt*tmp

c.....Calculate reaction zone thickness at which spalling occurs.

         xspl = 2.*1.*10.*(1-conpsn)/(yngmod*(1.8e-6*ce)**2)

c.....Calculate time when spalling occurs.

         tspl = xspl**2*ce/(2.*dfso4*so4o)
         t = 10.**(tmp/.32-alog10(so4o)+alog10(3577.)+alog10(12.2))
         tmp = tmp*.99

         if(tspl.lt.t) go to 100

c.....Concrete loss from outside of disposal facility.

         slfo = xspl*39.37/tspl*3.15e7

c.....Begin inside disposal facility calculations.

         tmp = 1.24                                 !Atkinson & Hearne
         so4i = so4i*1000.
```

```
 200    continue

c.....Estimate ettringite concentration.

        ce = wtcmnt*tmp

c.....Calculate reaction zone thickness at which spalling occurs.

        xspl = 2.*1.*10.*(1-conpsn)/(yngmod*(1.8e-6*ce)**2)

c.....Calculate time when spalling occurs.

        tspl = xspl**2*ce/(2.*dfso4*so4i)
        t = 10.**(tmp/.32-alog10(so4i)+alog10(3577.)+alog10(12.2))
        tmp = tmp*.99

        if(tspl.lt.t) go to 200

c.....Concrete loss from inside of disposal facility.

        slfi = xspl*39.37/tspl*3.15e7

c.....Begin pad calculations.

        if (nmember .eq. 4) then

          tmp = 1.24                                  !atkinson & hearne

 300      continue

c.....Estimate ettringite concentration.

          ce = pwtcmnt*tmp

c.....Calculate reaction zone thickness at which spalling occurs.

          xspl = 2.*1.*10.*(1-conpsn)/(yngmod*(1.8e-6*ce)**2)

c.....Calculate time when spalling occurs.

          tspl = xspl**2*ce/(2.*dfso4*so4o)
          t = 10.**(tmp/.32-alog10(so4o)+alog10(3577.)+alog10(12.2))
          tmp = tmp*.99

          if(tspl.lt.t) go to 300

c.....Concrete loss from pad.

          pslfo = xspl*39.37/tspl*3.15e7

        endif

      endif

c.....Update total member thicknesses.

      do 400 i=1,nmember

        if(i .ne. 4)then

          cmthk(i) = amax1(0.,cmthk(i)-(slfi+slfo))

        else

          cmthk(i) = amax1(0.,cmthk(i)-pslfo)

        endif

 400  continue

c.....Update cover thickness on compression and tension faces of concrete.

      do 500 i=1,3
```

```
         comcvx(i) = amax1(0.,comcvx(i)-slfo)
         comcvy(i) = amax1(0.,comcvy(i)-slfo)
         tencvx(i) = amax1(0.,tencvx(i)-slfi)
         tencvy(i) = amax1(0.,tencvy(i)-slfi)

  500 continue

      return
      end

      function sxierfc (x)

c-----------------------------------------------------------------------
c     Called by: flothru
c
c     Function used in diffusion leaching calculations (2 december 1991).
c
c     Calls: none
c-----------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      common/numb/mmax

      data rsrpi / 5.641895835477563d-1/

      xsq = x**2
      u = rsrpi*xsq
      sum = rsrpi - x + u
      d = 6.d0
      e = 9.d0
      thm2m = -1.d0

      do 100 m=2,30

         mmax = m
         u = u*xsq*thm2m/d
         sum = sum + u
         if (abs(u/sum) .lt. 5.d-9) go to 110
         d = d + e
         e = e + 4.d0
         thm2m = thm2m - 2.d0

  100 continue

      write(*,*) 'did not converge in sxierfc'

      return

  110 continue

      sxierfc = sum

      return
      end

      subroutine walls(iyear)

c-----------------------------------------------------------------------
c     Called by: sourcel
c
c     Performs cracking analysis for cask walls.
c
c     Calls: ccrack
c-----------------------------------------------------------------------

      common/cask/clhght,cllth,clwid,cmthk(4),comcvx(3),comcvy(3),
     #       cvrdns,cvrthk,flangl,noclx,nocly,ommthk(4),ostlrd(3),
     #       otncvx(3),otncvy(3),ovrhng,slangl,sldns,stlrad(3),
     #       stlspc(3),submod,tencvx(3),tencvy(3),wstdns,wsthk,wstht
      common/clcult/annprc,attk(4),crfrac(3),crfrcd(3),crfrcs(3),
     #       crfrcw(3),crpcof,csstrn,flaper(3),flfrac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ispl(3),ph(4),rfaper(3),rffrac(3),
```

```
     #      slfi,slfo,stlcor(3),ttlwat,wlaper(3,2),wlfrac(3,2),
     #      w2aper(3,2),w2frac(3,2),xload(3),xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #      co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #      yngmod
      common/moment/rfxmnt(3,11,11),rfymnt(3,11,11),flxmnt(3,11,11),
     #      flymnt(3,11,11),wlxmnt(3,11,11),w2xmnt(3,11,11),
     #      wlymnt(3,11,11),w2ymnt(3,11,11)
      common/shear/rfxshr(3,11,11),rfyshr(3,11,11),flxshr(3,11,11),
     #      flyshr(3,11,11),wlxshr(3,11,11),w2xshr(3,11,11),
     #      wlyshr(3,11,11),w2yshr(3,11,11)
      common/tumulus/lyr,numwid,numlth,numcsk,nmember
      common/wlforc/wlcmfy(3,11,11),w2cmfy(3,11,11)
      common/wlfrac/wlfdpx(3,11,11),wlfdpy(3,11,11),wlfspx(3,11,11),
     #      wlfspy(3,11,11),w2fdpx(3,11,11),w2fdpy(3,11,11),
     #      w2fspx(3,11,11),w2fspy(3,11,11)

      data pi,stredx/3.141592653589793,0.9/

c.....Calculate time-dependent parameters used in cracking analysis.
c.....Horizontal steel is x-direction steel; vertical steel is y-direction
c.....steel.

      time = iyear*365.
      comstr = aminl(time/(cfa+cfb*time)*com28d*attk(2),constr*attk(2))
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      wldstx = cmthk(2)-tencvx(2)
      wldsty = cmthk(2)-tencvy(2)
      starcm = 0.
      startn = stlrad(2)**2*pi/stlspc(2)
      cnmnti = cmthk(2)**3./12.
      crkmtw = cnmnti/(0.5*cmthk(2))*rupmod

c.....Calculate ultimate strength for horizontal (x) direction of walls.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c = .003*stlmod*starcm*comcvx(2)
      axsneu = (-b+sqrt(b**2-4.*a*c))/(2.*a)

      if(axsneu .le. comcvx(2)) then

        cmblk = startn*stlyld/(0.85*comstr)
        wlustx = amaxl(crkmtw,stredx*stlyld*startn*(wldstx-cmblk/2.))

      else

        csstrs = (axsneu-comcvx(2))/axsneu*.003*stlmod
        as2 = starcm*csstrs/stlyld
        asl = startn-as2
        cmblk = asl*stlyld/(0.85*comstr)
        wlustx = amaxl(crkmtw,stredx*(asl*stlyld*(wldstx-cmblk/2.)+
     #          starcm*csstrs*(wldstx-comcvx(2))))

      endif

c.....Calculate cracking moment of inertia for walls for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      ccx = comcvx(2)*starcm*(ratmod-1.)-wldstx*ratmod*startn
      ccy = comcvy(2)*starcm*(ratmod-1.)-wldsty*ratmod*startn
      rttlx = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rttly = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rttlx
      axneuy = rttly
      crmtix = 0.333*axneux**3.+starcm*(ratmod-1.)*(axneux-comcvx(2))
     #         **2+ratmod*startn*(wldstx-axneux)**2
      crmtiy = 0.333*axneuy**3.+starcm*(ratmod-1.)*(axneuy-comcvy(2))
```

```
     #         **2+ratmod*startn*(wldsty-axneuy)**2

c.....Calculate cracking due to shear for first wall for all layers of casks.

      xk = (1.6+2.4*(clwid/cllth-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*wldstx
      cmblk = 87000./(87000.+stlyid)*0.85*wldsty
      pb = 0.7*(0.85*comstr*cmblk-startn*stlyld)
      wlustc = 0.55*0.7*cmthk(2)*comstr*(1.-(clhght/(32.*cmthk(2)))**2)

      do 500 j=1,lyr

         krkx = 0
         krky = 0
         wlfrac(j,1) = 0.
         wlfrac(j,2) = 0.
         wlaper(j,1) = 0.
         wlaper(j,2) = 0.

         do 400 k=1,11

            do 300 l=1,11

               frcwdx = 0.
               frcwdy = 0.

               if (wlxshr(j,k,l) .ge. shrstx) then

                  if (wlxmnt(j,k,l) .gt. 0.) then

                     tmp = amin1(wlxshr(j,k,l)/wlxmnt(j,k,l)*wldstx,1.)
                     vcr = amin1((1.9*sqrt(comstr)+2500.*
     #                     startn/wldstx*tmp)*
     #                     wldstx,3.5*sqrt(comstr)*wldstx)

                  else

                     vcr = 3.5*sqrt(comstr)*wldstx

                  endif

                  if(wlxshr(j,k,l) .ge. vcr) then

                     sfrcdx = cmthk(2)
                     sfrcwx = 0.013
                     sfrcsx = clwid/10.

                  endif

               else

                  sfrcdx = 0.
                  sfrcwx = 0.
                  sfrcsx = 0.

               endif

c.....Calculate fracture characteristics for horizontal (x) direction due to
c.....bending.

               if (wlxmnt(j,k,l) .ge. crkmtw) then

                  if (tencvx(2).eq.0. .and. wlfspx(j,k,l).eq.0.) then

                     q = stlrad(2)**2*1.571/(stlspc(2)*otncvx(2))
                     if (stlrad(2) .lt. 1.e-15) q = ostlrd(2)**2*1.571/
     #                  (stlspc(2)*otncvx(2))

                  elseif (stlrad(2).lt.1.e-15 .and.
     #                   wlfspx(j,k,l).eq.0.) then

                     q = ostlrd(2)**2*1.571/(stlspc(2)*tencvx(2))
```

```
                  elseif (tencvx(2).gt.0. .and.
     #                   stlrad(2).ge.1.e-15) then

                     q = stlrad(2)**2*1.571/(stlspc(2)*tencvx(2))

                  endif

                  if (stlrad(2) .ge. 1.e-15) then

                     frspce = 0.5*xk*sqrt(2.*stlrad(2)*stlspc(2)/q)

                  elseif (stlrad(2).lt.1.e-15 .and.
     #                   wlfspx(j,k,l).eq.0.) then

                     frspce = 0.5*xk*sqrt(2.*ostlrd(2)*stlspc(2)/q)

                  endif

                  if (wlfspx(j,k,l).eq.0. .or.
     #               wlfspx(j,k,l).ge.2.*frspce)
     #                  wlfspx(j,k,l) = frspce

               endif

c.....X-moments exceed cracking moment but not ultimate strength of wall.

               if (wlxmnt(j,k,l).ge.crkmtw .and.
     #            wlxmnt(j,k,l).lt.wlustx) then

                  efmntx = (crkmtw/wlxmnt(j,k,l))**3.*cnmnti+
     #                     (1.-(crkmtw/wlxmnt(j,k,l))**3.)*crmtix
                  strsmx = wlxmnt(j,k,l)*axneux/efmntx
                  stltnx = ratmod*wlxmnt(j,k,l)*(wldstx-axneux)/efmntx
                  axsnex = wldstx/(stltnx/stlmod+strsmx/conmod)*
     #                     (stltnx/stlmod+csstrn)+tencvx(2)
                  betax = axsnex/(axsnex-tencvx(2))
                  wlfdpx(j,k,l) = axsnex
                  frcwdx = wlfspx(j,k,l)*(stltnx/stlmod*betax+csstrn)

               endif

c.....X-moments exceed ultimate strength of wall.

               if (wlxmnt(j,k,l).ge.wlustx.and.
     #            wlfdpx(j,k,l).lt.cmthk(2)) then

                  wlfdpx(j,k,l) = cmthk(2)
                  frcwdx = amin1((stlyld/stlmod+csstrn)*
     #                     wlfspx(j,k,l),3.e-3*wlfspx(j,k,l))

               endif

c.....Perform cracking calculations for first wall in vertical (y) direction.
c.....Calculate ultimate strength for vertical (y) direction.

               crkmnt = cnmnti/(0.5*cmthk(2))*(rupmod+wlcmfy(j,k,l)/
     #                  cmthk(2))
               strred = 0.9-0.2*wlcmfy(j,k,l)*1.7/amin1(0.1*comstr*
     #                  cmthk(2),pb)
               ecmblk = (startn*stlyld+wlcmfy(j,k,l)/strred)/
     #                  (0.85*comstr)
               wlusty = amaxl(crkmnt,strred*.085*comstr*ecmblk*
     #                  (wldsty-ecmblk/2.))

c.....Shear cracking calculations for vertical (y) direction. Calculation
c.....differs from those for horizontal (x) direction as there is also
c.....compressive force.

               mm = wlymnt(j,k,l)-wlcmfy(j,k,l)*(4.*cmthk(2)-wldsty)/8.
               tmp = 3.5*sqrt(comstr)*wldsty*(1.+wlcmfy(j,k,l)/(500.*
     #               cmthk(2)))**.5

               if(mm .gt. 0.) then
```

```
               vcr = amin1((1.9*sqrt(comstr)+2500.*startn/wldsty*
     #               wlyshr(j,k,l)*wldsty/mm)*wldsty,tmp)

            else

               vcr = tmp

            endif

            if(abs(wlyshr(j,k,l)) .ge. vcr) then

               sfrcdy = cmthk(2)
               sfrcwy = 0.013
               sfrcsy = clhght/10.

            else

               sfrcdy = 0.
               sfrcwy = 0.
               sfrcsy = 0.

            endif

c.....Calculate cracking of interior walls due to compression. No cracking
c.....due to bending is assumed for these walls. When cracking occurs there
c.....is a single (horizontal) crack per section.

            if(wlcmfy(j,k,l) .ge. wlustc) then

               cfrcdp = cmthk(2)
               cfrcwd = .003*clhght/10.
               cfrcsp = clhght/10.

            else

               cfrcdp = 0.
               cfrcwd = 0.
               cfrcsp = 0.

            endif

c.....Calculate fracture characteristics.

            if(wlymnt(j,k,l) .ge. crkmnt) then

               if (tencvy(2).eq.0. .and. wlfspy(j,k,l).eq.0.) then

                  q = stlrad(2)**2*1.571/(stlspc(2)*otncvy(2))
                  if (stlrad(2) .lt. 1.e-15) q = ostlrd(2)**2*1.571/
     #               (stlspc(2)*otncvy(2))

               elseif (stlrad(2).lt.1.e-15 .and.
     #                wlfspy(j,k,l).eq.0.) then

                  q = ostlrd(2)**2*1.571/(stlspc(2)*tencvy(2))

               elseif (tencvy(2).gt.0. .and.
     #                stlrad(2).ge.1.e-15) then

                  q = stlrad(2)**2*1.571/(stlspc(2)*tencvy(2))

               endif

               if (stlrad(2) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(2)*stlspc(2)/q)

               elseif (stlrad(2).lt.1.e-15 .and.
     #                wlfspy(j,k,l).eq.0.) then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(2)*stlspc(2)/q)

               endif
```

```
                if (wlfspy(j,k,l).eq.0. .or.
     #             wlfspy(j,k,l).ge.2.*frspce)
     #             wlfspy(j,k,l) = frspce

             endif

c.....Y-moments exceed cracking moment but not ultimate strength of wall.

             if (wlymnt(j,k,l).ge.crkmnt .and.
     #          wlymnt(j,k,l).lt.wlusty) then

                efmnty = (crkmnt/wlymnt(j,k,l))**3.*cnmnti+
     #                   (1.-(crkmnt/wlymnt(j,k,l))**3.)*crmtiy
                act = cmthk(2)+(ratmod-1.)*(startn+starcm)
                strsmy = wlymnt(j,k,l)*axneuy/efmnty+wlcmfy(j,k,l)/act
                stltny = ratmod*(wlymnt(j,k,l)*(wldsty-axneuy)/efmnty-
     #                   wlcmfy(j,k,l)/act)
                axsney = wldsty/(stltny/stlmod+strsmy/conmod)*
     #                   (stltny/stlmod+csstrn)+tencvy(2)
                betay = axsney/(axsney-tencvy(2))
                wlfdpy(j,k,l) = axsney
                frcwdy = wlfspy(j,k,l)*(stltny/stlmod*betay+csstrn)

             endif

c.....Y-moments exceed ultimate strength of wall.

             if (wlymnt(j,k,l).ge.wlusty.and.wlfdpy(j,k,l).lt.
     #          cmthk(2)) then

                wlfdpy(j,k,l) = cmthk(2)
                frcwdy = amin1((stlyld/stlmod+csstrn)*wlfspy(j,k,l),
     #                   3.e-3*wlfspy(j,k,l))

             endif

c.....Calculate cracking due to corrosion once it begins.

             if (icrflg(2).eq.1 .and. (j+k+l).eq.3)
     #          call ccrack(2,iyear)

c.....Calculate average crack characteristics for first wall. Calculations
c.....are performed for an exterior and interior wall. The exterior wall
c.....is denoted by a "1" in the second array position; the interior wall
c.....is denoted by a "2" in the second array position.

             if (cmthk(2) .eq. 0.) then

                do 200 m=1,lyr

                   do 100 n=1,2

                      wlaper(m,n) = 0.
                      wlfrac(m,n) = 0.

  100              continue

  200           continue
                return

             else

                fmax = .75*cmthk(2)
                depth = amax1(wlfdpx(j,k,l),sfrcdx,crfrcd(2))

                if (depth .ge. fmax) then

                   tmp1 = 0.
                   tmp2 = 0.
                   tmp3 = 0.
                   krkx = krkx+1
                   if (wlfspx(j,k,l) .gt. 0.) tmp1 = clwid/10./
     #                wlfspx(j,k,l)
```

```
                     if (crfrcs(2) .gt. 0.) tmp2 = clhght/10./crfrcs(2)
                     if (sfrcsx .gt. 0.) tmp3 = clwid/10./sfrcsx
                     tmp = tmp1+tmp2+tmp3
                     wlaper(j,1) = wlaper(j,1)+(frcwdx*tmp1+crfrcw(2)*
     #                             tmp2+sfrcwx*tmp3)/tmp
                     wlaper(j,2) = wlaper(j,2)+crfrcw(2)*tmp2
                     wlfrac(j,1) = wlfrac(j,1)+cmthk(2)*clhght/10.*
     #                             (frcwdx*tmp1+sfrcwx*tmp3)

                  endif

                  depth = amax1(wlfdpy(j,k,1),sfrcdy,crfrcd(2),cfrcdp)

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     tmp4 = 0.
                     krky = krky+1
                     if (wlfspy(j,k,1) .gt. 0.) tmp1 = clhght/10./
     #                  wlfspy(j,k,1)
                     if (crfrcs(2) .gt. 0.) tmp2 = clwid/10./crfrcs(2)
                     if (sfrcsy .gt. 0.) tmp3 = clhght/10./sfrcsy
                     if (cfrcsp .gt. 0.) tmp4 = clhght/10./cfrcsp
                     tmp = tmp1+tmp2+tmp3+tmp4
                     wlaper(j,1) = wlaper(j,1)+(frcwdy*tmp1+crfrcw(2)*
     #                             tmp2+sfrcwy*tmp3+cfrcwd*tmp4)/tmp
                     if (tmp2+tmp4 .gt. 0.) wlaper(j,2) = wlaper(j,2)+
     #                  (crfrcw(2)*tmp2+cfrcwd*tmp4)/(tmp2+tmp4)
                     wlfrac(j,1) = wlfrac(j,1)+cmthk(2)*clwid/10.*
     #                             (frcwdy*tmp1+sfrcwy*tmp3+cfrcwd*tmp4)
                     wlfrac(j,2) = wlfrac(j,2)+cmthk(2)*clwid/10.*
     #                             cfrcwd*tmp4

                  endif

               endif

  300       continue

  400    continue

         if (wlfrac(j,1).gt.0. .or. wlfrac(j,2).gt.0.) icrack(2) = 1
         wlfrac(j,1) = wlfrac(j,1)+crfrac(2)
         wlfrac(j,2) = wlfrac(j,2)+crfrac(2)

  500 continue

      do 700 j=1,lyr

         do 600 i=1,2

            if (wlfrac(j,i) .gt. 0.) then

               wlfrac(j,i) = wlfrac(j,i)/(cmthk(2)*clwid*clhght)
               wlaper(j,i) = wlaper(j,i)/(krkx+krky)*2.54

            endif

  600    continue

  700 continue

c.....Calculate cracking due to shear for second wall for all layers of casks.

      do 1200 j=1,lyr

         krkx = 0
         krky = 0
         w2frac(j,1) = 0.
         w2frac(j,2) = 0.
         w2aper(j,1) = 0.
```

```
         w2aper(j,2) = 0.

         do 1100 k=1,11

            do 1000 l=1,11

               frcwdx = 0.
               frcwdy = 0.

               if (w2xshr(j,k,l) .ge. shrstx) then

                  if (w2xmnt(j,k,l) .gt. 0.) then

                     tmp = amin1(w2xshr(j,k,l)/w2xmnt(j,k,l)*wldstx,1.)
                     vcr = amin1((1.9*sqrt(comstr)+2500.*
     #                     startn/wldstx*tmp)*
     #                     wldstx,3.5*sqrt(comstr)*wldstx)

                  else

                     vcr = 3.5*sqrt(comstr)*wldstx

                  endif

                  if(w2xshr(j,k,l) .ge. vcr) then

                     sfrcdx = cmthk(2)
                     sfrcwx = 0.013
                     sfrcsx = cllth/10.

                  endif

               else

                  sfrcdx = 0.
                  sfrcwx = 0.
                  sfrcsx = 0.

               endif

c.....Calculate fracture characteristics for horizontal (x) direction due to
c.....bending.

               if (w2xmnt(j,k,l).ge.crkmtw) then

                  if (tencvx(2).eq.0. .and. w2fspx(j,k,l).eq.0.) then

                     q = stlrad(2)**2*1.571/(stlspc(2)*otncvx(2))
                     if (stlrad(2) .lt. 1.e-15) q = ostlrd(2)**2*1.571/
     #                  (stlspc(2)*otncvx(2))

                  elseif (stlrad(2).lt.1.e-15 .and.
     #                   w2fspx(j,k,l).eq.0.) then

                     q = ostlrd(2)**2*1.571/(stlspc(2)*tencvx(2))

                  elseif (tencvx(2).gt.0. .and. stlrad(2).ge.1.e-15)
     #                   then

                     q = stlrad(2)**2*1.571/(stlspc(2)*tencvx(2))

                  endif

                  if (stlrad(2) .ge. 1.e-15) then

                     frspce = 0.5*xk*sqrt(2.*stlrad(2)*stlspc(2)/q)

                  elseif (stlrad(2).lt.1.e-15 .and. w2fspx(j,k,l).eq.0.)
     #                   then

                     frspce = 0.5*xk*sqrt(2.*ostlrd(2)*stlspc(2)/q)

                  endif
```

```
               if (w2fspx(j,k,l).eq.0. .or. w2fspx(j,k,l).ge.
     #            2.*frspce)w2fspx(j,k,l) = frspce

            endif

c.....X-moments exceed cracking moment but not ultimate strength of wall.

