UCB-NE-101

User's Manual

William W.-L. Lee

**Earth Sciences Division
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720**

February 1989

DISCLAIMER

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

Work supported by the Director, Office of Civilian Radioactive Waste Management, Office of System Integration and Regulation, Licensing and Compliance Division, of the U. S. Department of Energy under contract DE-AC03-76SF00098.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED *EB*

MASTER

INTRODUCTION

The purpose of this manual is to provide users of UCB-NE-101 with the information necessary to use UCB-NE-101 effectively.

DESCRIPTION OF THE CODE

UCB-NE-101 calculates the concentration of solubility-limited species as a function of space and time and its mass flux rates from a waste sphere buried in a nuclear waste repository in water-saturated rock. The waste is surrounded by one type of rock, and some distance away, there is another type of rock. The inner layer of rock can be a backfill around a nuclear waste package and the outer layer the natural rock. The mass flux calculated is at the interface of the two layers. The species concentration calculated is in the inner layer. A constant concentration of the species, usually the solubility, is specified at the waste sphere/inner layer interface. Dissolution and transport is governed by the solubility of the species, and diffusion in the porous media.

THEORY

Nuclear waste placed in geologic repositories in water-saturated rock will gradually dissolve. For many species such as actinides and rare earths, the process of dissolution is governed by the exterior flow field, and the chemical reaction rate or leaching rate [Chambré *et al.* 1987]. The need is to study the rate of dissolution, and the resultant concentration field of the dissolved species. Figure 1 shows the situation studied and some of the terms used. The waste sphere radius is R_0 and is surrounded by a layer of rock/sediment/backfill of thickness $b = R_1 - R_0$. At the waste surface, a constant species concentration of c_s is specified. The subscript 1 refers to the inner layer and the subscript 2 refers to the outer layer. The species retardation coefficient is K . The species diffusion coefficient is D_f and σ_1 and σ_2 are geometric factors for the two layers. The porosities of the layers are denoted by ϵ and the species concentration in ground water in the layers are c_1 and c_2 respectively.

By defining $\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{2}{r} \frac{\partial}{\partial r}$ we can write the governing equations for this mass transfer

$$\frac{\partial c_1}{\partial t} = D_1 \nabla^2 c_1 - \lambda c_1, \quad R_0 < r < R_1, t > 0, D_1 = \frac{\sigma_1 D_f}{K_1} \quad (1)$$

$$\frac{\partial c_2}{\partial t} = D_2 \nabla^2 c_2 - \lambda c_2, \quad R_1 < r < \infty, t > 0, D_2 = \frac{\sigma_2 D_f}{K_2} \quad (2)$$

$$c_1(r, 0) = 0, \quad R_0 < r \leq R_1, \quad (3)$$

$$c_2(r, 0) = 0, \quad R_1 \leq r < \infty \quad (4)$$

$$c_1(R_0, t) = c_s, \quad t \geq 0 \quad (5)$$

$$c_1(R_1, t) = c_2(R_1, t), \quad t \geq 0 \quad (6)$$

$$-\epsilon_1 \sigma_1 D_f \frac{\partial c_1}{\partial r} = -\epsilon_2 \sigma_2 D_f \frac{\partial c_2}{\partial r} \quad \text{at } r = R_1, \quad t \geq 0 \quad (7)$$

$