            if (w2xmnt(j,k,l).ge.crkmtw .and.
     #         w2xmnt(j,k,l).lt.wlustx) then

               efmntx = (crkmtw/w2xmnt(j,k,l))**3.*cnmnti+(1.-
     #                  (crkmtw/w2xmnt(j,k,l))**3.)*crmtix
               strsmx = w2xmnt(j,k,l)*axneux/efmntx
               stltnx = ratmod*w2xmnt(j,k,l)*(wldstx-axneux)/efmntx
               axsnex = wldstx/(stltnx/stlmod+strsmx/conmod)*
     #                  (stltnx/stlmod+csstrn)+tencvx(2)
               betax = axsnex/(axsnex-tencvx(2))
               w2fdpx(j,k,l) = axsnex
               frcwdx = w2fspx(j,k,l)*(stltnx/stlmod*betax+csstrn)

            endif

c.....X-moments exceed ultimate strength of wall.

            if (w2xmnt(j,k,l).ge.wlustx.and.w2fdpx(j,k,l).lt.
     #         cmthk(2)) then

               w2fdpx(j,k,l) = cmthk(2)
               frcwdx = amin1((stlyld/stlmod+csstrn)*
     #                  w2fspx(j,k,l),3.e-3*w2fspx(j,k,l))

            endif

c.....Perform cracking calculations for second wall in vertical (y) direction.
c.....Calculate ultimate strength for vertical (y) direction.

            crkmnt = cnmnti/(0.5*cmthk(2))*(rupmod+w2cmfy(j,k,l)/
     #               cmthk(2))
            strred = 0.9-0.2*w2cmfy(j,k,l)*1.7/amin1(0.1*comstr*
     #               cmthk(2),pb)
            ecmblk = (startn*stlyld+w2cmfy(j,k,l)/strred)/
     #               (0.85*comstr)
            wlusty = amax1(crkmnt,strred*.085*comstr*ecmblk*
     #               (wldsty-ecmblk/2.))

c.....Shear cracking calculations for vertical (y) direction. Calculation
c.....differs from those for horizontal (x) direction as there is also
c.....compressive force.

            mm = w2ymnt(j,k,l)-w2cmfy(j,k,l)*(4.*cmthk(2)-wldsty)/8.
            tmp = 3.5*sqrt(comstr)*wldsty*(1.+w2cmfy(j,k,l)/(500.*
     #            cmthk(2)))**.5

            if(mm .gt. 0.) then

               vcr = amin1((1.9*sqrt(comstr)+2500.*startn/wldsty*
     #               w2yshr(j,k,l)*wldsty/mm)*wldsty,tmp)

            else

               vcr = tmp

            endif

            if(abs(w2yshr(j,k,l)) .ge. vcr) then

               sfrcdy = cmthk(2)
               sfrcwy = 0.013
               sfrcsy = clhght/10.

            else

               sfrcdy = 0.
```

```
                  sfrcwy = 0.
                  sfrcsy = 0.

               endif

c.....Calculate cracking of interior walls due to compression. No cracking
c.....due to bending is assumed for these walls. When cracking occurs there
c.....is a single (horizontal) crack per section.

               if(w2cmfy(j,k,l) .ge. wlustc) then

                  cfrcdp = cmthk(2)
                  cfrcwd = .003*clhght/10.
                  cfrcsp = clhght/10.

               else

                  cfrcdp = 0.
                  cfrcwd = 0.
                  cfrcsp = 0.

               endif

c.....Calculate fracture characteristics.

               if(w2ymnt(j,k,l).ge.crkmnt) then

                  if (tencvy(2).eq.0. .and. w2fspy(j,k,l).eq.0.) then

                     q = stlrad(2)**2*1.571/(stlspc(2)*otncvy(2))
                     if (stlrad(2) .lt. 1.e-15) q = ostlrd(2)**2*1.571/
     #                  (stlspc(2)*otncvy(2))

                  elseif (stlrad(2).lt.1.e-15 .and.
     #                   w2fspy(j,k,l).eq.0.) then

                     q = ostlrd(2)**2*1.571/(stlspc(2)*tencvy(2))

                  elseif (tencvy(2).gt.0. .and.
     #                   stlrad(2).ge.1.e-15) then

                     q = stlrad(2)**2*1.571/(stlspc(2)*tencvy(2))

                  endif

                  if (stlrad(2) .ge. 1.e-15) then

                     frspce = 0.5*xk*sqrt(2.*stlrad(2)*stlspc(2)/q)

                  elseif (stlrad(2).lt.1.e-15 .and.
     #                   w2fspy(j,k,l).eq.0.) then

                     frspce = 0.5*xk*sqrt(2.*ostlrd(2)*stlspc(2)/q)

                  endif

                  if (w2fspy(j,k,l).eq.0. .or.
     #               w2fspy(j,k,l).ge.2.*frspce)
     #               w2fspy(j,k,l) = frspce

               endif

c.....Y-moments exceed cracking moment but not ultimate strength of wall.

               if (w2ymnt(j,k,l).ge.crkmnt .and.
     #            w2ymnt(j,k,l).lt.wlusty) then

                  efmnty = (crkmnt/w2ymnt(j,k,l))**3.*cnmnti+(1.-
     #                     (crkmnt/w2ymnt(j,k,l))**3.)*crmtiy
                  act = cmthk(2)+(ratmod-1.)*(startn+starcm)
                  strsmy = w2ymnt(j,k,l)*axneuy/efmnty+
     #                     w2cmfy(j,k,l)/act
                  stltny = ratmod*(w2ymnt(j,k,l)*(wldsty-axneuy)/
```

```
     #                     efmnty-w2cmfy(j,k,1)/act)
                  axsney = wldsty/(stltny/stlmod+strsmy/conmod)*
     #                     (stltny/stlmod+csstrn)+tencvy(2)
                  betay = axsney/(axsney-tencvy(2))
                  w2fdpy(j,k,1) = axsney
                  frcwdy = w2fspy(j,k,1)*(stltny/stlmod*betay+csstrn)

               endif

c.....Y-moments exceed ultimate strength of wall.

               if (w2ymnt(j,k,1).ge.wlusty.and.w2fdpy(j,k,1).lt.
     #            cmthk(2)) then

                  w2fdpy(j,k,1) = cmthk(2)
                  frcwdy = amin1((stlyld/stlmod+csstrn)*
     #                     w2fspy(j,k,1),3.e-3*w2fspy(j,k,1))

               endif

c.....Calculate cracking due to corrosion once it begins.

               if (icrflg(2).eq.1 .and. (j+k+1).eq.3)
     #            call ccrack(2,iyear)

c.....Calculate average crack characteristics for second wall. Calculations
c.....are performed for an exterior and interior wall. The exterior wall
c.....is denoted by a "1" in the second array position; the interior wall
c.....is denoted by a "2" in the second array position.

               if (cmthk(2) .eq. 0.) then

                  do 900 m=1,lyr

                     do 800 n=1,2

                        w2aper(m,n) = 0.
                        w2frac(m,n) = 0.

  800                continue

  900             continue

                  return

               else

                  fmax = .75*cmthk(2)
                  depth = amax1(w2fdpx(j,k,1),sfrcdx,crfrcd(2))

                  if (depth .ge. fmax) then

                     tmp1 = 0.
                     tmp2 = 0.
                     tmp3 = 0.
                     krkx = krkx+1
                     if (w2fspx(j,k,1) .gt. 0.)
     #                  tmp1 = cllth/10./w2fspx(j,k,1)
                     if (crfrcs(2) .gt. 0.) tmp2 = clhght/10./crfrcs(2)
                     if (sfrcsx .gt. 0.) tmp3 = cllth/10./sfrcsx
                        tmp = tmp1+tmp2+tmp3
                        w2aper(j,1) = w2aper(j,1)+(frcwdx*tmp1+
     #                                crfrcw(2)*tmp2+
     #                                sfrcwx*tmp3)/tmp
                        w2aper(j,2) = w2aper(j,2)+crfrcw(2)*tmp2
                        w2frac(j,1) = w2frac(j,1)+cmthk(2)*clhght/10.*
     #                                (frcwdx*tmp1+sfrcwx*tmp3)

                  endif

                  depth = amax1(w2fdpy(j,k,1),sfrcdy,crfrcd(2),cfrcdp)

                  if (depth .ge. fmax) then
```

```
                    tmp1 = 0.
                    tmp2 = 0.
                    tmp3 = 0.
                    tmp4 = 0.
                    krky = krky+1
                    if (w2fspy(j,k,1) .gt. 0.)
     #                 tmp1 = clhght/10./w2fspy(j,k,1)
                    if (crfrcs(2) .gt. 0.) tmp2 = cllth/10./crfrcs(2)
                    if (sfrcsy .gt. 0.) tmp3 = clhght/10./sfrcsy
                    if (cfrcsp .gt. 0.) tmp4 = clhght/10./cfrcsp
                    tmp = tmp1+tmp2+tmp3+tmp4
                    w2aper(j,1) = w2aper(j,1)+(frcwdy*tmp1+crfrcw(2)*
     #                            tmp2+sfrcwy*tmp3+cfrcwd*tmp4)/tmp
                    if (tmp2+tmp4 .gt. 0.) w2aper(j,2) =
     #                 w2aper(j,2)+(crfrcw(2)*
     #                 tmp2+cfrcwd*tmp4)/(tmp2+tmp4)
                    w2frac(j,1) = w2frac(j,1)+cmthk(2)*cllth/10.*
     #                            (frcwdy*tmp1+
     #                            sfrcwy*tmp3+cfrcwd*tmp4)
                    w2frac(j,2) = w2frac(j,2)+cmthk(2)*cllth/10.*
     #                            cfrcwd*tmp4

                 endif

              endif

1000       continue

1100    continue

        if (w2frac(j,1).gt.0. .or. w2frac(j,2).gt.0.) icrack(2) = 1
        w2frac(j,1) = w2frac(j,1)+crfrac(2)
        w2frac(j,2) = w2frac(j,2)+crfrac(2)

1200 continue

     do 1400 j=1,lyr

        do 1300 i=1,2

           if (w2frac(j,i) .gt. 0.) then

              w2frac(j,i) = w2frac(j,i)/(cmthk(2)*cllth*clhght)
              w2aper(j,i) = w2aper(j,i)/(krkx+krky)*2.54
              icrack(2) = 1

           endif

1300    continue

1400 continue

     return
     end
```

Exhibit D.2. Computer code listing for SOURCE2

```
      program source2

c.....SOURCE2 Version 2.0.

c.....Reference: Icenhour, A. S. and M. L. Tharp, "User's Manual for
c.....the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating
c.....Low-Level Radioactive Waste Disposal Facility Source Terms
c.....(Version 2.0)," ORNL/TM-13035, Oak Ridge National Laboratory,
c.....Oak Ridge, TN, 1996.

c-----------------------------------------------------------------------
c     Program driver
c
c     Calls:  input, output, sar2, concrete, sfl, srf, swl, wfl,
c             wrf, wwl, leach
c-----------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #      crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #      icrack(3),icrflg(3),ifail(3),isavel,isave2,ispl(3),ph(2,3),
     #      slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #      co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #      yngmod
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #      flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #      otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #      stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #      tencvy(2,3),wstdns,wsthk,wstht

c.....Read input data and perform preliminary calculations for simulation.

      call input(0, nyears)

c.....Opens the files which have been selected to provide summaries
c.....of the simulation results.

      call output(0,nyears)

c.....Start annual loop.

      do 100 iyear=1,nyears

c.....Read years when inventories were disposed and associated inventories
c.....and updates water infiltration values.

         call input(iyear, nyears)

c.....   if (mod(iyear,10) .eq. 0) write (*,*) 'year: ',iyear

c.....Calculate time-dependent properties of reinforced concrete.

         time = iyear*365.
         csstrn = amaxl(((time-28.)/(time+7.))*6.7e-4,0.0)
         crpcof = 5.83e-1*(time**0.6/(10.+time**0.6))

c.....Perform structural analysis for disposal technology.

         if (iyear .eq. 1) call sar2

         if (stlmod .ne. 0. .or. stlyld .ne. 0. .or.
     #       yngmod .ne. 0.) then

c.....Perform concrete deterioration analysis. Analysis results are
c.....used in cracking analysis until silo and/or well have at least
c.....one cracked or failed member. Degradation analysis is continued
c.....for entire simulation.
```

```
            call concrete(iyear)

c.....Perform cracking analysis for silo and well technologies. Analysis
c.....is performed until all disposal units have at least one cracked or
c.....failed member.

            if ((idflag.eq.1 .or. idflag.eq.3) .and. isave1.eq.0) then

               if (cmthk(1,1) .gt. 0.) call srf(attk(1,1),iyear)
               if (cmthk(1,2) .gt. 0.) call swl(attk(1,2),iyear)
               if (cmthk(1,3) .gt. 0.) call sfl(attk(1,3),iyear)

            endif

         else

            icrack(1) = 1
            icrack(2) = 1
            icrack(3) = 1
            frac(1) = 0.
            frac(2) = 0.
            frac(3) = 0.
            aper(1) = 0.
            aper(2) = 0.
            aper(3) = 0.

         endif

         if ((idflag.eq.2 .or. idflag.eq.3) .and. isave2.eq.0) then

            call wrf(attk(2,1),iyear)
            call wwl
            call wfl(attk(2,3),iyear)

         endif

c.....Monitor degree to which silo and well have cracked or failed.

         isave1 = 0
         isave2 = 0

         do 200 i=1,3

            if(idflag.eq.1 .or. idflag.eq.3) isave1 = isave1+icrack(i)
            if(idflag .gt. 1) isave2 = isave2+ifail(i)

200      continue

c.....Calculate advective and diffusive release rates.

         call leach(iyear,nyears)

c.....Output simulation results.

         call output(iyear, nyears)

100   continue

      stop
      end

      block data sorbd

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),ic1(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/miscel/acoef,bcoef,dpm(12)

      data isave1,isave2/0,0/
      data attk,icrack,ifail/6*1.,3*0,3*0/
      data ic1,ico2,icrflg/9*0/
      data ispl/3*0/
```

```
      data dpm/31.,28.25,31.,30.,31.,30.,31.,31.,30.,31.,30.,31./
      data annprc/0./

      end

      subroutine caoh(iyear)

c-----------------------------------------------------------------------
c     Called by concrete
c
c     Calculates loss of concrete strength and reduction in pH of concrete
c     due to leaching of Ca(OH)2.
c
c     Calls: none
c-----------------------------------------------------------------------

      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht

      dimension tlch1(2,3),tlch2(2,3)

      real*8 az1,az2,derf

      data tlch1,tlch2/6*0.,6*0./

c.....Calculate retardation factor for Ca(OH)2 leaching by diffusion.

      if (iyear .eq. 1) then

        rf = 1.+ccdns*cacon/amin1(casol,cap)/ccpor

c.....Calculate Langelier or calcium carbonate saturation index and ionic
c.....concentrations of Ca(OH)2, Mg2++, and CO3-.

        xmg2 = amin1(xmg2,xmgsol)
        co3 = amin1(co3,crbsol)
        cap = amin1(cap,casol)
        casum = xmg2+co3+cap
        pk = 2.268712-1.122e-2*temp+3.91e-5*temp**2+1.007e-3*tds -
     #       6.3e-7*tds**2
        phs = pk+log10(1/cagw)+log10(1/co3)
        xli = phgw - phs

      endif

c.....If initial pH is greater than 12.5, calculate rate of loss of NaOH
c.....and KOH and consequent decline in pH. Pore liquid and solid con-
c.....centrations are assumed to be equal.

      if(idflag.eq.1 .or. idflag.eq.3) then

         do 300 l=1,3

            if(ph(1,l) .gt. 12.5) then

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by advection.

               if(ommthk(1,l) .ne. 0.)then
```

```
                 tlch1(1,1) = amin1(1.,tlch1(1,1)+annprc/
     #                           2.54/ommthk(1,1))

              else

                 tlch1(1,1) = 0.

              endif

              xlch1 = 1.-tlch1(1,1)

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by diffusion.

              xlch2 = 0.
              xthk = ommthk(1,1)/39.37

              do 100 i=0,10

                 az1 = (i*xthk/21.+xthk/2.)/(2.*
     #                 sqrt(dfalk*iyear*3.15e7))
                 az2 = (i*xthk/21.-xthk/2.)/(2.*
     #                 sqrt(dfalk*iyear*3.15e7))
                 ft = 0.5*(derf(az1)-derf(az2))

                 if (i .eq. 0) then

                    xlch2 = xlch2+ft/21.

                 else

                    xlch2 = xlch2+2.*ft/21.

                 endif

  100         continue

c.....Determine total amount of alkalis lost from concrete.

              alklch = 2.-xlch1-xlch2
              ph(1,1) = amax1(12.5,phbeg-(phbeg-12.5)*alklch)

           else

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by advection. Groundwater leaching is assumed only if
c.....the Langelier index is negative, indicating the water is capable
c.....of dissolving Ca(OH)2.

              if (xli .lt. 0.) then

                 if(ommthk(1,1) .ne. 0.)then

                    tlch2(1,1) = amin1(1.,tlch2(1,1)+annprc/2.54*casum/
     #                              (ommthk(1,1)*cacon))
                 else

                    tlch2(1,1) = 0.

                 endif

                 xlch3 = 1.-tlch2(1,1)

              else

                 xlch3 = 1.

              endif

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by diffusion.

              xlch4 = 0.
```

```
              xthk = ommthk(1,1)/39.37

              do 200 i=0,10

                 az1 = (i*xthk/21.+xthk/2.)/(2.*sqrt(dfcaoh/rf*iyear*
     #                   3.15e7))
                 az2 = (i*xthk/21.-xthk/2.)/(2.*sqrt(dfcaoh/rf*iyear*
     #                   3.15e7))
                 ft = 0.5*(derf(az1)-derf(az2))

                 if (i .eq. 0) then

                    xlch4 = xlch4+ft/21.

                 else

                    xlch4 = xlch4+2.*ft/21.

                 endif

  200         continue

c.....Determine total amount of Ca lost from concrete.

              calch = 2.-xlch3-xlch4

c.....Adjust Ca concentration and recalculate Ca:Si ratio

              ca = cacon*(1.-calch)
              ca_si = ca/si

c.....Calculate average pH for concrete as a function of Ca:Si following
c.....the loss of NaOH and KOH.

              ph(1,1) = amin1(12.5,8.83533+3.143848*ca_si-0.6617*
     #                    ca_si**2)

c.....Calculate loss in strength for concrete members due leaching. If
c.....concrete member thickness goes to zero, strength is set to zero.

              x = calch

              if (cmthk(1,1) .gt. 0.) then

                 attk(1,1) = amax1(0.,1.-0.015*x/0.01)
                 if (attk(1,1) .eq. 0.) cmthk(1,1) = 0.

              else

                 attk(1,1) = 0.

              endif

            endif

  300     continue

        endif

        if(idflag .gt. 1) then

          do 600 l=1,3,2

            if(ph(2,l) .gt. 12.5) then

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by advection.

              tlch1(2,l) = amin1(1.,tlch1(2,l)+annprc/2.54/ommthk(2,l))
              xlch1 = 1.-tlch1(2,l)

c.....Calculate fraction of alkalis remaining in concrete following
c.....leaching by diffusion.
```

```
               xlch2 = 0.
               xthk = ommthk(2,1)/39.37

               do 400 i=0,10

                  az1 = (i*xthk/21.+xthk/2.)/(2.*
     #                  sqrt(dfalk*iyear*3.15e7))
                  az2 = (i*xthk/21.-xthk/2.)/(2.*
     #                  sqrt(dfalk*iyear*3.15e7))
                  ft = 0.5*(derf(az1)-derf(az2))

                  if (i .eq. 0) then

                     xlch2 = xlch2+ft/21.

                  else

                     xlch2 = xlch2+2.*ft/21.

                  endif

  400          continue

c.....Determine total amount of alkalis lost from concrete.

               alklch = 2.-xlch1-xlch2
               ph(2,1) = amax1(12.5,phbeg-(phbeg-12.5)*alklch)

            else

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by advection. Groundwater leaching is assumed only if
c.....the Langelier index is negative, indicating the water is capable
c.....of dissolving Ca(OH)2.

               if (xli .lt. 0.) then

                  xlch4 = 0.
                  tlch2(2,1) = amin1(1.,tlch2(2,1)+annprc/2.54*casum/
     #                         (ommthk(2,1)*cacon))
                  xlch3 = 1.-tlch2(2,1)

               else

                  xlch3 = 1.

               endif

c.....Calculate fraction of Ca(OH)2 remaining in concrete following
c.....leaching by diffusion.

               xthk = ommthk(2,1)/39.37

               do 500 i=0,10

                  az1 = (i*xthk/21.+xthk/2.)/(2.*sqrt(dfcaoh/rf*iyear*
     #                  3.15e7))
                  az2 = (i*xthk/21.-xthk/2.)/(2.*sqrt(dfcaoh/rf*iyear*
     #                  3.15e7))
                  ft = 0.5*(derf(az1)-derf(az2))

                  if (i .eq. 0) then

                     xlch4 = xlch4+ft/21.

                  else

                     xlch4 = xlch4+2.*ft/21.

                  endif

  500          continue
```

```
c.....Determine total amount of Ca lost from concrete.

               calch = 2.-xlch3-xlch4

c.....Adjust Ca concentration and recalculate Ca:Si ratio

               ca = cacon*(1.-calch)
               ca_si = ca/si

c.....Calculate average pH for concrete as a function of Ca:Si following
c.....the loss of NaOH and KOH.

               ph(2,1) = amin1(12.5,8.83533+3.143848*ca_si-0.6617*
     #                   ca_si**2)

c.....Calculate equivalent depth of Ca(OH)2 loss and loss in strength for
c.....roof, floor, and internal and external walls.

               x = calch

               if (cmthk(2,1) .gt. 0.) then
                  attk(2,1) = amax1(0.,1.-0.015*x/0.01)
                  if (attk(2,1) .eq. 0.) cmthk(2,1) = 0.

               else

                  attk(2,1) = 0.

               endif

            endif

  600    continue

      endif

      return
      end

      subroutine ccrack(i,iyear)

c-----------------------------------------------------------------------------
c     Called by: source2
c
c     Calculates cracking due to corrosion of reinforcing steel.
c
c     Calls: none
c-----------------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht

      data attack,stlpsn/1.,.30/

      time = 365.*iyear
      comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
      cdtstr = 4.*sqrt(comstr)
      crpcof = 5.83e-1*(time**0.6/(10.+time**0.6))
      csstrn = amax1(((time-28.)/(time+7.))*6.7e-4,0.0)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      tmp = amin1(tencvx(1,i),tencvy(1,i),comcvx(1,i),comcvy(1,i))+
     #      ostlrd(i)
```

```
      if (tmp .le. stlrad(i)+stlcor(i)) return


c.....Calculate internal pressure due to corrosion.

      pstl = (stlrad(i)+stlcor(i)-ostlrd(i))/ostlrd(i)*1./
     #       ((1.-stlpsn)/stlmod+((1.-conpsn)*ostlrd(i)**2+
     #       (1.+conpsn)*tmp**2)/(conmod*(tmp**2-ostlrd(i)**2)))

c.....Calculate maximum stress.

      conrad = stlrad(i)+stlcor(i)-pstl*ostlrd(i)*(1.-stlpsn)/stlmod
      ctstrs = 4.*pstl*conrad**2/(tmp**2-conrad**2)
      ststrs = pstl*(tmp**2+conrad**2)/(tmp**2-conrad**2)
      xzero = amax1(0.5*stlspc(i),sqrt(tmp**2-ostlrd(i)**2))

c.....Determine whether spalling has occurred. If it has, all of the
c.....concrete cover is assumed to be destroyed and all steel is exposed.

      if (ctstrs.gt.cdtstr .and. ststrs.gt.cdtstr .and.
     #   ctstrs.gt.ststrs) then

         ispl(i) = 1
         cmthk(1,i) = 0.
         tencvx(1,i) = 0.
         tencvy(1,i) = 0.
         comcvx(1,i) = 0.
         comcvy(1,i) = 0.

c.....Cracking extends through concrete cover to steel along steel members.

      elseif (ctstrs .gt. cdtstr) then

         crfrcd(i) = cmthk(1,i)-2.*stlrad(i)
         crfrcw(i) = 2.*xzero*(cdtstr/conmod+csstrn)
         crfrcs(i) = stlspc(i)
         ttllth = 0.

         do 100 i=1,int(silrad/stlspc(i))+1

            ttllth = ttllth+2.*sqrt(silrad**2-((i-1)*stlspc(i))**2)

 100     continue

         crfrac(i) = crfrcw(i)*crfrcd(i)*ttllth

c.....Calculate potential for cracking through concrete cover versus
c.....internal cracking only (i.e., not through the entire concrete cover).

      elseif (ststrs .gt. cdtstr) then

         rrr = sqrt((conrad**2*tmp**2*pstl)/(cdtstr*(tmp**2-
     #        conrad**2)-conrad**2*pstl))

         if (rrr-conrad .ge. 0.5*(tmp-conrad)) then

            crfrcd(i) = cmthk(1,i)-2.*stlrad(i)
            crfrcw(i) = 2.*xzero*(cdtstr/conmod+csstrn)
            crfrcs(i) = stlspc(i)
            ttllth = 0.

            do 200 i=1,int(silrad/stlspc(i))+1

               ttllth = ttllth+2.*sqrt(silrad**2-((i-1)*stlspc(i))**2)

 200        continue

            crfrac(i) = crfrcw(i)*crfrcd(i)*ttllth

         endif

      endif
```

```
      return
      end

      subroutine concrete(iyear)

c-----------------------------------------------------------------------------
c     Called by: source2
c
c     Calculates degradation of concrete with time.
c
c     Calls: caoh, corrode, sulfate
c-----------------------------------------------------------------------------

c.....Calculate loss of concrete strength and changes in pH of concrete
c.....structure due to leaching of Ca(OH)2.

      call caoh(iyear)

c.....Calculate loss of concrete due to sulfate attack.

      call sulfate(iyear)

c.....Calculate corrosion of steel reinforcement.

      call corrode(iyear)

      return
      end

      subroutine corrode(iyear)

c-----------------------------------------------------------------------------
c     Called by concrete
c
c     Calculates rate of corrosion of steel reinforcement in concrete, and
c     rates of loss of corrugated steel in silo and iron wall in well.
c
c     Calls: none
c-----------------------------------------------------------------------------

      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isavel,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/failure/eftl,deft,wftl,dwft,xlftl,dlft
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr

      dimension corvol(3)

      real*8 az,derf

      data crbcof,eff,pi/0.,0.,3.141592653589793/

c.....Calculate thickness of corrugated steel liner in silo and well wall.

      if(idflag.eq.1 .or. idflag.eq.3) then

         if (dlft .gt. 0) then

            aux = float(iyear)

            if(aux.gt.xlftl .and. aux.le.xlftl+dlft) then
```

```
               sttkcm = amaxl(0.,osttkc*(1.-(iyear-xlftl)/dlft))
               sttktn = amaxl(0.,osttkt*(1.-(iyear-xlftl)/dlft))

            endif

         endif

      endif

      if(idflag .gt. 1) then

         if (dwft .gt. 0) then

            aux = float(iyear)

            if(aux.gt.wftl .and. aux.le.wftl+dwft) then

               cmthk(2,2) = amaxl(0.,ommthk(2,2)*(1.-(iyear-wftl)/dwft))

            endif

         endif

      endif

c.....Current configuration of containment wells does not require completion
c.....of rest of corrosion analysis unless we are dealing with multiple
c.....containment wells. In that case, the steel reinforcement calculations
c.....are performed for the silo only.

      if(idflag .ne. 2) then

c.....Calculate failure function of epoxy coating on steel reinforcement.
c.....If no failure has occurred, corrosion does not occur.

         if (deft .gt. 0) then

            aux = float(iyear)

            if(aux .gt. eftl) then

               eff = aminl(1.,(aux-eftl)/deft)

            else

              return

            endif

         else

            eff = 1.

         endif

c.....Determine depth of concrete carbonation based on Crank formulation.

         if(crbcof .eq. 0.) then

            dhydr = 0.4 + 0.5*wcr
            ccon = ccon*dhydr
            cl = co2 + ccon
            crbcof = sqrt((cl/ccon - 1.)*4.*dfco2*3.15e7/sqrt(pi))

         endif

         aux=iyear
         dpcrb = crbcof*sqrt(aux)

c.....Check carbonation depth and concrete cover thickness for structural
c.....components.

         do 100 i=1,3
```

```
          if (stlrad(i) .gt. 0.) then

             tmp = (amin1(comcvx(1,i),tencvx(1,i),comcvy(1,i),
     #             tencvy(1,i))+ostlrd(i))/39.37

             if(dpcrb.ge.tmp .and. icrflg(i).eq.0 .and.
     #         eff.gt.0.) then

                ico2(i) = iyear
                icrflg(i) = 1

             endif

c.....Calculate [OH-] in pore solution based on concrete pH.

             poh = 14.-ph(1,i)
             oh = 1./10.**poh

c.....Calculate chloride ion concentration at steel reinforcement.  If
c.....ratio of chloride concentration to hydroxide ion concentration
c.....exceeds 0.61 corrosion is initiated (Hausman).

             if(icrflg(i) .eq. 0) then

                tmp = (amin1(comcvx(1,i),tencvx(1,i),comcvy(1,i),
     #                tencvy(1,i))+ostlrd(i))/39.37
                aux=iyear
                az = tmp/(2.*sqrt(dfcl*aux*3.15e7))
                clstl = cl-(cl-clcon)*derf(az)

                if(clstl/oh.ge.0.61 .and. eff.gt.0.) then

                   icl(i) = iyear
                   icrflg(i) = 1

                endif

             endif

c.....Upon de-passivation, the rate of corrosion is determined by the
c.....rate of O2 diffusion to steel. If concrete cover is gone, corrosion
c.....is assumed to proceed at a rate of .025 cm/yr. Densities of steel
c.....and corrosion product (feo) are 7.86 and 5.70 g/cm**3, respectively.

             if (icrflg(i).eq.1 .and. stlrad(i).gt.0.) then

                tmp = amin1(comcvx(1,i),tencvx(1,i),comcvy(1,i),
     #               tencvy(1,i))+ostlrd(i)

                if(tmp .le. ostlrd(i)) then

                   if (stlrad(i) .gt. .01) then

                      volfe = 3.142*(.02*stlrad(i)-.0001)
                      stlrad(i) = stlrad(i)-0.01

                   else

                      volfe = 3.142*stlrad(i)**2
                      stlrad(i) = 0.

                   endif

                elseif (stlrad(i) .gt. 0.) then

c.....Calculate oxygen flux at the steel reinforcement.

                   stlara = eff*stlspc(i)*6.45e-4        !1./39.39**2. (m2)
                   o2grd = 39.37e3*o2/tmp               !1000.*39.37 (mole/m4)
                   o2flux = dfo2*stlara*o2grd*3.15e7            !(mole/yr)
                   xmole = 7.245*stlrad(i)**2 !pi*7.86*2.54**3/55.85 (mole)
                   o2flux = amin1(o2flux,0.5*xmole)
                   volfe = .867*o2flux              !2*55.85/(7.86*2.54**3)
```

```
                    stlrad(i) = sqrt(amax1(0.,stlrad(i)**2-
     #                          volfe/3.142))

                 endif

                 corvol(i) = corvol(i)+1.77*volfe  !(7.86/5.7)*(71.85/55.85)
                 stlcor(i) = sqrt(stlrad(i)**2+corvol(i)/3.142)-
     #                       stlrad(i)

              endif

           endif

  100    continue

      endif

      return
      end

      function derf(z)

c-----------------------------------------------------------------------------
c     Called by: caoh
c
c     Function used in KOH, NaOH, and Ca(OH)2 diffusion calculations. series
c     expansion for erf(z) found in Abramowitz & Stegun #7.1.5.
c
c     Calls: derfc
c-----------------------------------------------------------------------------

      implicit real*8 (a-h,o-z)

      if(dabs(z).lt.2.) go to 10

      derf=1.-derfc(z)

      go to 50

   10 sum=0.

      if(z.eq.0.) go to 40
      z2=z*z
      zpowr=z
      fac=1.
      n2p1=1
      term=z

      do 20 i=1,30

         sum=sum+term
         zpowr=-zpowr*z2
         fac=fac*i
         n2p1=n2p1+2
         term=zpowr/(fac*n2p1)
         if(dabs(term/sum).lt.1.e-15) go to 30

   20 continue

   30 sum=sum+term
   40 derf=1.128379167095513*sum

   50 return
      end

      function derfc(z)

c-----------------------------------------------------------------------------
c     Called by:. derf
c
c     Function used in KOH, NaOH, and Ca(OH)2 diffusion calculations. based
c     on continued fraction from Abramowitz & Stegun #7.1.14.
c
```

```
c     Calls: none
c------------------------------------------------------------------------------

      implicit real*8 (a-h,o-z)

      if(dabs(z).ge.2.) go to 10

      derfc=1.-derf(z)

      go to 30

   10 xnum=20.

      zab=dabs(z)
      frac=zab

      do 20 i=1,40

         frac=zab+xnum/frac
         xnum=xnum-0.5

   20 continue

      derfc=0.
      if(zab.le.9.3) derfc=dexp(-zab*zab)/(frac*1.772453850905516)
      if(z.lt.0.) derfc=2.-derfc

   30 return
      end

      subroutine flothru(d1,d2,flam,iyear,m,qzero,rel)

c------------------------------------------------------------------------------
c
c        computes the monthly release r(t) of a radioactive
c        contaminant released from a slab of half-thickness a,
c        through a layer of thickness b - a, remaining at time t,
c        given an initial concentration of zero in the
c        outer layer and an initial amount of q0 in the inner
c        layer. subscripts 1 and 2 refer to the inner and outer layers.
c
c        matl   - label for material (20 characters maximum)
c        d1, d2 - diffusion coefficients (cm**2/sec)
c        a      - inner layer half-thickness (cm)
c        thk    - outer layer thickness (cm)
c        t      - time (sec)
c        flam   - decay constant (sec**-1)
c        qzero  - initial amount in inner layer (gm)
c        rel    - monthly release (gm)
c        v      - contains f (x), n = -1, 0, ..., 81
c                           n
c                 such that
c                  n
c                 i erfc  (x) = (2/sqrt(pi))*(exp(-x**2))*v(n)/v(-1)
c
c     Reference: Icenhour, A. S. and M. L. Tharp, "User's Manual for
c     the SOURCE1 and SOURCE2 Computer Codes: Models for Evaluating
c     Low-Level Radioactive Waste Disposal Facility Source Terms
c     (Version 2.0),"  ORNL/TM-13035, Oak Ridge National Laboratory,
c     Oak Ridge, TN, 1996.
c------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      parameter (maxnuc = 10)
      parameter (r12=1.d0/12.d0)

      common/miscel/acoef,bcoef,dpm(12)

      dimension v(-1:81), ff(30), fx2(30), save(3),
     #          rel(maxnuc,12),resave(maxnuc)

      save resave
```

```
      real*4 acoef,bcoef,dpm,d1,d2,flam,qzero,rel

      data tosrpi/1.128379167095513d0/
      data pi/3.141592653589793d0/
      data secpyr/3.15576d7/
      data resave /maxnuc*0.d0/

      a = acoef
      thk = bcoef
      xkap = d2/d1
      xkap = dsqrt (xkap)
      tk = xkap + xkap
      alfa = thk/(xkap*a)
      apk = alfa + xkap
      b = a + thk
      boa = b/a

c.....Compute first 30 roots of transcendental equation.

      c1 = ((alfa*xkap + 3.d0)*alfa + (xkap*3.d0))*alfa + 1.d0
      c2 = ((alfa + (xkap*3.d0))*alfa + 3.d0)*alfa + xkap
      gam = dsqrt(((alfa**2 + 1.d0)*xkap + alfa + alfa)/xkap)
      x1 = 0.5d0*pi/gam
      x = x1

      do 100 i=1,30

         n = 0

   50    continue

         call fxcal (x, alfa, xkap, f, fp)
         x2 = x - f/fp

         if (abs ((x2 - x)/x2) .gt. 5.d-9) then

            n = n + 1
            x = x2
            if (n .le. 20) go to 50
            write (*, '(1x, a)') 'not converged after 20 iterations'

         endif

         fx2(i) = d1*(x/a)**2
         ax = alfa*x
         c = cos(x)
         s = sin(x)
         ca = cos(ax)
         sa = sin(ax)
         ff(i) = s/((apk*ca*s + boa*sa*c)*x**2)
         f3 = c1*c*sa + c2*s*ca
         gam = -f3/fp

         if (gam .lt. 0.d0) then

            write(*,*) 'fp and f3 have same sign'
            return

         endif

         x = x + pi/dsqrt(gam)

  100 continue

c.....Set a few constants.

      ropk = 1.d0/(1.d0 + xkap)
      c3 = ropk + ropk
      fac1 = (c3 + c3)/a
      fxk = (xkap - 1.d0)*ropk
      if(resave(m) .lt. 0.d0) resave(m) = 0.d0
      resold = resave(m)
```

```
c.....Compute monthly releases for current year.

      do 500 n=iyear,iyear

         yr = float (n - 1)

         do 400 mo=1,12

            decay = 0.d0
            t = (yr + r12*float(mo))*secpyr
            t0 = (r12*float(mo))*secpyr
            arg = log(2.d0)/flam*t0
            decay = exp(-arg)
            arg1 = a/dsqrt(d1*t)

            if (arg1 .lt. 4.d0) then

c.....Sum the series.

               sum = 0.d0

               do 200 k=1,30

                  exx = 0.d0
                  arg = fx2(k)*t
                  if (arg .le. 80.d0) exx = exp (-arg)
                  trm = ff(k)*exx
                  sum = sum + trm

                  if (abs(ff(k)/sum) .gt. 5.d-9) then

                     if (abs(trm/sum) .lt. 5.d-9) go to 210

                  endif

  200          continue

               write(*,*) 'series not converged'

           return

  210          continue

               res = 1.d0 - tk*sum

            else

c.....Use the ierfc series.

               arg2 = 0.5d0*alfa*arg1
               sum = 0.d0
               extold = 0.d0
               sign = 1.d0
               odd = 1.d0
               fxz = 1.d0
               l = 0

               do 300 k=1,30

                  arg = odd*arg2

                  if (arg .le. 1.d0) then

                     res2 = sxierfc (arg)

                  else

                     res2 = 0.d0

                     if (arg .le. 10.d0) then

                        call ierfc (arg, v, 1, 5.d-9)
                        res2 = tosrpi*v(1)*exp(-arg**2)/v(-1)
```

```
                  endif

               endif

               if (res2 .eq. 0.d0) go to 310

               trm = fxz*res2
               sum = sum + sign*trm
               l = l + 1
               save(l) = sum

               if (l .eq. 3) then

c.....Aitken Delta-Squared extrapolation:.

                  d21 = save(2) - save(1)
                  d32 = save(3) - save(2)
                  ext = save(3) + d32**2/(d21 - d32)
                  l = 0

                  if (extold .ne. 0.d0) then

                     if (abs(1.d0 - extold/ext) .lt. 5.d-9) then

                        sum = ext
                        go to 310

                     endif

                  endif

                 extold = ext

               endif

               if (sum .ne. 0.d0) then

                  if (abs(trm/sum) .lt. 5.d-9) go to 310

               endif

               odd = odd + 2.d0
               fxz = fxz*fxk
               sign = -sign

  300       continue

            write (*, '(1x, a, 1p, 2e13.5)') 'trm, sum: ', trm, sum
            write(*,*) 'ierfc series not converged'

            return

  310    continue

         res = fac1*dsqrt(d2*t)*sum

         endif

         rel(m,mo) = qzero*decay*(res - resold)
         resold = res

  400   continue

  500 continue

      resave(m) = resold

      return
      end

      subroutine fxcal (x,alfa,xkap,f,fp)

c-------------------------------------------------------------------------------
```

```
c     Called by: flothru
c
c     Used in diffusion leaching calculations (2 december 1991).
c
c     Calls: none
c------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      ax = alfa*x
      c = cos (x)
      s = sin (x)
      ca = cos (ax)
      sa = sin (ax)
      f = xkap*c*ca - s*sa
      fp = -(xkap + alfa)*s*ca - (1.d0 + xkap*alfa)*c*sa

      return
      end

      subroutine ierfc (x,v,n,tol)

c------------------------------------------------------------------------------
c     Called by: flothru
c
c     Used in diffusion leaching calculations (2 december 1991).
c
c     Calls: none
c------------------------------------------------------------------------------

c     Compute the repeated integrals of the complementary error
c               n
c     function i erfc (x) by backward recurrence and normalization.
c     input parameters:
c       x   - argument
c       n   - maximum value of n
c                                n
c       tol - relative error in i erfc (x)
c       v   - double precision array, dimensioned (-1:81) in the
c             calling program
c     output parameters:
c       v   - contains f (x), n = -1, 0, ..., 81
c                        n
c             such that
c              n
c             i erfc(x) = (2/sqrt(pi))*exp(-x**2))*v(n)/v(1)
c     see W. Gautschi, Recursive Computation of the Repeated
c     Integrals of the Error Function, Mathematics of Computation
c     15, 227-232(1961)
c------------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      common/numb/mmax

      dimension v(-1:81), xt(21)

      xsq = x**2
      l = 0

      do 200 m=21,81,5

         v(m) = 0.d0
         v(m-1) = 10.d0**(-20)
         x2 = x + x
         a = float (m + m)

         do 100 k=m,1,-1

            v(k-2) = a*v(k) + x2*v(k-1)

c.....Watch growth in backward recurrence.  Scale down if needed.
c.....this works for n=1 only.
```

```
           if(v(k-2) .gt. 1.d20) then

              if(k.gt.1) then

                 v(k-1)=v(k-1)/v(k-2)
                 v(k-2)=1.d0

              endif

           endif

           a = a - 2.d0

100     continue

        l = l + 1
        xt(l) = v(n)/v(-1)

        if (l .gt. 1) then

           if (abs(xt(l)/xt(l-1) - 1.d0) .lt. tol) go to 210

        endif

  200 continue

      write (*, '(1x, a, i2, 1x, a, 1p, e11.3)')
     1      'm = 81 not enough for n =', n, ' x =', x
      m = 81
      write (*, '(1x, 1p, 4e15.7)') (xt(n), n=1,l)

  210 continue

      mmax = m

      return
      end

      subroutine input(iyear, nyears)

c------------------------------------------------------------------------------
c     Called by source2
c
c     Reads and checks input data, prints summary, and performs initial
c     calculations.
c
c     Calls: none
c------------------------------------------------------------------------------

      parameter (maxnuc = 10, maxyr = 9999999)

      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/dump/ndump,refyear
      common/failure/eftl,deft,wftl,dwft,xlftl,dlft
      common/files/iprint,fname,iprn1,ifrq1,iprn2,ifrq2,iprn3,ifrq3,
     #       iprn4,ifrq4,iprn5,ifrq5,filenam(7)
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/miscel/acoef,bcoef,dpm(12)
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qsw(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc),xleach(maxnuc)
      common/runid/title
```

```
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr

      dimension qswsav(maxnuc),qswlast(maxnuc),ext(7)

      integer refyear,bgndump, enddump
      character fname*16,filenam*20,ext*4
      character*8 nuclid
      character*80 title
      character*60 wat_inp*60

      data ext/'.inp','.con','.lch','.rch','.lat','.sum','.h2o'/
      data iyr2/0/, ndump/1/, refyear/1/,
     #     bgndump/0/, enddump/0/

      if(iyear .eq. 0) then

c.....Input filename of input file for the simulation.

         write(*,*) ' Enter first extension of filename for opening'
         write(*,*) ' the input file and output files '
         write(*,*) ' associated with this run. '
         read(*,'(a16)') fname

         do il=1,16

            ilen = il

            if(fname(il:il) .eq. ' ')then

               ilen = ilen-1
               go to 10

            endif

         enddo

   10    continue

c.....Create filenames for all input and output files.

         do ifile=1,7

            filenam(ifile) = fname(1:ilen)//ext(ifile)

         enddo

c.....Open input file with name "fname".inp.

c.....Open input data set.

         open(unit=1,file=
     #        filenam(1),status='old')

c.....Read input data set; read title of simulation and simulation options.

         read(1,'(a80)') title
         read(1,'(i10,i5,i2,5(i2,i5))') nyears,idflag,iprint,
     #        iprn1,ifrq1,iprn2,ifrq2,
     #        iprn3,ifrq3,iprn4,ifrq4,iprn5,ifrq5

c.....Set default for output files to print every year.

         if(iprn1 .eq. 0 .and. ifrq1 .eq. 0)ifrq1 = 1
         if(iprn2 .eq. 0 .and. ifrq2 .eq. 0)ifrq2 = 1
         if(iprn3 .eq. 0 .and. ifrq3 .eq. 0)ifrq3 = 1
         if(iprn4 .eq. 0 .and. ifrq4 .eq. 0)ifrq4 = 1
         if(iprn5 .eq. 0 .and. ifrq5 .eq. 0)ifrq5 = 1
```

```
c.....Silo dimensions and design specifications.

         if(idflag.eq.1 .or. idflag.eq.3) then

            read(1,'(2e10.3)') slhght,silrad
            read(1,'(3e10.3)') (cmthk(1,i),i=1,3)
            read(1,'(6e10.3)') (tencvx(1,i),tencvy(1,i),i=1,3)
            read(1,'(2e10.3)') sttkcm,sttktn
            read(1,'(6e10.3)') (stlrad(i),stlspc(i),i=1,3)

         endif

c.....Well dimensions and design specifications.

         if(idflag .gt. 1) then

            read(1,'(2e10.3)') wlhght,wlrad
            read(1,'(3e10.3)') (cmthk(2,i),i=1,3)
            read(1,'(6e10.3)') (tencvx(2,i),tencvy(2,i),i=1,3)
            read(1,'(3e10.3)') wlstr,stlpsn,stldns

         endif

         read(1,'(4e10.3)') submod,flangl,sldns,slangl
         read(1,'(4e10.3)') cvrthk,cvrdns,wstdns,wstht

c.....Concrete and steel specifications.

         read(1,'(7e10.3)') ccdns,ccpor,conpsn,com28d,wcr,
     #        phbeg,wtcmnt
         read(1,'(4e10.3)') clcon,ccon,cfa,cfb
         read(1,'(3e10.3)') stlmod,stlyld,yngmod
         read(1,'(3e10.3)') cacon,cap,si

c.....Chemical exposure conditions, diffusion coefficients, groundwater
c.....properties, and solubilities.

         read(1,'(8e10.3)') cagw,cl,co2,co3,xmg2,o2,so4i,so4o
         read(1,'(6e10.3)') dfalk,dfcaoh,dfcl,dfco2,dfo2,dfso4
         read(1,'(3e10.3)') phgw,tds,temp
         read(1,'(3e10.3)') casol,crbsol,xmgsol

c.....Metal failure function data.

         if(idflag.eq.1 .or. idflag.eq.3) then

            read(1,'(4e10.3)') eftl,deft,xlftl,dlft

         endif

         if(idflag .gt. 1) then

            read(1,'(2e10.3)') wftl,dwft

         endif

c.....Hydraulic parameters.

         read(1,'(4e10.3)') sitara,slkr,slk,cck

c.....Input name of file containing water seepage data.

         read(1,'(a)') wat_inp
         open(unit = 4, file = wat_inp, status = 'old')

c.....Radionuclide-specific data.

         read(1,'(i5)') noncld

         if(noncld .gt. maxnuc)then

            write
     #         (*,'('' The value of the variable noncld is greater'')')
```

```
            write(*,'('' than the value specified for maxnuc on the'')')
            write
     #         (*,'('' parameter statements.  Increase the value of'')')
            write(*,'('' maxnuc.'')')
            stop

         endif

         do 100 i = 1,noncld

            read(1,'(a8,7e10.3)') nuclid(i),am(i),hlife(i),sol(i),
     #           xkd(i),qsw(i),dfwst(i),dfcon(i)

            if(qsw(i) .le. 0.)ndump=0

  100    continue

         if(ndump .ne. 1) then

            read(1,'(i10)') refyear
            read(1,'(2i10,10e10.3)')
     #          bgndump, enddump, (qsw(i),i=1,noncld)
            ndump = bgndump

            do i = 1,noncld

               qswlast(i) = qsw(i)

            enddo

         endif

c.....Calculate or initialize various parameters for silo or well. The
c.....characteristic dimensions used in diffusion leaching calculations
c.....are based, initially, on the assumption that the releases occur
c.....through the roof and floor of the silo or well.

         if(idflag .eq. 1) then

            acoef = (silrad-0.5*cmthk(1,2))/39.37
            bcoef = cmthk(1,2)/39.37

         else

            acoef = (wlhght/2.-0.5*cmthk(2,3))/39.37
            bcoef = cmthk(2,3)/39.37

         endif

         constr = com28d/cfb

         if(idflag .eq. 1) then

            wsthk = (silrad-0.5*cmthk(1,2))*2.54

         else

            wsthk = (wlrad-0.5*cmthk(2,2))*2.54

         endif

         osttkc = sttkcm
         osttkt = sttktn

         do 300 l=1,2

            do 200 i=1,3

               comcvx(l,i) = cmthk(l,i)-tencvx(l,i)
               comcvy(l,i) = cmthk(l,i)-tencvy(l,i)
               ommthk(l,i) = cmthk(l,i)
               otncvx(l,i) = tencvx(l,i)
               otncvy(l,i) = tencvy(l,i)
```

```
               ph(1,i) = phbeg
  200       continue
  300    continue
         do 400 i=1,3
            ostlrd(i) = stlrad(i)
  400    continue
c.....Calculate uniform load on roof.
         xload = cvrthk*cvrdns*3.61e-2+cmthk(1,1)*ccdns*3.61e-2
c.....Convert hlife from years to seconds.
         do 500 n=1,noncld
            hlife(n) = hlife(n)*3.15576e7
  500    continue
      endif
c.....Read inventories and years of disposal.
      if(ndump .ne. 1) then
         if(iyear .gt. 1 .and. enddump .lt. maxyr) then
            if(ndump + iyear - 1 .gt. enddump) then
c.....Save the current simulated inventory.
               do i = 1,noncld
                  qswsav(i) = qsw(i)
               enddo
c.....Read beginning and ending year of disposal and
c.....associated inventory.
               read(1,'(2i10,10e10.3)')
     #              bgndump, enddump, (qsw(i),i=1,noncld)
               do i = 1,noncld
c.....Store new inventory to be used for updating until
c.....another disposal occurs.
                  qswlast(i) = qsw(i)
c.....Update inventory for model simulation with new inventory.
                  qsw(i) = qsw(i) + qswsav(i)
               enddo
            else
c.....Update simulated inventory with most recently disposed
c.....inventory.
               do i = 1,noncld
                  qsw(i) = qsw(i) + qswlast(i)
               enddo
            endif
```

```
         endif

      endif

c.....Update water seepage values.

      if (iyear .gt. iyr2) then

         read(4,'(2i10,12f5.2)')iyr1, iyr2, (water(i),i=1,12)

c.....Calculate annual percolation rate through intact concrete.

         annprc = 0.

         do mo=1,12

            annprc = annprc+amin1(cck*8.64e4*dpm(mo),water(mo))

         enddo

      endif


      return
      end

      subroutine leach(iyear,nyears)
c-----------------------------------------------------------------------------
c
c     Calculates annual radionuclide releases due to advection and diffusion.
c
c     Calls: flothru
c-----------------------------------------------------------------------------

      parameter (maxnuc = 10)

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/failure/eftl,deft,wftl,dwft,xlftl,dlft
      common/flagg/iflags(maxnuc)
      common/flows/rflow,sflow
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/miscel/acoef,bcoef,dpm(12)
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qsw(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc),xleach(maxnuc)
      common/pleach/cumlch(maxnuc),sladv(maxnuc),sldif(maxnuc)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht

      dimension rel(maxnuc,12)
      dimension dkayyr(maxnuc),dkaymo(maxnuc,12),dkaymon(maxnuc,12)

      real*8 t1,t2,lmbda(maxnuc),lmbdad(maxnuc)
      character*8  nuclid

      save dkayyr,dkaymo,slched

c.....Define the local vectors  dkayyr, dkaymo on first call.
c.....dkaymo(n,mo) = decay constant for first mo months of a year.

      if(iyear.eq.1) then

         nyd5p1 = nyears/5 + 1
```

```
         dcon = -log(2.)*3.15576e7
         ccon = -80./dcon

         do 100 n=1,noncld

            arg=dcon/hlife(n)
            dkayyr(n)=exp(arg)
            arg=arg/365.25
            lmbdad(n)=(-arg)

            do 50 mo=1,12

               dkaymon(n,mo) = exp(arg*dpm(mo))
               sum = 0.

               if(mo .eq. 12) then

                  dkaymo(n,mo) = 1.

               else

                  do imo = mo+1,12

                     sum = sum+dpm(imo)

                  enddo

                  dkaymo(n,mo) = exp(arg*sum)

               endif

  50        continue

  100    continue

      endif

c.....Calculate failure fraction for corrugated steel liner in silo to
c.....determine the inventory available for leaching from silo
c.....or multiple containment well unit.

      if (idflag.eq.1 .or. idflag.eq.3) then

         if (dlft .gt. 0.) then

            aux = float(iyear)

            if (aux .gt. xlft1) then

               xlff = amin1(1.,(aux-xlft1)/dlft)

            else

               do 150 n=1,noncld
                  qsw(n)=dkayyr(n)*qsw(n)
  150          continue

               return

            endif

         else

            xlff = 1.

         endif

      else

         xlff = 1.

      endif
```

```
      do 200 n=1,noncld

         sladv(n)=0.
         sldif(n)=0.

  200 continue

c.....Determine which xperc() to use.

      idx=2

      if(idflag.eq.3) then

         if(isave1.eq.0 .or. isave2.eq.0) idx=1

      else

         if(isave1.eq.0 .and. isave2.eq.0) idx=1

      endif

      rflow = 0.
      sflow = 0.

      do n = 1,noncld

c.....Set very small inventory values to zero to prevent numerical
c.....problems in the leaching calculations.

         if(qsw(n) .lt. 1.e-25) qsw(n) =0.

      enddo

      do mo = 1,12

         do l = 1,noncld

            rel(l,mo) = 0.

         enddo

      enddo

c.....Begin monthly loop

      t1=1.

      do 500 mo=1,12

         t2=t1+dpm(mo)-1.
         ttlwat = water(mo)/100.*sitara
         xperc(1) = amin1(cck*8.64e4*dpm(mo),water(mo))
         xperc(2) = amin1(slk*8.64e4*dpm(mo),water(mo))

c.....Partition flow into a vertical and lateral component

         tmp = amin1(1.,slkr*8.64e4*dpm(mo)/water(mo))
         rflow = rflow+tmp*water(mo)
         sflow = sflow+(1.-tmp)*water(mo)

c.....Begin nuclide loop

         do 300 l=1,noncld

c.....Calculate monthly leach rates due to advection.
c.....Advective releases are based on percolation rates for intact and
c.....cracked/failed disposal units.  In the case of multiple containment
c.....wells, it is assumed that the higher percolation rate applies only
c.....when both the silo and well have failed.
c.....Calculate leach rate constant.

            lmbda(l)=xperc(idx)/(wsthk*(wstht+wstdns*xkd(l)))
            lmbda(l)=lmbda(l)/dpm(mo)
```

```
c.....Calculate monthly release (integrated) due to advection.

            xleach(l)=((lmbda(l)*qsw(l))/(lmbda(l)+lmbdad(l)))*
     #                (dexp(-t1*(lmbda(l)+lmbdad(l)))-
     #                dexp(-t2*(lmbda(l)+lmbdad(l))))

c.....Calculate monthly leach rates due to diffusion for entire year
c.....using the flothru computer code and initialize leach fractions
c.....for recharge and lateral flow components.

            if(mo .eq. 1) then

               if(qsw(l) .ne. 0.)
     #            call flothru(dfwst(l),dfcon(l),hlife(l),
     #            iyear,l,qsw(l),rel)
               rlch(l) = 0.
               xllch(l) = 0.

            endif

c.....Sum diffusive and advective releases and correct for portion of
c.....corrugated steel liner still present.

            xleach(l) = (xleach(l)+rel(l,mo))*xlff
            rel(l,mo) = rel(l,mo)*xlff

  300    continue

c.....Check calculated releases against solubility limits using the total
c.....amount of water passing through the disposal unit.

         call maxlch(xperc(idx)/100. * sitara)

         do 400 l=1,noncld

            qsw(l)=amax1(0.,dkaymon(l,mo)*qsw(l)-xleach(l))

c.....Partition release into lateral flow and recharge components assuming
c.....same contaminant concentration in each component.  Decay partitioned
c.....releases to end of current year.

            rlch(l) = rlch(l)+xleach(l)*tmp*dkaymo(l,mo)
            if(tmp .lt. 1.) xllch(l) = xllch(l)+xleach(l)*
     #         (1.-tmp)*dkaymo(l,mo)

c.....Sum leaching due to advection and diffusion.

            sladv(l)=sladv(l)+dkaymo(l,mo)*(xleach(l)-rel(l,mo))
            sldif(l)=sldif(l)+dkaymo(l,mo)*rel(l,mo)

  400    continue

         t1=t2+1.

  500 continue

c.....Determine total annual release for output to summary file.

      do 600 l=1,noncld

         xleach(l) = rlch(l)+xllch(l)

c.....Determine cumulative amount leached.

         if(iyear.eq.1)then

            cumlch(l)=xleach(l)

         else

            cumlch(l)=cumlch(l)+xleach(l)

         endif
```

```
  600 continue

c.....Reset negative diffusion values.

      do n=1,noncld

         if (sldif(n) .lt. 0) sldif(n) = - sldif(n)

      enddo

      return
      end

      subroutine maxlch(ttlwat)

c-----------------------------------------------------------------------------
c     Called by leach
c
c     Calculates solubility limitations on leach rate.
c
c     Calls: none
c-----------------------------------------------------------------------------

      parameter (maxnuc = 10)

      common/flagg/iflags(maxnuc)
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qsw(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc),xleach(maxnuc)

      dimension match(maxnuc)

      character*8 nuclid
      character*2 xn(maxnuc)

      data ifl/0/
      data iflags/maxnuc * 0/

c.....Find occurrences of multiple isotopes of the same element.

      if (ifl .eq. 0) then

         do 100 i=1,noncld

            match(i) = 0
            xn(i) = nuclid(i)

  100    continue

         do 300 i=1,noncld

            do 200 j=i,noncld

               if (match(j).eq.0 .and. xn(j).eq.xn(i)) match(j)=i

  200       continue

  300    continue

         ifl=1

      endif

c.....Calculate maximum leach fraction allowed by solubility.

      do 600 i=1,noncld

         if(sol(i).eq.0. .or. match(i).lt.i) goto 600
         emole=0.

         do 400 j=1,noncld
```

```
            if(match(j) .eq. i) emole=emole+qsw(j)/am(j)

  400    continue

         if (emole .eq. 0.) goto 600
         xlmax = 1000.*sol(i)*ttlwat/emole

         do 500 j=1,noncld

            if (match(j).eq.i)then

               if(xleach(j) .gt. (qsw(j)*xlmax))
     #            iflags(j) = iflags(j) + 1
               xleach(j) = amin1(qsw(j)*xlmax,xleach(j))

            endif

  500    continue

  600 continue

      return
      end

      subroutine output(iyear, nyears)

c------------------------------------------------------------------------------
c     Called by main
c
c     Prints results of concrete cracking analyses and leach calculations.
c
c     Calls: none
c------------------------------------------------------------------------------

      parameter (maxnuc = 10)

      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/dump/ndump,refyear
      common/failure/eft1,deft,wft1,dwft,xlft1,dlft
      common/files/iprint,fname,iprn1,ifrq1,iprn2,ifrq2,iprn3,ifrq3,
     #       iprn4,ifrq4,iprn5,ifrq5,filenam(7)
      common/flagg/iflags(maxnuc)
      common/flows/rflow,sflow
      common/hydraul/cck,phgw,sitara,slkr,slk,tds,temp,water(12),
     #       iyr1,iyr2
      common/nuclide/noncld,nuclid(maxnuc),am(maxnuc),
     #       dfcon(maxnuc),dfwst(maxnuc),
     #       hlife(maxnuc),xllch(maxnuc),qsw(maxnuc),
     #       rlch(maxnuc),sol(maxnuc),
     #       xkd(maxnuc),xleach(maxnuc)
      common/pleach/cumlch(maxnuc),sladv(maxnuc),sldif(maxnuc)
      common/runid/title
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr

      dimension name1(3),name2(3)

      integer refyear
      character*8 nuclid,title*80
      character*10 name1,name2
      character fname*12,filenam*20
```

```
      character*23 label1(maxnuc)/10*'Inventory remaining (g)'/,
     #             label2(maxnuc)*24/10*'Annual leach rate (g/yr)'/,
     #             label3(maxnuc)*25/10*'Cumulative leached (g/yr)'/
      character*21 label4(maxnuc)/10*'Adv leach rate (g/yr)'/,
     #             label5(maxnuc)/10*'Dif leach rate (g/yr)'/
      character*12 label6(maxnuc)/10*'Recharge (g)'/,
     #             label7(maxnuc)/10*'Lateral (g)'/
      character*1  dashs(76)/76*'-'/

      data name1,name2/'Silo roof','Silo wall','Silo floor','Well roof',
     #                 'Well wall','Well floor'/
      data zero/0./

      if(iyear .eq. 0) then

         if(iprint .eq. 0 .or. iprn3 .eq. 0)
     #      open(unit=7,file=filenam(2),status='new')

c.....Open file for input data summary and concrete analysis
c.....with name "fname".con.

         if (iprint .eq. 0) then

            write(7,1000) title
            write(7,1025)
            write(7,1125) nyears
            write(7,1060) ifrq3

            if(idflag .eq. 1) then

               write(7,1050)

            elseif(idflag .eq. 2) then

               write(7,1075)

            elseif(idflag .eq. 3) then

               write(7,1100)

            endif

            write(7,1150) sitara,tds,temp,phgw,slkr,slk,cck
            write(7,1175) cagw,cl,co3,xmg2,so4i,so4o,o2
            write(7,1200) casol,crbsol,xmgsol
            write(7,1225) cacon,cap,ccon,clcon,si
            write(7,1250) com28d*7.04e-2,conpsn,stlmod*7.04e-2,
     #            stlyld*7.04e-2,submod*7.04e-2,yngmod*1.02e-5,wcr,
     #            ccdns,ccpor,wtcmnt,phbeg
            if(idflag .gt. 1) write(7,1275) stldns,stlpsn,wlstr
            write(7,1300) dfalk,dfcaoh,dfcl,dfco2,dfo2,dfso4
            if(idflag.eq.1 .or. idflag.eq.3) write(7,1325)
     #         silrad/39.37,
     #         slhght/39.37,(cmthk(1,i)*2.54,i=1,3),(stlrad(i)*
     #         2.54,i=1,3),(stlspc(i)*2.54,i=1,3),sttkcm*2.54,
     #         sttktn*2.54,(tencvx(1,i)*2.54,tencvy(1,i)*2.54,
     #         i=1,3)
            if(idflag .gt. 1) write(7,1350) wlrad*2.54,wlhght*2.54,
     #         (cmthk(2,i)*2.54,i=1,3),(tencvx(2,i)*2.54,
     #         tencvy(2,i)*2.54,i=1,3,2)
            write(7,1375) xload*7.04e-2
            write(7,1400) cvrthk/39.37,cvrdns,flangl,slangl,sldns,
     #            wstdns,wstht
            write(7,1425)
            if(idflag.eq.1 .or. idflag.eq.3) write(7,1450) eftl,deft,
     #         xlftl,dlft
            if(idflag .gt. 1) write(7,1475) wftl,dwft
            write(7,1500)
            write(7,1525) (nuclid(i),hlife(i)/3.15576e7,sol(i),
     #            xkd(i),dfwst(i),dfcon(i),qsw(i),i=1,noncld)

         endif
```

```
         if(iprn1 .eq. 0) then

c.....Open file for recharge components with name "fname".rch.

            open(unit=2,file=filenam(4),recl=246,status='new')
            write(2,1000) title
            write(2,'(t1,''Water and total grams in '',
     #            ''recharge component''/)')
            write(2,'(t41,10(a8,12x))')
     #            (nuclid(n),n=1,noncld)
            write(2,'(t8,''Year'',
     #            t19,''Water infiltration (cm)'',
     #            t48,10(a,8x)/)')
     #            (label6(n),n=1,noncld)

            if(ndump .gt. refyear) then

               do iz = 1, ndump-refyear

                  write(2,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #                  (refyear+iz-1),zero,(zero,n=1,noncld)

               enddo

            endif

         endif

         if(iprn2 .eq. 0) then

c.....Open file for lateral flow with name "fname".lat.

            open(unit=3,file=filenam(5),recl=246,status='new')
            write(3,1000) title
            write(3,'(t1,''Water and total grams in '',
     #            ''lateral component''/)')
            write(3,'(t41,10(a8,12x))')
     #            (nuclid(n),n=1,noncld)
            write(3,'(t8,''Year'',
     #            t19,''Water infiltration (cm)'',
     #            t48,10(a,8x)/)')
     #            (label7(n),n=1,noncld)

            if(ndump .gt. refyear) then

               do iz = 1, ndump-refyear

                  write(3,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #                  (refyear+iz-1),zero,(zero,n=1,noncld)

               enddo

            endif

         endif

         if(iprn4 .eq. 0) then

c.....Open file for output summary information
c.....with name "fname".sum.

            open(unit=10,file=filenam(6),recl=829,status='new')
            write(10,1000) title
            write(10,1500)
            write(10,1525) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #            dfwst(i),dfcon(i),qsw(i),i=1,noncld)
            write(10,1550)
            write(10,'(t44,10(a8,70x))')
     #            (nuclid(n),n=1,noncld)
            write(10,'(t14,10(76(a),2x))')(dashs,n=1,noncld)
            write(10,'(t8,''Year'',
     #            t14,10(3(a,2x))/)')
     #            (label1(n),label2(n),label3(n),n=1,noncld)
```

```
         endif

         if(iprn5 .eq. 0) then

c.....Open file for annual advective loss, diffusive loss, and total loss
c.....with name "fname".lch.

            open(unit=11,file=filenam(3),recl=766,status='new')
            write(11,1000) title
            write(11,1500)
            write(11,1525) (nuclid(i),hlife(i)/3.15576e7,sol(i),xkd(i),
     #            dfwst(i),dfcon(i),qsw(i),i=1,noncld)
            write(11,1550)
            write(11,'(t41,10(a8,64x))')
     #            (nuclid(n),n=1,noncld)
            write(11,'(t14,10(70(a),2x))')
     #            ((dashs(id),id=1,70),n=1,noncld)
            write(11,'(t8,''Year'',
     #            t14,10(3(a,2x))/)')
     #            (label4(n),label5(n),label2(n),n=1,noncld)

         endif

         if(iprint .eq. 0) then

c.....Open file for water infiltration summary information
c.....with name "fname".h2o.

            open(unit=12,file=filenam(7),status='new')
            write(12,'('' Summary of Infiltration Data '',/)')
            write(12,1000) title
            write(12,'(''      Year1      Year2     Jan   Feb'',
     #           ''   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct'',
     #           ''   Nov   Dec'' /)')

         endif

         return

      endif

      if((iprn3 .eq. 0 .and. mod(iyear,ifrq3) .eq. 0) .or.
     #   (iyear .eq. 1 .and. iprn3 .eq. 0)) then

         if(iyear .eq. 1) then

            write(7,2000) ndump

         else

            if(ndump .gt. 1) then

               write(7,2000) ndump+iyear

            else

               write(7,2000) iyear

            endif

         endif

c.....Print concrete degradation.

         write(7,2005)

         if(idflag.eq.1 .or. idflag.eq.3) then

            if(idflag .eq. 3) write(7,2010)
            write(7,2015) (name1(i),cmthk(1,i)*2.54,i=1,3),(name1(i),
     #            amin1(ommthk(1,i),iyear*(slfi+slfo))*2.54,i=1,3),
     #            (name1(i),1.-attk(1,i),i=1,3)
```

```
         endif

         if(idflag .gt. 1) then

            if(idflag .eq. 3) write(7,2018)
            write(7,2020) (name2(i),cmthk(2,i)*2.54,i=1,3),(name2(i),
     #            aminl(ommthk(2,i),iyear*(slfi+slfo))*2.54,i=1,3,2),
     #            (name2(i),1.-attk(2,i),i=1,3,2)

         endif

         if(idflag.eq.1 .or. idflag.eq.3) then

            write(7,2030) (namel(i),max0(icl(i),ico2(i)),i=1,3,2),
     #            (namel(i),stlcor(i)*2.54,i=1,3,2),
     #            (namel(i),stlrad(i)*
     #            2.54,i=1,3,2),sttkcm*2.54,sttktn*2.54

         endif

c.....Print results of cracking analyses.

         write(7,2040)

         if(idflag.eq.1 .or. idflag.eq.3) then

            write(7,2050)

            do 100 i=1,3

               if (ispl(i) .eq. 1) then

                  write(7,2060) namel(i)

               elseif (crfrcd(i) .gt. .75*cmthk(1,i)) then

                  write(7,2070) namel(i)

               else

                  write(7,2080) namel(i)

               endif

  100       continue

         endif

         write(7,2090)

         if(idflag.eq.1 .or. idflag.eq.3) then

            do 200 i=1,3

               if (icrack(i).eq.1) then

                  write(7,2070) namel(i)

               else

                  write(7,2080) namel(i)

               endif

  200       continue

         endif

         if(idflag .gt. 1) then

            do 300 i=1,3

               if (idflag.eq.3 .and. i.eq.1) write(7,*)
```

```
           if (ifail(i).eq.1) then

              write(7,2070) name2(i)

           else

              write(7,2080) name2(i)

           endif

  300   continue

        endif

        if(idflag.eq.1 .or. idflag.eq.3) then

           if (isavel .gt. 0) write(7,2130)

           do 400 i=1,3

               if (aper(i) .gt. 0.) then

                  write(7,2140) name1(i),aper(i)
                  write(7,2150) frac(i)

               endif

  400      continue

        endif

c.....Print radionuclide release rates.

        write(7,2160)
        write(7,2170) (nuclid(i),xleach(i),i=1,noncld)

      endif

c.....Output summary values for inventory and leaching.

      if(iprn4 .eq. 0 .and. mod(iyear,ifrq4) .eq. 0)
     #   write(10,'(i10,t18,10(1pe12.4,14x,1pe12.4,14x,
     #         1pe12.4,14x))')
     #         ndump+iyear-1,(qsw(n),xleach(n),
     #         cumlch(n),n=1,noncld)

c.....Output values for leaching.

      if(iprn5 .eq. 0 .and. mod(iyear,ifrq5) .eq. 0)
     #   write(11,'(i10,t17,10(1pe12.4,11x,1pe12.4,13x,
     #         1pe12.4,12x))')
     #         ndump+iyear-1,(sladv(n),sldif(n),xleach(n),n=1,noncld)

c.....Output results of solubility check.

      if(iyear .ge. nyears)then

         do n = 1,noncld

            if(iflags(n) .ne. 0 )then

               if(iprn4 .eq. 0)write(10,702) nuclid(n)
               if(iprn5 .eq. 0)write(11,702) nuclid(n)
  702          format(///' The solubility constraints were exceeded ',
     #                'for ',a)

            else

               if(iprn4 .eq. 0)write(10,703) nuclid(n)
               if(iprn5 .eq. 0)write(11,703) nuclid(n)
  703          format(///' The solubility constraints were not ',
     #                'exceeded for ',a)
```

```
            endif

         enddo

      endif

c.....Write annual releases to lateral and recharge component files.

      if(iprn1 .eq. 0 .and. mod(iyear,ifrq1) .eq. 0)
     #   write(2,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #         (ndump+iyear-1),rflow,(r1ch(n),n=1,noncld)

      if(iprn2 .eq. 0 .and. mod(iyear,ifrq2) .eq. 0)
     #   write(3,'(i10,10x,1pe16.8,8x,10(1pe16.8,4x))')
     #         (ndump+iyear-1),sflow,(x11ch(n),n=1,noncld)

      if (iyear .eq. iyr2 .or. iyear .eq. nyears) then

         if(iprint .eq. 0)
     #      write(12,'(1h ,i10,1x,i10,3x,12f6.2)') iyr1, iyr2,
     #            (water(i),i=1,12)

      endif

      return

c-------------------------------------------------------------------------------
c     format statements
c-------------------------------------------------------------------------------

1000  format(/80('-')/a80/80('-')/)
1025  format(/t1,'Input Data Summary:'/t1,19('-'))
1050  format(/t1,' Disposal technology: silo')
1060  format(' Output edit frequency',t50,i10,t61,'years')
1075  format(/t1,' Disposal technology: well')
1100  format(/t1,' Disposal technology: multiple containment wells')
1135  format(t4,1pe10.2,5(2x,e10.2))
1125  format(/' Simulation length',t50,i10,t61,'years')
1150  format(/t6,'Disposal unit area',t50,1pe10.2,' m**2'/t6,'Total ',
     #       'dissolved solids',t50,e10.2,' ppm'/t6,'Groundwater ',
     #       'temperature',t50,e10.2,' deg c'/t6,'Groundwater pH',t50,
     #       e10.2//t6,'Saturated hydraulic conductivity:'/
     #       t8,'Recharge',t50,e10.2,' cm/s'/
     #       t8,'Soil ',
     #       'backfill',t50,e10.2,' cm/s'/t8,'Concrete',t50,e10.2,
     #       ' cm/s')
1175  format(/' Groundwater constituent concentrations:'/t6,'Ca++',t50,
     #       1pe10.2,' mole/L'/t6,'Cl-',t50,e10.2,' mole/L'/t6,'CO3--',
     #       t50,e10.2,' mole/L'/t6,'Mg++',t50,e10.2,' mole/L'/t6,
     #       'SO4-- (inside silo or well)',t50,e10.2,' mole/L'/t6,
     #       'SO4-- (outside silo or well)',t50,e10.2,' mole/L'/t6,
     #       'O2',t50,e10.2,' mole/L')
1200  format(/' Constituent solubilities:'/t6,'Ca(OH)2',t50,1pe10.2,
     #       ' mole/L'/t6,'CO3--',t50,e10.2,' mole/L'/t6,'Mg++',t50,
     #       e10.2,' mole/L')
1225  format(/' Concrete constituent concentrations:'/t6,'Calcium ',
     #       'Concentration in C-S-H system',t50,1pe10.2,' mole/L'/t6,
     #       'Calcium concentration in pore fluid',t50,e10.2,' mole/L'/
     #       t6,'CaO content in cement',t50,e10.2,' mole/L'/t6,'Free ',
     #       'Cl-',t50,e10.2,' mole/L'/t6,'Silica concentration in ',
     #       'C-S-H system',t50,e10.2,' mole/L')
1250  format(/,' Concrete design specifications:'/t6,'Compressive ',
     #       'strength at 28 days',t50,1pe10.2,' kg/cm**2'/t6,
     #       'Poisson''s ratio of concrete',t50,e10.2/t6,'Modulus of ',
     #       'elasticity of ',
     #       'steel',t50,e10.2,' kg/cm**2'/t6,'Yield strength of steel',
     #       t50,e10.2,' kg/cm**2'/t6,'Modulus of subgrade reaction',
     #       t50,e10.2,' kg/cm**2'/t6,'Young''s modulus of elasticity',
     #       t50,e10.2,' kg/cm**2'/t6,'Concrete water/cement ratio',t50,
     #       e10.2/t6,'Concrete density',t50,e10.2,' g/cm**3'/t6,
     #       'Concrete porosity',t50,e10.2/t6,'Cement content',t50,
     #       e10.2,' kg/m**3'/t6,'Initial pH',t50,e10.2)
1275  format(/,' Well steel properties:'/t6,'Steel density',t50,1pe10.2,
```

```
     #       ' g/cm**3'/t6,'Steel poisson ratio',t50,e10.2/t6,'Yield ',
     #       'strength of steel wall',t50,e10.2,' lb/in.**2')
1300  format(/,' Diffusion coefficients in concrete:',/,t6,'NaOH, KOH',
     #       t50,1pe10.2,' m**2/s',/,t6,'Ca(OH)2',t50,e10.2,' m**2/s',
     #      /,t6,'Cl-',t50,e10.2,' m**2/s',/,t6,'CO2',t50,e10.2,
     #       ' m**2/s',/,t6,'O2',t50,e10.2,' m**2/s',/,t6,'SO4--',t50,
     #       e10.2,' m**2/s')
1325  format(/' Silo design specifications:'//t6,'Silo dimensions:'/t8,
     #       'Radius',t50,1pe10.2,' m'/t8,'Height',t50,e10.2,
     #       ' m'//t6'Concrete member thickness:'/t8,'Roof',t50,e10.2,
     #       ' cm'/t8,'Walls',t50,e10.2,' cm'/t8,'Floor',t50,e10.2,' cm'
     #      //t6,'Steel reinforcement radius:'/t8,'Roof',t50,e10.2,
     #       ' cm'/t8,'Walls',t50,e10.2,' cm'/t8,'Floor',t50,e10.2,' cm'
     #      //t6,'Spacing of steel reinforcement:'/t8,'Roof',t50,e10.2,
     #       ' cm'/t8,'Walls',t50,e10.2,' cm'/t8,'Floor',t50,e10.2,
     #       ' cm'//t6,'Corrugated steel thickness:'/t8,'Compression ',
     #       'face',t50,e10.2,' cm'/t8,'Tension face',t50,e10.2,' cm'//
     #      t6,'Concrete cover thickness on tension face:'/t8,'Roof:'/
     #      t10,'X-direction',t50,e10.2,' cm'/t10,'Y-direction',t50,
     #      e10.2,' cm'/t8,'Walls:'/t10,'Horizontal direction',t50,
     #      e10.2,' cm'/t10,'Vertical direction',t50,e10.2,' cm'/t8,
     #      'Floor:'/t10,'X-direction',t50,e10.2,' cm'/t10,
     #      'Y-direction',t50,e10.2,' cm')
1350  format(/' Well design specifications:'//t6,'Well dimensions:'/t8,
     #       'Radius',t50,1pe10.2,' cm'/t8,'Height',t50,e10.2,
     #       ' cm'//t6,'Structural member thickness:'/t8,'Roof',t50,
     #       e10.2,' cm'/t8,'Wall',t50,e10.2,' cm'/t8,'Floor',t50,e10.2,
     #       ' cm'//t6,'Concrete cover thickness on tension face:'/t8,
     #       'Roof:'/t10,'X-direction',t50,e10.2,' cm'/t10,'Y-',
     #       'direction',t50,e10.2,' cm'/t8,'Floor:'/t10,'X-direction',
     #       t50,e10.2,' cm'/t10,'Y-direction',t50,e10.2,' cm')
1375  format(/t6,'Static load',t50,1pe10.2,' kg/cm**2')
1400  format(/' Soil and waste properties:'/t6,'Earthen cover ',
     #       'thickness',t50,1pe10.2,' m'/t6,'Earthen cover density',
     #       t50,e10.2,' g/cm**3'/t6,'Friction angle of waste backfill',
     #       t50,e10.2,' deg'/t6,'Friction angle of soil backfill',t50,
     #       e10.2,' deg'/
     #       t6,'Density of soil backfill',t50,e10.2,
     #       ' g/cm**3'/t6,'Waste density',t50,e10.2,' g/cm**3'/t6,
     #       'Relative saturation of waste',t50,e10.2)
1425  format(/' Concrete and steel failure rates:')
1450  format(t6,'Epoxy coating:'/t8,'Start of failure',t50,1pe10.2,
     #       ' years'/t8,'Time to complete failure',t50,e10.2,' years'/
     #       t6,'Steel liner:'/t8,'Start of failure',t50,e10.2,' years'/
     #       t8,'Time to complete failure',t50,e10.2,' years')
1475  format(t6,'Well wall:'/t8,'Start of failure',t50,1pe10.2,' years'/
     #       t8,'Time to complete failure',t50,e10.2,' years')
1500  format(/,' Nuclide-specific parameters:'//t2,'Nuclide',t13,
     #       'Half-life',t23,'Solubility',t36,'Waste',t45,'Diffusion ',
     #       'coefficient',t69,'Initial'/t38,'kd',t47,'Waste',t57,
     #       'Concrete',t68,'inventory'/t15,'(yr)',t24,'(mole/l)',t36,
     #       '(ml/g)',t46,'(m**2/s)',t57,'(m**2/s)',t71,'(g)'/t2,
     #       '---------',t12,'----------',t23,'----------',t34,
     #       '----------',t45,'-----------',t56,'-----------',t68,
     #       '-----------')
1525  format(t3,a8,t11,1pe10.2,t22,e10.2,t33,e10.2,t44,e10.2,t55,e10.2,
     #       t67,e10.2)
1550  format(//t1,'Output summary:'/t1,15('-'))
2000  format(/' ------------------------------'/' Annual Summary for ',
     #       'Year ',i10/' ------------------------------')
2005  format(/' Concrete Degradation Summary'/)
2010  format(/' Disposal silo:'/' --------------')
2015  format(/' Member thickness:'/3(t6,a10,t50,1pe10.2,' cm'/)/
     #       ' Concrete loss due to sulfate attack:'/3(t6,a10,t50,e10.2,
     #       ' cm'/)/' Fractional loss of yield strength'/' due to ',
     #       'Ca(OH)2 leaching:'/3(t6,a10,t50,e10.2/))
2018  format(/' Disposal well:'/' --------------')
2020  format(/' Member thickness:'/3(t6,a10,t50,1pe10.2,' cm'/)/
     #       ' Concrete loss due to sulfate attack:'/2(t6,a10,t50,e10.2,
     #       ' cm'/)/' Fractional loss of yield strength'/' due to ',
     #       'Ca(OH)2 leaching:'/2(t6,a10,t50,e10.2/)/)
2030  format(' Corrosion results:'/t6,'Time to onset of corrosion:'/
     #       2(t8,a10,t55,i5,' years'/)/t6,'Corrosion product layer ',
```

```
     #       'thickness:'/2(t8,a10,t50,1pe10.2,' cm'/)/t6,'Remaining ',
     #       'steel reinforcement:'/2(t8,a10,t50,e10.2,' cm'/)/t6,
     #       'Remaining corrugated steel:'/t8,'Compression face',t50,
     #       e10.2,' cm'/t8,'Tension face',t50,e10.2,' cm')
2040  format(/' Concrete Cracking Analysis')
2050  format(/' Cracking due to corrosion of steel:'/)
2060  format(t4,a10,t52,'spalled out')
2070  format(t4,a10,t52,'cracked')
2080  format(t4,a10,t52,'none')
2090  format(/' Cracking due to loading and shear:'/)
2130  format(//' Concrete crack characteristics:'/)
2140  format(/t3,a10/t6,'Average crack width (cm)',t50,1pe10.2)
2150  format(t6,'Fractional volume of cracks',t50,1pe10.2)
2160  format(///,3x,'Kd/diffusion controlled leach rates (g/yr)',/,
     #       1x,46('-'))
2170  format(3(1x,a8,1pe10.2,8x))

      end

      subroutine sar2

c-----------------------------------------------------------------------
c     Called by source2
c
c     Performs structural analysis of silos and wells.
c
c     Calls: none
c
c-----------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),ic1(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr
      common/wlforc/wlcmfr(11),wlxrc(11),wlwxrc(11),wwcmfr(11)

      real*8 alpha,beta

      data pi/3.141592653589793/

c.....Calculate modulus of elasticity of concrete for use in structural
c.....analysis of floor.

      time = 365.
      comstr = amin1(time/(cfa+cfb*time)*com28d,constr)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)

c.....Begin roof structural analysis.

c.....Calculate maximum shear and reaction for roof.

      if(idflag.eq.1 .or. idflag.eq.3) then

        rfshr = xload*(silrad-cmthk(1,2)/2.-cmthk(1,1)-tencvx(1,1))/2.
        rfrxn = xload*silrad/2.

      endif

      if(idflag .gt. 1) then
```

```
        rfwshr = xload*(wlrad-cmthk(2,2)/2.-cmthk(2,1)-tencvx(2,1))/2.
        rfwrxn = xload*wlrad/2.

      endif

c.....Calculate x- and y-direction moments, and shear and reaction for silo
c.....roof. Moment curves are discretized into eleven segments, over any
c.....one of which the moment is constant.

      if(idflag.eq.1 .or. idflag.eq.3) then

         do 200 k=0,5

            do 100 l=0,5

               if (l .gt. 0.) then

                  theta = atan(k*0.2*silrad/(l*0.2*silrad))

               elseif (l.eq.0 .and. k.gt.0) then

                  theta = pi/2.

               else

                  theta = 0.

               endif

               r = sqrt((0.2*k*silrad)**2+(0.2*l*silrad)**2)

               if (r .le. silrad) then

                  tmp1 = r/silrad
                  tmp2 = xload*silrad**2/16.*(3.+conpsn)*(1.-tmp1**2)
                  tmp3 = xload*silrad**2/16.*(3.+conpsn-(1.+3.*
     #                   conpsn)*tmp1**2)
                  rfxmnt(k+6-k*2,l+6) = abs(tmp2*(cos(theta))**2+
     #                                  tmp3*(sin(theta))**2)
                  rfymnt(k+6-k*2,l+6) = abs(tmp2*(sin(theta))**2+
     #                                  tmp3*(cos(theta))**2)
                  rfxmnt(k+6-k*2,6-l) = rfxmnt(k+6-k*2,l+6)
                  rfymnt(k+6-k*2,6-l) = rfymnt(k+6-k*2,l+6)
                  rfxmnt(k+6,l+6) = rfxmnt(k+6-k*2,l+6)
                  rfymnt(k+6,l+6) = rfymnt(k+6-k*2,l+6)
                  rfxmnt(k+6,6-l) = rfxmnt(k+6-k*2,l+6)
                  rfymnt(k+6,6-l) = rfymnt(k+6-k*2,l+6)

               endif

 100        continue

 200     continue

      endif

c.....Calculate x- and y-direction moments, and shear and reaction for
c.....well roof. Moment curves are discretized into eleven segments, over
c.....any one of which the moment is constant.

      if(idflag .gt. 1) then

         do 400 k=0,5

            do 300 l=0,5

               if (l .gt. 0.) then

                  theta = atan(k*0.2*wlrad/(l*0.2*wlrad))

               elseif (l.eq.0 .and. k.gt.0) then

                  theta = pi/2.
```

```
               else

                  theta = 0.

               endif

               r = sqrt((0.2*k*wlrad)**2+(0.2*l*wlrad)**2)

               if (r .le. wlrad) then

                  tmp1 = r/wlrad
                  tmp2 = xload*wlrad**2/16.*(3.+conpsn)*(1.-tmp1**2)
                  tmp3 = xload*wlrad**2/16.*(3.+conpsn-(1.+3.*
     #                   conpsn)*tmp1**2)
                  rwxmnt(k+6-k*2,l+6) = abs(tmp2*(cos(theta))**2+
     #                                  tmp3*(sin(theta))**2)
                  rwymnt(k+6-k*2,l+6) = abs(tmp2*(sin(theta))**2+
     #                                  tmp3*(cos(theta))**2)
                  rwxmnt(k+6-k*2,6-l) = rwxmnt(k+6-k*2,l+6)
                  rwymnt(k+6-k*2,6-l) = rwymnt(k+6-k*2,l+6)
                  rwxmnt(k+6,l+6) = rwxmnt(k+6-k*2,l+6)
                  rwymnt(k+6,l+6) = rwymnt(k+6-k*2,l+6)
                  rwxmnt(k+6,6-l) = rwxmnt(k+6-k*2,l+6)
                  rwymnt(k+6,6-l) = rwymnt(k+6-k*2,l+6)

               endif

  300       continue

  400    continue

      endif

c.....Begin wall structural analysis.

c.....Calculate ring compression, moments, and shears due to uniform
c.....load for silo wall.

      if(idflag.eq.1 .or. idflag.eq.3) then

         flarg = 1.-sin(flangl*pi/180.)
         slarg = 1.-sin(slangl*pi/180.)
         d = conmod*cmthk(1,3)**3/(12.*(1.-conpsn**2))
         beta = (3*(1.-conpsn**2)/(silrad**2*cmthk(1,2)**2))**0.25
         alpha = beta*slhght/2.
         unfld = sldns*3.61e-2*(cvrthk+0.5*cmthk(1,1))*slarg+0.5*slhght*
     #           (sldns*3.61e-2*slarg-wstdns*3.61e-2*flarg)
         hydrld = 0.5*slhght*(sldns*3.61e-2*slarg-wstdns*3.61e-2*flarg)

c.....Calculate compressive forces for vertical direction.

         do 500 m=0,10

            wlcmfr(m+1) = amax1(0.,rfrxn+cmthk(1,2)*m*slhght/10.*
     #                    ccdns*3.61e-2)

  500    continue

c.....Calculate ring compression for horizontal direction.

         do 600 k=5,-5,-1


        temp1a = (2.*sin(alpha)*sinh
     #               (alpha))/(cos(2.*alpha)+dcosh(2.*alpha))*sin
     #               (beta*k*slhght/10.)*dsinh(beta*k*slhght/10.)

        temp1b = (2.*cos(alpha)*dcosh(alpha))/(cos(2.*alpha)+
     #               dcosh(2.*alpha))*cos(beta*k*slhght/10.)*cosh
     #               (beta*k*slhght/10.)

        temp1 = 1.-temp1a-temp1b
            wlxrc(6+k-k*2) = unfld*silrad*temp1
```

```
c.....Calculate moments and shears for vertical direction.


        templa = (sin(alpha)*dsinh(alpha)/
     #              (cos(2.*alpha)+dcosh(2.*alpha)))*cos(beta*k*
     #              slhght/10.)*dcosh(beta*k*slhght/10.)

        templb = (cos(alpha)*dcosh(alpha)/(cos(2.*alpha)+
     #              dcosh(2.*alpha)))*sin(beta*k*slhght/10.)*
     #              dsinh(beta*k*slhght/10.)
            templ = templa-templb
            wlymnt(6+k-k*2) = unfld*(slhght**2)*templ/(4.*alpha**2)
            templa = sin(alpha)*dsinh(alpha)/
     #              (cos(2.*alpha)+dcosh(2.*alpha))
            templb = (cos(beta*k*slhght/10.)*dsinh(beta*k*slhght/
     #              10.)-sin(beta*k*slhght/10.)*dcosh(beta*k*
     #              slhght/10.))

        templc = (cos(beta*k*slhght/
     #              10.)*dsinh(beta*k*slhght/10.)+sin(beta*k*
     #              slhght/10.)*dcosh(beta*k*slhght/10.))
            wlyshr(6+k-k*2) = unfld*slhght/(2.*alpha)*( templa *
     #                       templb - cos(alpha)*dcosh(alpha)/(cos(2.*
     #                       alpha)+dcosh(2.*alpha))* templc )

  600    continue

c.....Calculate ring compression, moments, and shears due to hydrostatic
c.....load for silo wall. Start with ring compression for horizontal
c.....direction. combine results for uniform and hydrostatic loads.

         do 700 k=5,-5,-1


        templa = sin(alpha)*
     #              dcosh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha))*
     #              sin(beta*k*
     #              slhght/10.)*dcosh(beta*k*slhght/10.)

        templb = cos(alpha)*
     #              dsinh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha))*
     #              cos(beta*k*
     #              slhght/10.)*dsinh(beta*k*slhght/10.)
            tmpl = -2.*hydrld*silrad*(k*slhght/10./slhght- templa -
     #

               templb )
            wlxrc(6+k-k*2) = abs(wlxrc(6+k-k*2)+tmpl)

c.....Calculate moments and shears for vertical direction.


        temp2a = (sin(alpha)*
     #              dcosh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha)))*
     #              cos(beta*k*slhght/10.)*dsinh(beta*k*slhght/10.)
            temp2b = (cos(alpha)*
     #              dsinh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha)))*
     #              sin(beta*k*slhght/10.)*dcosh(beta*k*slhght/10.)
            tmp2 = hydrld*slhght**2/(4.*alpha**2)*( temp2a - temp2b)
            wlymnt(6+k-k*2) = abs(wlymnt(6+k-k*2)+tmp2)

        temp3a = sin(alpha)*dcosh(alpha)/
     #             (dcosh(2.*alpha)-cos(2.*alpha))

        temp3b = (cos(beta*k*slhght/10.)*
     #              dcosh(beta*k*slhght/10.)-sin(beta*k*slhght/10.)*
     #              sinh(beta*k*slhght/10.))

        temp3c = (cos(beta*k*slhght/10.)*cosh
     #              (beta*k*slhght/10.)+sin(beta*k*slhght/10.)*
     #              dsinh(beta*k*
     #              slhght/10.))
            tmp3 = hydrld*slhght/(2.*alpha)*( temp3a * temp3b -
     #            cos(alpha)*dsinh(alpha)/(dcosh(2.*
```

```
     #            alpha)-cos(2.*alpha))* temp3c )
             wlyshr(6+k-k*2) = abs(wlyshr(6+k-k*2)+tmp3)

  700    continue

      endif

c.....Calculate ring compression, moments, and shears due to
c.....uniform load for well wall.

      if(idflag .gt. 1) then

         flarg = 1.-sin(flangl*pi/180.)
         slarg = 1.-sin(slangl*pi/180.)
         d = stlmod*cmthk(2,3)**3/(12.*(1.-stlpsn**2))
         beta = (3*(1.-stlpsn**2)/(wlrad**2*cmthk(2,2)**2))**0.25
         alpha = beta*wlhght/2.
         unfld = sldns*3.61e-2*(cvrthk+0.5*cmthk(2,1))*slarg+0.5*wlhght*
     #           (sldns*3.61e-2*slarg-wstdns*3.61e-2*flarg)
         hydrld = 0.5*wlhght*(sldns*3.61e-2*slarg-wstdns*3.61e-2*flarg)

c.....Calculate compressive forces for vertical direction.

         do 800 m=0,10

            wwcmfr(m+1) = rfwrxn+cmthk(2,2)*m*wlhght/10.*stldns*3.61e-2

  800    continue

c.....Calculate ring compression for horizontal direction.

         do 900 k=5,-5,-1


        templa = (2.*sin(alpha)*sinh
     #              (alpha))/(cos(2.*alpha)+dcosh(2.*alpha))*sin
     #              (beta*k*wlhght/10.)*dsinh(beta*k*wlhght/10.)

        templb = (2.*cos(alpha)*dcosh(alpha))/(cos(2.*alpha)+
     #               dcosh(2.*alpha))*cos(beta*k*wlhght/10.)*cosh
     #               (beta*k*wlhght/10.)

        templ = 1.-templa-templb
            wlwxrc(6+k-k*2) = unfld*wlrad*templ

c.....Calculate moments and shears for vertical direction.


        templa = (sin(alpha)*dsinh(alpha)/
     #              (cos(2.*alpha)+dcosh(2.*alpha)))*cos(beta*k*
     #              wlhght/10.)*dcosh(beta*k*wlhght/10.)
            templb = (cos(alpha)*dcosh(alpha)/(cos(2.*alpha)+
     #              dcosh(2.*alpha)))*sin(beta*k*wlhght/10.)*
     #              dsinh(beta*k*wlhght/10.)
            templ = templa-templb
            wwymnt(6+k-k*2) = unfld*(wlhght**2)*templ/(4.*alpha**2)
            templa = sin(alpha)*dsinh(alpha)/
     #              (cos(2.*alpha)+dcosh(2.*alpha))
            templb = (cos(beta*k*wlhght/10.)*dsinh(beta*k*wlhght/
     #              10.)-sin(beta*k*wlhght/10.)*dcosh(beta*k*
     #              wlhght/10.))
            templc = (cos(beta*k*wlhght/
     #              10.)*dsinh(beta*k*wlhght/10.)+sin(beta*k*
     #              wlhght/10.)*dcosh(beta*k*wlhght/10.))
            wwyshr(6+k-k*2) = unfld*wlhght/(2.*alpha)*( templa *
     #                       templb - cos(alpha)*dcosh(alpha)/(cos(2.*
     #                       alpha)+dcosh(2.*alpha))* templc )

  900    continue

c.....Calculate ring compression, moments, and shears due to hydrostatic
c.....load. start with ring compression for horizontal direction. combine
c.....results for uniform and hydrostatic loads.
```

```
         do 1000 k=5,-5,-1

            templa = sin(alpha)*
     #               dcosh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha))*
     #               sin(beta*k*
     #               wlhght/10.)*dcosh(beta*k*wlhght/10.)
            templb = cos(alpha)*
     #               dsinh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha))*
     #               cos(beta*k*
     #               wlhght/10.)*dsinh(beta*k*wlhght/10.)
            tmpl = -2.*hydrld*wlrad*(k*wlhght/10./wlhght- templa -
     #             templb )
            wlwxrc(6+k-k*2) = abs(wlwxrc(6+k-k*2)+tmpl)

c.....Calculate moments and shears for vertical direction.

            temp2a = (sin(alpha)*
     #               dcosh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha)))*
     #               cos(beta*k*wlhght/10.)*dsinh(beta*k*wlhght/10.)

        temp2b = (cos(alpha)*
     #               dsinh(alpha)/(dcosh(2.*alpha)-cos(2.*alpha)))*
     #               sin(beta*k*wlhght/10.)*dcosh(beta*k*wlhght/10.)
            tmp2 = hydrld*wlhght**2/(4.*alpha**2)*( temp2a - temp2b)
            wwymnt(6+k-k*2) = abs(wwymnt(6+k-k*2)+tmp2)
            temp3a = sin(alpha)*dcosh(alpha)/
     #               (dcosh(2.*alpha)-cos(2.*alpha))
            temp3b = (cos(beta*k*wlhght/10.)*
     #               dcosh(beta*k*wlhght/10.)-sin(beta*k*wlhght/10.)*
     #               sinh(beta*k*wlhght/10.))
            temp3c = (cos(beta*k*wlhght/10.)*cosh
     #               (beta*k*wlhght/10.)+sin(beta*k*wlhght/10.)*
     #               dsinh(beta*k*
     #               wlhght/10.))
            tmp3 = hydrld*wlhght/(2.*alpha)*( temp3a * temp3b -
     #             cos(alpha)*dsinh(alpha)/(dcosh(2.*
     #             alpha)-cos(2.*alpha))* temp3c )
            wwyshr(6+k-k*2) = abs(wwyshr(6+k-k*2)+tmp3)

 1000    continue

      endif

c.....Begin floor structural analysis.

c.....Calculate maximum shear for silo floor.

      if(idflag.eq.1 .or. idflag.eq.3) then

         cncfrc = rfrxn+cmthk(1,2)*slhght*(ccdns-wstdns)*3.61e-2
         d = conmod*cmthk(1,3)**3/(12.*(1.-conpsn**2))
         xl = (d/submod)**.25
         x = silrad/(xl+xl)

         call series(x,10,z1,z2,zlp,z2p)

         phi = z1*z2p-zlp*z2+xl/silrad*(1.-conpsn)*(zlp**2+z2p**2)
         c1 = -cncfrc/(submod*xl*phi)*(z1+xl/silrad*(1.-conpsn)*z2p)
         c2 = -cncfrc/(submod*xl*phi)*(z2-xl/silrad*(1.-conpsn)*zlp)
         x = (silrad-cmthk(1,2)/2.-(cmthk(1,3)-tencvx(1,3)))/(xl+xl)

         call series(x,15,z1,z2,zlp,z2p)

         flshr = abs(-d/xl**3*(c1*z2p-c2*zlp))

c.....Calculate x- and y-direction moments, shear and reaction for silo
c.....floor. Moment curves are discretized into eleven segments, over any
c.....one of which the moment is constant.

         do 1200 k=0,5

            do 1100 l=0,5
```

```
             if (l .gt. 0) then

                theta = atan(k*0.2*silrad/(0.2*l*silrad))

             elseif (l.eq.0 .and. k.gt.0) then

                theta = pi/2.

             else

                theta = 0.

             endif

             r = sqrt((0.2*k*silrad)**2+(0.2*l*silrad)**2)

             if (r .le. silrad) then

                x = r/(xl+xl)

                call series(x,15,z1,z2,z1p,z2p)

                if (r .gt. 0.) then

                   tmp2 = -d/xl**2*(c1*z2-c2*z1-xl/r*(1.-conpsn)*
     #                    (c1*z1p+c2*z2p))
                   tmp3 = -d/xl**2*(conpsn*(c1*z2-c2*z1)+xl/r*
     #                    (1.-conpsn)*(c1*z1p+c2*z2p))
                else

                   tmp2 = d/(2.*xl**2)*c2*(1.+conpsn)
                   tmp3 = d/(2.*xl**2)*c2*(1.+conpsn)

                endif

                flxmnt(k+6-k*2,l+6) = abs(tmp2*(cos(theta))**2+
     #                                tmp3*(sin(theta))**2)
                flymnt(k+6-k*2,l+6) = abs(tmp2*(sin(theta))**2+
     #                                tmp3*(cos(theta))**2)
                flxmnt(k+6-k*2,6-l) = flxmnt(k+6-k*2,l+6)
                flymnt(k+6-k*2,6-l) = flymnt(k+6-k*2,l+6)
                flxmnt(k+6,l+6) = flxmnt(k+6-k*2,l+6)
                flymnt(k+6,l+6) = flymnt(k+6-k*2,l+6)
                flxmnt(k+6,6-l) = flxmnt(k+6-k*2,l+6)
                flymnt(k+6,6-l) = flymnt(k+6-k*2,l+6)

             endif

 1100      continue

 1200    continue

      endif

c.....Calculate maximum shear for well floor.

      if(idflag .gt. 1) then

         cncfrc = rfwrxn+cmthk(2,2)*wlhght*(stldns-wstdns)*3.61e-2
         d = conmod*cmthk(2,3)**3/(12.*(1.-conpsn**2))
         xl = (d/submod)**.25
         x = wlrad/(xl+xl)

         call series(x,10,z1,z2,z1p,z2p)

         phi = z1*z2p-z1p*z2+xl/wlrad*(1.-conpsn)*(z1p**2+z2p**2)
         c1 = -cncfrc/(submod*xl*phi)*(z1+xl/wlrad*(1.-conpsn)*z2p)
         c2 = -cncfrc/(submod*xl*phi)*(z2-xl/wlrad*(1.-conpsn)*z1p)
         x = (wlrad-cmthk(2,2)/2.-(cmthk(2,3)-tencvx(2,3)))/(xl+xl)

         call series(x,15,z1,z2,z1p,z2p)

         flwshr = abs(-d/xl**3*(c1*z2p-c2*z1p))
```

```
c.....Calculate x- and y-direction moments, shear and reaction for well
c.....floor. Moment curves are discretized into eleven segments, over any
c.....one of which the moment is constant.

         do 1400 k=0,5

            do 1300 l=0,5

               if (l .gt. 0) then

                  theta = atan(k*0.2*wlrad/(0.2*l*wlrad))

               elseif (l.eq.0 .and. k.gt.0) then

                  theta = pi/2.

               else

                  theta = 0.

               endif

               r = sqrt((0.2*k*wlrad)**2+(0.2*l*wlrad)**2)

               if (r .le. wlrad) then

                  x = r/(xl+xl)

                  call series(x,15,z1,z2,z1p,z2p)

                  if (r .gt. 0.) then

                     tmp2 = -d/xl**2*(c1*z2-c2*z1-xl/r*(1.-conpsn)*
     #                      (c1*z1p+c2*z2p))
                     tmp3 = -d/xl**2*(conpsn*(c1*z2-c2*z1)+xl/r*
     #                      (1.-conpsn)*(c1*z1p+c2*z2p))

                  else

                     tmp2 = d/(2.*xl**2)*c2*(1.+conpsn)
                     tmp3 = d/(2.*xl**2)*c2*(1.+conpsn)

                  endif

                  fwxmnt(k+6-k*2,l+6) = abs(tmp2*(cos(theta))**2+
     #                                  tmp3*(sin(theta))**2)
                  fwymnt(k+6-k*2,l+6) = abs(tmp2*(sin(theta))**2+
     #                                  tmp3*(cos(theta))**2)
                  fwxmnt(k+6-k*2,6-l) = fwxmnt(k+6-k*2,l+6)
                  fwymnt(k+6-k*2,6-l) = fwymnt(k+6-k*2,l+6)
                  fwxmnt(k+6,l+6) = fwxmnt(k+6-k*2,l+6)
                  fwymnt(k+6,l+6) = fwymnt(k+6-k*2,l+6)
                  fwxmnt(k+6,6-l) = fwxmnt(k+6-k*2,l+6)
                  fwymnt(k+6,6-l) = fwymnt(k+6-k*2,l+6)

               endif

 1300       continue

 1400    continue

      endif

      return
      end

      subroutine series(x,nmax,z1,z2,z1p,z2p)

c------------------------------------------------------------------------------
c     Called by sar2
c
c     This series approximated in several places in sar2.
c
```

```
c     Calls: none
c
c-----------------------------------------------------------------------------

c.....use x=r/(2l) and these relations:
c.....  z1r(n)  = -z1r(n-1)*x**4/(2n(2n-1))**2,  z1r(0)=1, n=1,...
c.....  z1r'(n) = (2n/x)*z1r(n)
c.....  z2r(n)  = (2n/x)**2*z1r(n)
c.....  z2r'(n) = ((2n-1)/x)*z2r(n)

      z1 = 0.
      z1p = 0.
      z2 = 0.
      z2p = 0.
      if(x.eq.0.) go to 200
      x4m = -x**4
      a2n = 0.
      z1rn = 1.

      do 100 n=1,nmax

         a2n = a2n+2.
         z1rn = z1rn*x4m/(a2n*(a2n-1.))**2
         z1 = z1+z1rn
         tmp1 = a2n/x
         z2rn = tmp1*tmp1*z1rn
         z2 = z2+z2rn
         z1p = z1p+tmp1*z1rn
         z2p = z2p+((a2n-1.)/x)*z2rn

  100 continue

  200 z1 = z1+1.

      return
      end

      subroutine sfl(attack,iyear)

c-----------------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for silo floor.
c
c     Calls: ccrack
c-----------------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht

      dimension flfdpx(11,11),flfdpy(11,11),flfspx(11,11),flfspy(11,11)

      data pi/3.141592653589793/
      data strred/.9/

c.....Calculate time-dependent parameters used in cracking analysis.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
```

```
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      fldstx = cmthk(1,3)-tencvx(1,3)
      fldsty = cmthk(1,3)-tencvy(1,3)
      starcm = 0.
      startn = stlrad(3)**2*pi/stlspc(3)
      cnmnti = cmthk(1,3)**3/12.
      crkmtf = cnmnti/(0.5*cmthk(1,3))*rupmod

c.....Calculate ultimate strength for floor.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c1 = .003*stlmod*starcm*comcvx(1,3)
      c2 = .003*stlmod*starcm*comcvy(1,3)
      axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
      axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

      if(axisn1 .le. comcvx(1,3)) then

         cmblk = startn*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,strred*stlyld*startn*(fldstx-cmblk/2.))

      else

         csstrs = (axisn1-comcvx(1,3))/axisn1*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,strred*(as1*stlyld*(fldstx-cmblk/2.)+
     #            starcm*csstrs*(fldstx-comcvx(1,3))))

      endif

      if(axisn2 .le. comcvy(1,3)) then

         cmblk = startn*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,strred*stlyld*startn*(fldsty-cmblk/2.))

      else

         csstrs = (axisn2-comcvy(1,3))/axisn2*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,strred*(as1*stlyld*(fldsty-cmblk/2.)+
     #            starcm*csstrs*(fldsty-comcvy(1,3))))

      endif

c.....Calculate cracking moment of inertia for floor for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      ccx = comcvx(1,3)*starcm*(ratmod-1.)-fldstx*ratmod*startn
      ccy = comcvy(1,3)*starcm*(ratmod-1.)-fldsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3+starcm*(ratmod-1.)*(axneux-comcvx(1,3))
     #         **2+ratmod*startn*(fldstx-axneux)**2
      crmtiy = 0.333*axneuy**3+starcm*(ratmod-1.)*(axneuy-comcvy(1,3))
     #         **2+ratmod*startn*(fldsty-axneuy)**2

c.....Calculate cracking due to shear for floor of silo.

      xk = (1.6+2.4*(silrad/silrad-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*fldstx
      shrsty = 1.7*sqrt(comstr)*fldsty
```

```
      krkx = 0
      krky = 0
      frac(3) = 0.
      aper(3) = 0.
      ii = 0

      do 200 k=1,6

         do 100 l=1,6

            frcwdx = 0.
            frcwdy = 0.

            if(flshr .ge. shrstx) then

               if (flxmnt(k,l) .gt. 0.) then
                  tmp = amin1(flshr/flxmnt(k,l)*fldstx,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/fldstx*
     #                  tmp)*fldstx,3.5*sqrt(comstr)*fldstx)

               else

                  vcr = 3.5*sqrt(comstr)*fldstx

               endif

               if(flshr .ge. vcr) then

                  sfrcdx = cmthk(l,3)
                  sfrcwx = 0.013
                  sfrcsx = silrad/5.

               endif

            else

               sfrcdx = 0.
               sfrcwx = 0.
               sfrcsx = 0.

            endif

c.....Calculate fracture characteristics for floor due to bending.

            if (flxmnt(k,l) .ge. crkmtf) then

               if (tencvx(l,3).eq.0. .and. flfspx(k,l).eq.0.) then
                  q = stlrad(3)**2*1.571/(stlspc(3)*otncvx(l,3))
                  if (stlrad(3) .lt. 1.e-15) q = ostlrd(3)**2*1.571/
     #               (stlspc(3)*otncvx(l,3))

               elseif (stlrad(3).lt.1.e-15 .and. flfspx(k,l).eq.0.)then

                  q = ostlrd(3)**2*1.571/(stlspc(3)*tencvx(l,3))

               elseif (tencvx(l,3) .gt. 0. .and. stlrad(3) .ge.
     #                1.e-15) then

                  q = stlrad(3)**2*1.571/(stlspc(3)*tencvx(l,3))

               endif

               if (stlrad(3) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(3)*stlspc(3)/q)

               elseif (stlrad(3).lt.1.e-15 .and. flfspx(k,l).eq.0.)then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(3)*stlspc(3)/q)

               endif

               if (flfspx(k,l).eq.0. .or. flfspx(k,l).ge.2.*frspce)
```

```
     #            flfspx(k,l) = frspce

            endif

c.....X-moments exceed cracking moment but not ultimate strength of floor.

            if (flxmnt(k,l).ge.crkmtf .and. flxmnt(k,l).lt.flustx)then

               efmntx = (crkmtf/flxmnt(k,l))**3*cnmnti+(1.-(crkmtf/
     #                  flxmnt(k,l))**3)*crmtix
               strsmx = flxmnt(k,l)*axneux/efmntx
               stltnx = ratmod*flxmnt(k,l)*(fldstx-axneux)/efmntx
               axsnex = fldstx/(stltnx/stlmod+strsmx/conmod)*(stltnx/
     #                  stlmod+csstrn)+tencvx(1,3)
               betax = axsnex/(axsnex-tencvx(1,3))
               flfdpx(k,l) = axsnex
               frcwdx = flfspx(k,l)*(stltnx/stlmod*betax+csstrn)

            endif

c.....X-moments exceed ultimate strength of floor.

            if (flxmnt(k,l).ge.flustx .and.
     #         flfdpx(k,l).lt.cmthk(1,3)) then

               flfdpx(k,l) = cmthk(1,3)
               frcwdx = amin1((stlyld/stlmod+csstrn)*flfspx(k,l),3.e-3*
     #                  flfspx(k,l))

            endif

c.....Perform calculations for y direction of floor. Start with shear
c.....cracking calculations.

            if(flshr .ge. shrsty) then

               if (flymnt(k,l) .gt. 0.) then

                  tmp = amin1(flshr/flymnt(k,l)*fldsty,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/fldsty*
     #                  tmp)*fldsty,3.5*sqrt(comstr)*fldsty)

               else

                  vcr = 3.5*sqrt(comstr)*fldsty

               endif

               if(flshr .ge. vcr) then

                  sfrcdy = cmthk(1,3)
                  sfrcwy = 0.013
                  sfrcsy = silrad/5.

               endif

            else

               sfrcdy = 0.
               sfrcwy = 0.
               sfrcsy = 0.

            endif

c.....Calculate fracture characteristics for y direction.

            if (flymnt(k,l) .ge. crkmtf) then

               if (tencvy(1,3).eq.0. .and. flfspy(k,l).eq.0.) then

                  q = stlrad(3)**2*1.571/(stlspc(3)*otncvy(1,3))
                  if (stlrad(3) .lt. 1.e-15) q = ostlrd(3)**2*1.571/
     #               (stlspc(3)*otncvy(1,3))
```

```
               elseif (stlrad(3).lt.1.e-15 .and. flfspy(k,l).eq.0.)then

                  q = ostlrd(3)**2*1.571/(stlspc(3)*tencvy(1,3))

               elseif (tencvy(1,3).gt.0. .and. stlrad(3).ge.1.e-15)then

                  q = stlrad(3)**2*1.571/(stlspc(3)*tencvy(1,3))

               endif

               if (stlrad(3) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(3)*stlspc(3)/q)

               elseif (stlrad(3).lt.1.e-15 .and. flfspy(k,l).eq.0.)then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(3)*stlspc(3)/q)

               endif

               if (flfspy(k,l).eq.0. .or. flfspy(k,l).ge.2.*frspce)
     #            flfspy(k,l) = frspce

            endif

c.....Y-moments exceed cracking moment but not ultimate strength of floor.

            if (flymnt(k,l).ge.crkmtf .and. flymnt(k,l).lt.flusty)then

               efmnty = (crkmtf/flymnt(k,l))**3*cnmnti+(1.-(crkmtf/
     #                  flymnt(k,l))**3)*crmtiy
               strsmy = flymnt(k,l)*axneuy/efmnty
               stltny = ratmod*flymnt(k,l)*(fldsty-axneuy)/efmnty
               axsney = fldsty/(stltny/stlmod+strsmy/conmod)*(stltny/
     #                  stlmod+csstrn)+tencvy(1,3)
               betay = axsney/(axsney-tencvy(1,3))
               flfdpy(k,l) = axsney
               frcwdy = flfspy(k,l)*(stltny/stlmod*betay+csstrn)

            endif

c.....Y-moments exceed ultimate strength of floor.

            if (flymnt(k,l).ge.flusty .and.
     #         flfdpy(k,l).lt.cmthk(1,3)) then

               flfdpy(k,l) = cmthk(1,3)
               frcwdy = amin1((stlyld/stlmod+csstrn)*flfspy(k,l),3.e-3*
     #                  flfspy(k,l))

            endif

c.....Calculate cracking due to corrosion once it begins.

            if (icrflg(3).eq.1 .and. (k+l).eq.2) call ccrack(3,iyear)

c.....Calculate average crack characteristics for floor.

            if (cmthk(1,3) .eq. 0.) then
               aper(3) = 0.
               frac(3) = 0.
               return

            else
               fmax = .75*cmthk(1,3)
               depth = amax1(flfdpx(k,l),sfrcdx,crfrcd(3))

               if (depth .ge. fmax) then
                  tmp1 = 0.
                  tmp2 = 0.
                  tmp3 = 0.
                  krkx = krkx+1
                  if (flfspx(k,l) .gt. 0.) tmp1 = silrad/5./flfspx(k,l)
```

```
                  if (crfrcs(3) .gt. 0.) tmp2 = silrad/5./crfrcs(3)
                  if (sfrcsx .gt. 0.) tmp3 = silrad/5./sfrcsx
                  tmp = tmp1+tmp2+tmp3
                  aper(3) = aper(3)+(frcwdx*tmp1+crfrcw(3)*tmp2+sfrcwx*
     #                      tmp3)/tmp
                  frac(3) = frac(3)+2.*cmthk(1,3)*(frcwdx*silrad/5.*
     #                      tmp1+sfrcwx*pi*silrad*tmp3)

               endif

               depth = amax1(flfdpy(k,1),sfrcdy,crfrcd(3))

               if (depth .ge. fmax) then
                  tmp1 = 0.
                  tmp2 = 0.
                  tmp3 = 0.
                  krky = krky+1
                  if (flfspy(k,1) .gt. 0.) tmp1 = silrad/5./flfspy(k,1)
                  if (crfrcs(3) .gt. 0.) tmp2 = silrad/5./crfrcs(3)
                  if (sfrcsy .gt. 0.) tmp3 = silrad/5./sfrcsy
                  tmp = tmp1+tmp2+tmp3
                  aper(3) = aper(3)+(frcwdy*tmp1+crfrcw(3)*tmp2+sfrcwy*
     #                      tmp3)/tmp
                  frac(3) = frac(3)+2.*cmthk(1,3)*(frcwdy*silrad/5.*
     #                      tmp1+sfrcwy*pi*silrad*tmp3)

               endif

            endif

  100    continue

  200 continue

      frac(3) = frac(3)+crfrac(3)

      if (frac(3) .gt. 0.) then

        icrack(3) = 1
        frac(3) = frac(3)/(cmthk(1,3)*pi*silrad**2)
        aper(3) = aper(3)/(krkx+krky)*2.54

      endif

      return
      end

      subroutine srf(attack,iyear)

c-----------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for silo roof.
c
c     Calls: ccrack
c-----------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
```

```
      dimension rffdpx(11,11),rffdpy(11,11),rffspx(11,11),rffspy(11,11)

      data pi/3.141592653589793/
      data strred/.9/

c.....Calculate time-dependent parameters used in cracking analysis.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      rfdstx = cmthk(1,1)-tencvx(1,1)
      rfdsty = cmthk(1,1)-tencvy(1,1)
      starcm = 0.
      startn = stlrad(1)**2*pi/stlspc(1)
      cnmnti = cmthk(1,1)**3/12.
      crkmtr = cnmnti/(0.5*cmthk(1,1))*rupmod

c.....Calculate ultimate strength for roof.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c1 = .003*stlmod*starcm*comcvx(1,1)
      c2 = .003*stlmod*starcm*comcvy(1,1)
      axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
      axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

      if(axisn1 .le. comcvx(1,1)) then
         cmblk = startn*stlyld/(0.85*comstr)
         rfustx = amax1(crkmtr,strred*stlyld*startn*(rfdstx-cmblk/2.))

      else

         csstrs = (axisn1-comcvx(1,1))/axisn1*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         rfustx = amax1(crkmtr,strred*(as1*stlyld*(rfdstx-cmblk/2.)+
     #            starcm*csstrs*(rfdstx-comcvx(1,1))))

      endif

      if(axisn2 .le. comcvy(1,1)) then
         cmblk = startn*stlyld/(0.85*comstr)
         rfusty = amax1(crkmtr,strred*stlyld*startn*(rfdsty-cmblk/2.))

      else

         csstrs = (axisn2-comcvy(1,1))/axisn2*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         rfusty = amax1(crkmtr,strred*(as1*stlyld*(rfdsty-cmblk/2.)+
     #            starcm*csstrs*(rfdsty-comcvy(1,1))))

      endif

c.....Calculate cracking moment of inertia for roof for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      ccx = comcvx(1,1)*starcm*(ratmod-1.)-rfdstx*ratmod*startn
      ccy = comcvy(1,1)*starcm*(ratmod-1.)-rfdsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3+starcm*(ratmod-1.)*(axneux-comcvx(1,1))
     #         **2+ratmod*startn*(rfdstx-axneux)**2
      crmtiy = 0.333*axneuy**3+starcm*(ratmod-1.)*(axneuy-comcvy(1,1))
```

```
     #         **2+ratmod*startn*(rfdsty-axneuy)**2

c.....Calculate cracking due to shear for roof of silos.

      xk = (1.6+2.4*(silrad/silrad-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*rfdstx
      shrsty = 1.7*sqrt(comstr)*rfdsty
      krkx = 0
      krky = 0
      frac(1) = 0.
      aper(1) = 0.

      do 200 k=1,6

         do 100 l=1,6

            frcwdx = 0.
            frcwdy = 0.

            if(rfshr .ge. shrstx) then

               if (rfxmnt(k,l) .gt. 0.) then

                  tmp = amin1(rfshr/rfxmnt(k,l)*rfdstx,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/rfdstx*
     #                  tmp)*rfdstx,3.5*sqrt(comstr)*rfdstx)

               else

                  vcr = 3.5*sqrt(comstr)*rfdstx

               endif

               if(rfshr .ge. vcr) then

                  sfrcdx = cmthk(1,1)
                  sfrcwx = 0.013
                  sfrcsx = silrad/5.

               endif

            else

               sfrcdx = 0.
               sfrcwx = 0.
               sfrcsx = 0.

            endif

c.....Calculate fracture characteristics for roof due to bending.

            if (rfxmnt(k,l) .ge. crkmtr) then

               if (tencvx(1,1).eq.0. .and. rffspx(k,l).eq.0.) then

                  if (stlrad(1) .lt. 1.e-15) then
                     q = ostlrd(1)**2*1.571/(stlspc(1)*otncvx(1,1))

                  else

                     q = stlrad(1)**2*1.571/(stlspc(1)*otncvx(1,1))

                  endif

               elseif (stlrad(1).lt.1.e-15 .and. rffspx(k,l).eq.0.)then

                  q = ostlrd(1)**2*1.571/(stlspc(1)*tencvx(1,1))

               elseif (tencvx(1,1).gt.0. .and. stlrad(1).ge.1.e-15)then

                  q = stlrad(1)**2*1.571/(stlspc(1)*tencvx(1,1))

               endif
```

```
            if (stlrad(1) .ge. 1.e-15) then

               frspce = 0.5*xk*sqrt(2.*stlrad(1)*stlspc(1)/q)

            elseif (stlrad(1).lt.1.e-15 .and. rffspx(k,1).eq.0.)then

               frspce = 0.5*xk*sqrt(2.*ostlrd(1)*stlspc(1)/q)

            endif

            if (rffspx(k,1).eq.0. .or. rffspx(k,1).ge.2.*frspce)
     #         rffspx(k,1) = frspce

         endif

c.....X-moments exceed cracking moment but not ultimate strength of roof.

         if (rfxmnt(k,1).ge.crkmtr .and. rfxmnt(k,1).lt.rfustx)then

            efmntx = (crkmtr/rfxmnt(k,1))**3*cnmnti+(1.-(crkmtr/
     #               rfxmnt(k,1))**3)*crmtix
            strsmx = rfxmnt(k,1)*axneux/efmntx
            stltnx = ratmod*rfxmnt(k,1)*(rfdstx-axneux)/efmntx
            axsnex = rfdstx/(stltnx/stlmod+strsmx/conmod)*(stltnx/
     #               stlmod+csstrn)+tencvx(1,1)
            betax = axsnex/(axsnex-tencvx(1,1))
            rffdpx(k,1) = axsnex
            frcwdx = rffspx(k,1)*(stltnx/stlmod*betax+csstrn)

         endif

c.....X-moments exceed ultimate strength of roof.

         if (rfxmnt(k,1).ge.rfustx .and. rffdpx(k,1).lt.
     #      cmthk(1,1)) then

            rffdpx(k,1) = cmthk(1,1)
            frcwdx = amin1((stlyld/stlmod+csstrn)*rffspx(k,1),
     #               3.e-3*rffspx(k,1))

         endif

c.....Perform calculations for y direction of roof. Start with shear
c.....cracking calculations.

         if(rfshr .ge. shrsty) then

            if (rfymnt(k,1) .gt. 0.) then

               tmp = amin1(rfshr/rfymnt(k,1)*rfdsty,1.)
               vcr = amin1((1.9*sqrt(comstr)+2500.*startn/rfdsty*
     #               tmp)*rfdsty,3.5*sqrt(comstr)*rfdsty)

            else

               vcr = 3.5*sqrt(comstr)*rfdsty

            endif

            if (rfshr .ge. vcr) then

               sfrcdy = cmthk(1,1)
               sfrcwy = 0.013
               sfrcsy = silrad/5.

            endif

         else

            sfrcdy = 0.
            sfrcwy = 0.
            sfrcsy = 0.
```

```
            endif

c.....Calculate fracture characteristics for y direction.

            if (rfymnt(k,l) .ge. crkmtr) then

               if (tencvy(1,1) .eq. 0. .and. rffspy(k,l).eq.0.) then

                  q = stlrad(1)**2*1.571/(stlspc(1)*otncvy(1,1))
                  if (stlrad(1) .lt. 1.e-15) q = ostlrd(1)**2*1.571/
     #               (stlspc(1)*otncvy(1,1))

               elseif (stlrad(1).lt.1.e-15 .and. rffspy(k,l).eq.0.)then

                  q = ostlrd(1)**2*1.571/(stlspc(1)*tencvy(1,1))

               elseif (tencvy(1,1).gt.0. .and. stlrad(1).ge.1.e-15)then

                  q = stlrad(1)**2*1.571/(stlspc(1)*tencvy(1,1))

               endif

               if (stlrad(1) .ge. 1.e-15) then

                  frspce = 0.5*xk*sqrt(2.*stlrad(1)*stlspc(1)/q)

               elseif (stlrad(1).lt.1.e-15 .and. rffspy(k,l).eq.0.)then

                  frspce = 0.5*xk*sqrt(2.*ostlrd(1)*stlspc(1)/q)

               endif

               if (rffspy(k,l).eq.0. .or. rffspy(k,l).ge.2.*frspce)
     #             rffspy(k,l) = frspce

            endif

c.....Y-moments exceed cracking moment but not ultimate strength of roof.

            if (rfymnt(k,l).ge.crkmtr .and. rfymnt(k,l).lt.rfusty)then

               efmnty = (crkmtr/rfymnt(k,l))**3*cnmnti+(1.-(crkmtr/
     #                  rfymnt(k,l))**3)*crmtiy
               strsmy = rfymnt(k,l)*axneuy/efmnty
               stltny = ratmod*rfymnt(k,l)*(rfdsty-axneuy)/efmnty
               axsney = rfdsty/(stltny/stlmod+strsmy/conmod)*(stltny/
     #                  stlmod+csstrn)+tencvy(1,1)
               betay = axsney/(axsney-tencvy(1,1))
               rffdpy(k,l) = axsney
               frcwdy = rffspy(k,l)*(stltny/stlmod*betay+csstrn)

            endif

c.....Y-moments exceed ultimate strength of roof.

            if (rfymnt(k,l).ge.rfusty .and.
     #         rffdpy(k,l).lt.cmthk(1,1)) then

               rffdpy(k,l) = cmthk(1,1)
               frcwdy = amin1((stlyld/stlmod+csstrn)*rffspy(k,l),
     #                  3.e-3*rffspy(k,l))

            endif

c.....Calculate cracking due to corrosion once it begins.

            if (icrflg(1).eq.1 .and. (k+1).eq.2) call ccrack(1,iyear)

c.....Calculate average crack characteristics for roof.

            if (cmthk(1,1) .eq. 0.) then

               frac(1) = 0.
```

```
               aper(1) = 0.

               return

            else

               fmax = .75*cmthk(1,1)
               depth = amax1(rffdpx(k,1),sfrcdx,crfrcd(1))

               if (depth .ge. fmax) then
                  tmp1 = 0.
                  tmp2 = 0.
                  tmp3 = 0.
                  krkx = krkx+1
                  if (rffspx(k,1) .gt. 0.) tmp1 = silrad/5./rffspx(k,1)
                  if (crfrcs(1) .gt. 0.) tmp2 = silrad/5./crfrcs(1)
                  if (sfrcsx .gt. 0.) tmp3 = silrad/5./sfrcsx
                  tmp = tmp1+tmp2+tmp3
                  aper(1) = aper(1)+(frcwdx*tmp1+crfrcw(1)*tmp2+sfrcwx*
     #                      tmp3)/tmp
                  frac(1) = frac(1)+2.*cmthk(1,1)*(frcwdx*silrad/5.*
     #                      tmp1+sfrcwx*pi*silrad*tmp3)

               endif

               depth = amax1(rffdpy(k,1),sfrcdy,crfrcd(1))

               if (depth .ge. fmax) then
                  tmp1 = 0.
                  tmp2 = 0.
                  tmp3 = 0.
                  krky = krky+1
                  if (rffspy(k,1) .gt. 0.) tmp1 = silrad/5./rffspy(k,1)
                  if (crfrcs(1) .gt. 0.) tmp2 = silrad/5./crfrcs(1)
                  if (sfrcsy .gt. 0.) tmp3 = silrad/5./sfrcsy
                  tmp = tmp1+tmp2+tmp3
                  aper(1) = aper(1)+(frcwdy*tmp1+crfrcw(1)*tmp2+sfrcwy*
     #                      tmp3)/tmp
                  frac(1) = frac(1)+2.*cmthk(1,1)*(frcwdy*silrad/5.*
     #                      tmp1+sfrcwy*pi*silrad*tmp3)

               endif

            endif

  100    continue

  200 continue

      frac(1) = frac(1)+crfrac(1)

      if (frac(1) .gt. 0.) then

         icrack(1) = 1
         frac(1) = frac(1)/(cmthk(1,1)*pi*silrad**2)
         aper(1) = aper(1)/(krkx+krky)*2.54

      endif

      return
      end

      subroutine sulfate(iyear)

c-----------------------------------------------------------------------
c     Called by concrete
c
c     Calculates loss of concrete thickness due to sulfate attack.
c
c     Calls: none
c-----------------------------------------------------------------------

      common/chemcl/cl,co2,o2,so4i,so4o,xmg2,dfalk,dfcaoh,dfcl,dfco2,
```

```
     #       dfo2,dfso4,casol,crbsol,xmgsol
      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht

c.....Rate of degradation calculated as per Atkinson and Hearne.

      if (iyear .eq. 1) then

c.....Begin outside disposal facility calculations.

         tmp=1.24
         so4o = so4o*1000.

100      continue

c.....Estimate ettringite concentration.

         ce = wtcmnt*tmp

c.....Calculate reaction zone thickness at which spalling occurs.

         xspl = 2.*1.*10.*(1-conpsn)/(yngmod*(1.8e-6*ce)**2)

c.....Calculate time when spalling occurs.

         tspl = xspl**2*ce/(2.*dfso4*so4o)
         t = 10.**(tmp/.32-alog10(so4o)+alog10(3577.)+alog10(12.2))
         tmp = tmp*.99

         if(tspl.lt.t) go to 100

c.....Concrete loss from outside of disposal facility.

         slfo = xspl*39.37/tspl*3.15e7

c.....Begin inside disposal facility calculations.

         tmp = 1.24
         so4i = so4i * 1000.

200      continue

c.....Estimate ettringite concentration.

         ce = wtcmnt*tmp

c.....Calculate reaction zone thickness at which spalling occurs.

         xspl = 2.*1.*10.*(1-conpsn)/(yngmod*(1.8e-6*ce)**2)

c.....Calculate time when spalling occurs.

         tspl = xspl**2*ce/(2.*dfso4*so4i)
         t = 10.**(tmp/.32-alog10(so4i)+alog10(3577.)+alog10(12.2))
         tmp = tmp*.99

         if(tspl.lt.t) go to 200

c.....Concrete loss from inside of disposal facility.

         slfi = xspl*39.37/tspl*3.15e7

      endif
```

```
c.....Update total member, compression, and tension face cover thicknesses
c.....for silo and well.

      if(idflag.eq.1 .or. idflag.eq.3) then

         do 300 i=1,3

            cmthk(1,i) = amax1(0.,cmthk(1,i)-(slfi+slfo))
            comcvx(1,i) = amax1(0.,comcvx(1,i)-slfo)
            comcvy(1,i) = amax1(0.,comcvy(1,i)-slfo)
            tencvx(1,i) = amax1(0.,tencvx(1,i)-slfi)
            tencvy(1,i) = amax1(0.,tencvy(1,i)-slfi)

  300    continue

      endif

      if(idflag .gt. 1) then

         do 400 i=1,3,2

            cmthk(2,i) = amax1(0.,cmthk(2,i)-(slfi+slfo))
            comcvx(2,i) = amax1(0.,comcvx(2,i)-slfo)
            comcvy(2,i) = amax1(0.,comcvy(2,i)-slfo)
            tencvx(2,i) = amax1(0.,tencvx(2,i)-slfi)
            tencvy(2,i) = amax1(0.,tencvy(2,i)-slfi)

  400    continue

      endif

      return
      end

      subroutine swl(attack,iyear)

c------------------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for silo wall.
c
c     Calls: ccrack
c------------------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/wlforc/wlcmfr(11),wlxrc(11),wlwxrc(11),wwcmfr(11)

      dimension wlfrdp(11),wlfrsp(11)

      data pi,sstred/3.141592653589793,0.9/

c.....Calculate time-dependent parameters used in cracking analysis.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      wldstx = cmthk(1,2)-tencvx(1,2)
```

```
      wldsty = cmthk(1,2)-tencvy(1,2)
      starcm = 0.
      startn = 0.

      if (sttktn .gt. 0.) then

         rupmod = 7.5*sqrt(comstr)

      else

         rupmod = 3.25*sqrt(comstr)
         sstred = .65

      endif

      cnmnti = cmthk(1,2)**3/12.
      crkmtw = cnmnti/(0.5*cmthk(1,2))*rupmod

c.....Calculate ultimate strength for silo wall.

      a = .7225*comstr
      b = sttkcm*stlyld+.003*stlmod*starcm-startn*stlyld-sttktn*stlyld
      c = .003*stlmod*starcm*comcvx(1,2)
      axsneu = (-b+sqrt(b**2-4.*a*c))/(2.*a)

      if(axsneu .le. comcvx(1,2)) then

         cmblk = (startn*stlyld+sttktn*stlyld-sttkcm*stlyld)/
     #           (.85*comstr)
         templ = sstred*(stlyld*startn*(wldstx-cmblk/2.)+
     #           sttktn*stlyld*(cmthk(1,2)+sttktn/2.-cmblk/2.)+sttkcm*
     #           stlyld*(cmblk/2.+sttkcm/2.))
         wlustx = amaxl(crkmtw,templ)

      else

         csstrs = (axsneu-comcvx(1,2))/axsneu*.003*stlmod
         as2 = starcm*csstrs/stlyld
         asl = startn-as2
         cmblk = asl*stlyld/(0.85*comstr)
         templ = sstred*(asl*stlyld*(wldstx-cmblk/2.)+
     #           starcm*csstrs*(wldstx-comcvx(1,2))+sttktn*stlyld*
     #           (cmthk(1,2)+sttktn/2.-cmblk/2.)+sttkcm*stlyld*(cmblk/
     #           2.+sttkcm/2.))
         wlustx = amaxl(crkmtw,templ)

      endif

c.....Calculate cracking moment of inertia for walls for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      cc = comcvx(1,2)*starcm*(ratmod-1.)-wldstx*ratmod*startn
      rttl = (-bb+sqrt(bb**2-4.*aa*cc))/(2.*aa)
      rtt2 = (-bb-sqrt(bb**2-4.*aa*cc))/(2.*aa)
      axneu = rttl
      templ = 0.333*axneu**3+starcm*(ratmod-1.)*(axneu-comcvx(1,2))
     #        **2+ratmod*startn*(wldstx-axneu)**2
      temp2 =  ratmod*sttktn*
     #         (cmthk(1,2)+sttktn/2.-axneu)**2+ratmod*sttkcm*(axneu+
     #         sttkcm/2.)**2+ratmod*sttktn**3/12.+ratmod*
     #         sttkcm**3/12.
      crkmti = templ + temp2

c.....Calculate stability force for use in ring compression cracking analysis.

      tmpl = 0.85*0.7*comstr*(cmthk(1,2)-(starcm+startn))+stlyld*
     #       (starcm+startn)
      tmp2 = conmod*cmthk(1,2)**3/(12.*(1-conpsn**2))
      tmp5 = 0.
      nsave = 1

      do 100 n=2,10
```

```
         tmp3 = (n*slhght/(pi*silrad))**2
         tmp4 = tmp2/silrad**2*(n**2-1+(2*n**2-1.-conpsn)/(1+tmp3))+
     #          conmod*cmthk(1,2)*1./((n**2-1)*(1+tmp3)**2)

         if(tmp4.lt.tmp5 .or. tmp5.eq.0.) then

            tmp5 = tmp4
            nsave2 = n

         endif

  100 continue

      wlusrc = amin1(tmp1,tmp5)
      if (tmp5 .lt. tmp1) nsave = nsave2

c.....Calculate compression strength for compression cracking analysis
c.....for vertical wall.

      tmp1 = 0.55*0.7*cmthk(1,2)*comstr
      tmp4 = 0.

      do 200 m=1,5

         tmp3 = tmp2/slhght**2*m**2*pi**2+conmod*cmthk(1,2)*
     #          slhght**2/silrad**2*m**2*pi**2
         if (tmp3.lt.tmp4 .or. tmp4.eq.0.) tmp4 = tmp3

  200 continue

      wlustc = amin1(tmp1,tmp4)

c.....Begin cracking analysis.

      krkx = 0
      krky = 0
      frac(2) = 0.
      aper(2) = 0.
      xk = (1.6+2.4*(silrad/silrad-0.5))*0.29
      shrstr = 1.7*sqrt(comstr)*wldstx

      do 300 k=1,11
         frcwdy = 0.

c.....Cracking analysis due to ring compression and compression of
c.....vertical wall.

         if (wlxrc(k).ge.wlusrc .or. wlcmfr(k).ge.wlustc) then

            cmfrdp = cmthk(1,2)
            cmfrwd = .003*slhght/10.
            cmfrsp = nsave

         else

           cmfrdp = 0.
           cmfrwd = 0.
           cmfrsp = 0.

         endif

c.....Cracking analysis due to shear.

         mm = wlymnt(k)-wlcmfr(k)*(4.*cmthk(1,2)-cmthk(1,2))/8.
         tmp = 3.5*sqrt(comstr)*cmthk(1,2)*(1.+wlcmfr(k)/(500.*
     #         cmthk(1,2)))**.5

         if(mm .gt. 0.) then
            vcr = amin1((1.9*sqrt(comstr)+2500.*startn/cmthk(1,2)*
     #            wlyshr(k)*cmthk(1,2)/mm)*cmthk(1,2),tmp)
         else

            vcr = tmp
```

```
         endif

         if(wlyshr(k) .ge. vcr) then

            sfrcdy = cmthk(1,2)
            sfrcwy = 0.013
            sfrcsy = slhght/10.

         else

            sfrcdy = 0.
            sfrcwy = 0.
            sfrcsy = 0.

         endif

c.....Calculate fracture characteristics for horizontal (x) direction due to
c.....bending.

         if (wlymnt(k) .ge. crkmtw) wlfrsp(k) = slhght/10.

c.....Moments exceed cracking moment but not ultimate strength of wall.

         if (wlymnt(k).ge.crkmtw .and. wlymnt(k).lt.wlustx) then

            efmnt = (crkmtw/wlymnt(k))**3*cnmnti+(1.-(crkmtw/
     #              wlymnt(k))**3)*crkmti
            strsm = wlymnt(k)*axneu/efmnt
            stltn = ratmod*wlymnt(k)*(wldstx-axneu)/efmnt
            axsnex = wldstx/(stltn/stlmod+strsm/conmod)*(stltn/stlmod+
     #               csstrn)+tencvx(1,2)
            betax = axsnex/(axsnex-tencvx(1,2))
            wlfrdp(k) = axsnex
            frcwdy = wlfrsp(k)*(stltn/stlmod*betax+csstrn)

         endif

c.....Moments exceed ultimate strength of wall.

         if (wlymnt(k).ge.wlustx .and. wlfrdp(k).lt.cmthk(1,2)) then

            wlfrdp(k) = cmthk(1,2)
            frcwdy = amin1((stlyld/stlmod+csstrn)*wlfrsp(k),3.e-3*
     #               wlfrsp(k))

         endif

c.....Calculate average fracture characteristics.

         if (cmthk(1,2) .eq. 0.) then

            aper(2) = 0.
            frac(2) = 0.
            return

         else

            fmax = .75*cmthk(1,2)
            depth = amax1(wlfrdp(k),sfrcdy,cmfrdp)

            if (depth .ge. fmax) then

               tmp1 = 0.
               tmp2 = 0.
               tmp3 = 0.
               krky = krky+1
               if (wlfrsp(k) .gt. 0.) tmp1 = slhght/10./wlfrsp(k)
               if (sfrcsy .gt. 0.) tmp2 = slhght/10./sfrcsy
               if (cmfrsp .gt. 0.) tmp3 = cmfrsp
               tmp = tmp1+tmp2+tmp3
               aper(2) = aper(2)+(frcwdy*tmp1+sfrcwy*tmp2+
     #                   cmfrwd*tmp3)/tmp
               frac(2) = frac(2)+2.*silrad*pi*cmthk(1,2)*(frcwdy*tmp1+
```

```
     #                   sfrcwy*tmp2)

               if (tmp3 .eq. nsave2) then

                  frac(2) = frac(2)+slhght/10.*cmthk(1,2)*cmfrwd*tmp3

               else

                  frac(2) = frac(2)+2.*silrad*pi*cmthk(1,2)*cmfrwd*tmp3

               endif

            endif

         endif

  300 continue

      if (frac(2).gt.0.) icrack(2) = 1

      do 400 j=1,2

         if (frac(2) .gt. 0.) then

               frac(2) = frac(2)/(slhght*pi*((silrad+0.5*cmthk(1,2))
     #                   **2-(silrad-0.5*cmthk(1,2))**2))
               aper(2) = aper(2)/(krkx+krky)*2.54

         endif

  400 continue

      return
      end

      function sxierfc (x)

c-----------------------------------------------------------------------------
c     Called by: flothru
c
c     Function used in diffusion leaching calculations (2 december 1991).
c
c     Calls: none
c-----------------------------------------------------------------------------

      implicit double precision (a-h, o-z)

      common/numb/mmax

      data rsrpi / 5.641895835477563d-1/

      xsq = x**2
      u = rsrpi*xsq
      sum = rsrpi - x + u
      d = 6.d0
      e = 9.d0
      thm2m = -1.d0

      do 100 m=2,30

         mmax = m
         u = u*xsq*thm2m/d
         sum = sum + u

         if (abs(u/sum) .lt. 5.d-9) go to 110

         d = d + e
         e = e + 4.d0
         thm2m = thm2m - 2.d0

  100 continue

      write(*,*) 'sxierfc: series did not converge'
```

```
  110 continue

      sxierfc = sum

      return
      end

      subroutine wfl(attack,iyear)

c--------------------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for silo floor.
c
c     Calls: none
c--------------------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isavel,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr

      data wstred/.65/

c.....Calculate time-dependent parameters used in cracking analysis.

      time = iyear*365.
      comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
      conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
      ratmod = stlmod/conmod
      rupmod = 7.5*sqrt(comstr)
      fldstx = cmthk(2,3)-tencvx(2,3)
      fldsty = cmthk(2,3)-tencvy(2,3)
      starcm = 0.
      startn = 0.
      cnmnti = cmthk(2,3)**3/12.
      crkmtf = cnmnti/(0.5*cmthk(2,3))*rupmod

c.....Calculate ultimate strength for floor.

      a = .7225*comstr
      b = .003*stlmod*starcm-startn*stlyld
      c1 = .003*stlmod*starcm*comcvx(2,3)
      c2 = .003*stlmod*starcm*comcvy(2,3)
      axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
      axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

      if(axisn1 .le. comcvx(2,3)) then
         cmblk = startn*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,wstred*stlyld*startn*(fldstx-cmblk/2.))

      else

         csstrs = (axisn1-comcvx(2,3))/axisn1*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flustx = amax1(crkmtf,wstred*(as1*stlyld*(fldstx-cmblk/2.)+
     #                 starcm*csstrs*(fldstx-comcvx(2,3))))
```

```
      endif

      if(axisn2 .le. comcvy(2,3)) then

         cmblk = startn*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,wstred*stlyld*startn*(fldsty-cmblk/2.))

      else

         csstrs = (axisn2-comcvy(2,3))/axisn2*.003*stlmod
         as2 = starcm*csstrs/stlyld
         as1 = startn-as2
         cmblk = as1*stlyld/(0.85*comstr)
         flusty = amax1(crkmtf,wstred*(as1*stlyld*(fldsty-cmblk/2.)+
     #            starcm*csstrs*(fldsty-comcvy(2,3))))

      endif

c.....Calculate cracking moment of inertia for floor for x and y directions.

      aa = 0.5
      bb = starcm*(ratmod-1.)+startn*ratmod
      ccx = comcvx(2,3)*starcm*(ratmod-1.)-fldstx*ratmod*startn
      ccy = comcvy(2,3)*starcm*(ratmod-1.)-fldsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3+starcm*(ratmod-1.)*(axneux-comcvx(2,3))
     #         **2+ratmod*startn*(fldstx-axneux)**2
      crmtiy = 0.333*axneuy**3+starcm*(ratmod-1.)*(axneuy-comcvy(2,3))
     #         **2+ratmod*startn*(fldsty-axneuy)**2

c.....Calculate shear cracking.

      xk = (1.6+2.4*(wlrad/wlrad-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*fldstx
      shrsty = 1.7*sqrt(comstr)*fldsty

      do 200 k=1,6

         do 100 l=1,6

            if(flwshr .ge. shrstx) then

               if (fwxmnt(k,l) .gt. 0.) then

                  tmp = amin1(flwshr/fwxmnt(k,l)*fldstx,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/fldstx*
     #                  tmp)*fldstx,3.5*sqrt(comstr)*fldstx)

               else

                  vcr = 3.5*sqrt(comstr)*fldstx

               endif

               if(flwshr .ge. vcr) ifail(3) = 1

            endif

c.....X-moments exceed cracking moment but not ultimate strength of floor.

            if (fwxmnt(k,l).ge.crkmtf .and. fwxmnt(k,l).lt.flustx) then

               efmntx = (crkmtf/fwxmnt(k,l))**3*cnmnti+(1.-(crkmtf/
     #                  fwxmnt(k,l))**3)*crmtix
               strsmx = fwxmnt(k,l)*axneux/efmntx
               stltnx = ratmod*fwxmnt(k,l)*(fldstx-axneux)/efmntx
               axsnex = fldstx/(stltnx/stlmod+strsmx/conmod)*(stltnx/
     #                  stlmod+csstrn)+tencvx(2,3)
```

```
               betax = axsnex/(axsnex-tencvx(2,3))
               depth = axsnex
               if(depth .ge. cmthk(2,3)) ifail(3) = 1

            endif

c.....X-moments exceed ultimate strength of floor.

            if (fwxmnt(k,1) .ge. flustx) ifail(3) = 1

c.....Perform shear cracking calculations for y direction of floor.

            if(flwshr .ge. shrsty) then

               if (fwymnt(k,1) .gt. 0.) then

                  tmp = amin1(flwshr/fwymnt(k,1)*fldsty,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/fldsty*
     #                  tmp)*fldsty,3.5*sqrt(comstr)*fldsty)

               else

                  vcr = 3.5*sqrt(comstr)*fldsty

               endif

                  if(flwshr .ge. vcr) ifail(3) = 1

            endif

c.....Y-moments exceed cracking moment but not ultimate strength of floor.

            if (fwymnt(k,1).ge.crkmtf .and. fwymnt(k,1).lt.flusty)then

               efmnty = (crkmtf/fwymnt(k,1))**3*cnmnti+(1.-(crkmtf/
     #                  fwymnt(k,1))**3)*crmtiy
               strsmy = fwymnt(k,1)*axneuy/efmnty
               stltny = ratmod*fwymnt(k,1)*(fldsty-axneuy)/efmnty
               axsney = fldsty/(stltny/stlmod+strsmy/conmod)*(stltny/
     #                  stlmod+csstrn)+tencvy(2,3)
               betay = axsney/(axsney-tencvy(2,3))
               depth = axsney
               if(depth .ge. cmthk(2,3)) ifail(3) = 1

            endif

c.....Y-moments exceed ultimate strength of floor.

            if (fwymnt(k,1) .ge. flusty) ifail(3) = 1

  100    continue

  200 continue

      return
      end

      subroutine wrf(attack,iyear)

c-----------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for well roof.
c
c     Calls: none
c-----------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isave1,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
```

```
#       yngmod
 common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
#       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
#       fwymnt(11,11),wlymnt(11),wwymnt(11)
 common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
 common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
#       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
#       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
#       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
#       tencvy(2,3),wstdns,wsthk,wstht
 common/well/stldns,stlpsn,wlhght,wlrad,wlstr

 data wstred/.65/

c.....Calculate time-dependent parameters used in cracking analysis.

 time = iyear*365.
 comstr = amin1(time/(cfa+cfb*time)*com28d*attack,constr*attack)
 conmod = 5.7e4*sqrt(comstr)/(1.+crpcof)
 ratmod = stlmod/conmod
 rupmod = 7.5*sqrt(comstr)
 rfdstx = cmthk(2,1)-tencvx(2,1)
 rfdsty = cmthk(2,1)-tencvy(2,1)
 starcm = 0.
 startn = 0.
 cnmnti = cmthk(2,1)**3/12.
 if(cmthk(2,1) .ne. 0.0)crkmtr = cnmnti/(0.5*cmthk(2,1))*rupmod

c.....Calculate ultimate strength for roof.

 a = .7225*comstr
 b = .003*stlmod*starcm-startn*stlyld
 c1 = .003*stlmod*starcm*comcvx(2,1)
 c2 = .003*stlmod*starcm*comcvy(2,1)
 axisn1 = (-b+sqrt(b**2-4.*a*c1))/(2.*a)
 axisn2 = (-b+sqrt(b**2-4.*a*c2))/(2.*a)

 if(axisn1 .le. comcvx(2,1)) then

    cmblk = startn*stlyld/(0.85*comstr)
    rfustx = amax1(crkmtr,wstred*stlyld*startn*(rfdstx-cmblk/2.))

 else

    csstrs = (axisn1-comcvx(2,1))/axisn1*.003*stlmod
    as2 = starcm*csstrs/stlyld
    as1 = startn-as2
    cmblk = as1*stlyld/(0.85*comstr)
    rfustx = amax1(crkmtr,wstred*(as1*stlyld*(rfdstx-cmblk/2.)+
#             starcm*csstrs*(rfdstx-comcvx(2,1))))

 endif

 if(axisn2 .le. comcvy(2,1)) then

    cmblk = startn*stlyld/(0.85*comstr)
    rfusty = amax1(crkmtr,wstred*stlyld*startn*(rfdsty-cmblk/2.))

 else

    csstrs = (axisn2-comcvy(2,1))/axisn2*.003*stlmod
    as2 = starcm*csstrs/stlyld
    as1 = startn-as2
    cmblk = as1*stlyld/(0.85*comstr)
    rfusty = amax1(crkmtr,wstred*(as1*stlyld*(rfdsty-cmblk/2.)+
#             starcm*csstrs*(rfdsty-comcvy(2,1))))

 endif

c.....Calculate cracking moment of inertia for roof for x and y directions.

 aa = 0.5
 bb = starcm*(ratmod-1.)+startn*ratmod
```

```
      ccx = comcvx(2,1)*starcm*(ratmod-1.)-rfdstx*ratmod*startn
      ccy = comcvy(2,1)*starcm*(ratmod-1.)-rfdsty*ratmod*startn
      rtt1x = (-bb+sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt1y = (-bb+sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      rtt2x = (-bb-sqrt(bb**2-4.*aa*ccx))/(2.*aa)
      rtt2y = (-bb-sqrt(bb**2-4.*aa*ccy))/(2.*aa)
      axneux = rtt1x
      axneuy = rtt1y
      crmtix = 0.333*axneux**3+starcm*(ratmod-1.)*(axneux-comcvx(2,1))
     #         **2+ratmod*startn*(rfdstx-axneux)**2
      crmtiy = 0.333*axneuy**3+starcm*(ratmod-1.)*(axneuy-comcvy(2,1))
     #         **2+ratmod*startn*(rfdsty-axneuy)**2

c.....Calculate cracking due to shear for roof of silos.

      xk = (1.6+2.4*(wlrad/wlrad-0.5))*0.29
      shrstx = 1.7*sqrt(comstr)*rfdstx
      shrsty = 1.7*sqrt(comstr)*rfdsty

      do 200 k=1,6

         do 100 l=1,6

            if(rfwshr .ge. shrstx) then

               if (rwxmnt(k,l) .gt. 0.) then

                  tmp = amin1(rfwshr/rwxmnt(k,l)*rfdstx,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/rfdstx*
     #                  tmp)*rfdstx,3.5*sqrt(comstr)*rfdstx)

               else

                  vcr = 3.5*sqrt(comstr)*rfdstx

               endif

               if(rfwshr .ge. vcr) ifail(1) = 1

            endif

c.....X-moments exceed cracking moment but not ultimate strength of roof.

            if (rwxmnt(k,l).ge.crkmtr .and. rwxmnt(k,l).lt.rfustx)then

               efmntx = (crkmtr/rwxmnt(k,l))**3*cnmnti+(1.-(crkmtr/
     #                  rwxmnt(k,l))**3)*crmtix
               strsmx = rwxmnt(k,l)*axneux/efmntx
               stltnx = ratmod*rwxmnt(k,l)*(rfdstx-axneux)/efmntx
               axsnex = rfdstx/(stltnx/stlmod+strsmx/conmod)*(stltnx/
     #                  stlmod+csstrn)+tencvx(2,1)
               betax = axsnex/(axsnex-tencvx(2,1))
               depth = axsnex
               if(depth .ge. 0.75*cmthk(2,1)) ifail(1) = 1

            endif

c.....X-moments exceed ultimate strength of roof.

            if (rwxmnt(k,l) .ge. rfustx) ifail(1) = 1

c.....Perform calculations for y direction of roof. Start with shear
c.....cracking calculations.

            if(rfwshr .ge. shrsty) then

               if (rwymnt(k,l) .gt. 0.) then

                  tmp = amin1(rfwshr/rwymnt(k,l)*rfdsty,1.)
                  vcr = amin1((1.9*sqrt(comstr)+2500.*startn/rfdsty*
     #                  tmp)*rfdsty,3.5*sqrt(comstr)*rfdsty)

               else
```

```
                  vcr = 3.5*sqrt(comstr)*rfdsty

               endif

               if (rfwshr .ge. vcr) ifail(1) = 1

            endif

c.....Y-moments exceed cracking moment but not ultimate strength of roof.

            if (rwymnt(k,1).ge.crkmtr .and. rwymnt(k,1).lt.rfusty)then

               efmnty = (crkmtr/rwymnt(k,1))**3*cnmnti+(1.-(crkmtr/
     #                  rwymnt(k,1))**3)*crmtiy
               strsmy = rwymnt(k,1)*axneuy/efmnty
               stltny = ratmod*rwymnt(k,1)*(rfdsty-axneuy)/efmnty
               axsney = rfdsty/(stltny/stlmod+strsmy/conmod)*(stltny/
     #                  stlmod+csstrn)+tencvy(2,1)
               betay = axsney/(axsney-tencvy(2,1))
               depth = axsney
               if(depth .ge. cmthk(2,1)) ifail(1) = 1

            endif

c.....Y-moments exceed ultimate strength of roof.

            if (rwymnt(k,1) .ge. rfusty) ifail(1) = 1

  100    continue

  200 continue

      return
      end

      subroutine wwl

c------------------------------------------------------------------------------
c     Called by: source2
c
c     Performs cracking analysis for silo wall.
c
c     Calls: none
c------------------------------------------------------------------------------

      common/clcult/annprc,aper(3),attk(2,3),crfrac(3),crfrcd(3),
     #       crfrcw(3),crfrcs(3),crpcof,csstrn,frac(3),icl(3),ico2(3),
     #       icrack(3),icrflg(3),ifail(3),isavel,isave2,ispl(3),ph(2,3),
     #       slfi,slfo,stlcor(3),xload,xperc(2)
      common/concrt/ca,cacon,cagw,cap,ccdns,ccon,ccpor,cfa,cfb,clcon,
     #       co3,com28d,conpsn,constr,phbeg,si,stlmod,stlyld,wcr,wtcmnt,
     #       yngmod
      common/moment/rfxmnt(11,11),rfymnt(11,11),rwxmnt(11,11),
     #       rwymnt(11,11),flxmnt(11,11),flymnt(11,11),fwxmnt(11,11),
     #       fwymnt(11,11),wlymnt(11),wwymnt(11)
      common/shear/rfshr,rfwshr,flshr,flwshr,wlyshr(11),wwyshr(11)
      common/silo/cmthk(2,3),comcvx(2,3),comcvy(2,3),cvrdns,cvrthk,
     #       flangl,idflag,ommthk(2,3),ostlrd(3),osttkc,osttkt,
     #       otncvx(2,3),otncvy(2,3),silrad,slangl,sldns,slhght,
     #       stlrad(3),stlspc(3),sttkcm,sttktn,submod,tencvx(2,3),
     #       tencvy(2,3),wstdns,wsthk,wstht
      common/well/stldns,stlpsn,wlhght,wlrad,wlstr
      common/wlforc/wlcmfr(11),wlxrc(11),wlwxrc(11),wwcmfr(11)

c.....Calculate ultimate strength of well under ring compression and
c.....allowable stress in tension.

      pusrc = amin1(2.*cmthk(2,2)*2.e6*cmthk(2,2)/wlrad*abs(1.-33.333*
     #        cmthk(2,2)/wlrad),3.e4*cmthk(2,2))
      pipshr = 0.4*1.7*cmthk(2,2)*wlstr

c.....Compare calculated ring compressions and shears to ultimate strength
c.....and allowable stress.
```

```
      do 100 k=1,11

         if (wlwxrc(k).ge.pusrc .or. wwyshr(k).ge.pipshr) ifail(2) = 1

c.....Calculate maximum stress due to axial compression and bending, and
c.....check for failure.

         if (cmthk(2,2) .gt. 0.) then

            pustr = wwcmfr(k)/cmthk(2,2)+6.*wwymnt(k)/cmthk(2,2)**2
            if (pustr .ge. wlstr) ifail(2) = 1

         else

            ifail(2) = 1

         endif

  100 continue

      return
      end
```

INTERNAL DISTRIBUTION

1. J. M. Begovich
2. G. R. Cunningham
3. D. L. Daugherty
4. J. R. Forgy, Jr.
5–10. A. S. Icenhour
11. J. A. Klein
12. D. C. Kocher
13. D. W. Lee
14. S. L. Loghry
15. R. J. Luxmoore
16. L. E. McNeese
17. D. E. Reichle
18. R. Salmon
19. T. F. Scanlan
20. R. W. Sharpe
21. R. D. Spence
22. S. N. Storch
23. F. J. Sweeney
24. J. D. Tauxe
25. M. L. Tharp
26. M. W. Tull
27. M. W. Yambert
28–29. Central Research Library
30. ORNL Laboratory Records, RC

EXTERNAL DISTRIBUTION

31. H. W. Godbee, 104 Tidewater Lane, Oak Ridge, TN 37830

32. L. F. Miller, University of Tennessee, Department of Nuclear Engineering, Pasqua Engineering Bldg., Knoxville, TN 37996-2300

33. R. Shuman, Rogers & Associates Engineering Corporation, 21124 E. Lakeshore Road, Big Fork, MT 59911

34. Office of Assistant Manager, Energy Research and Development, DOE-OR, P.O. Box 2001, Oak Ridge, TN 37831

35. Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831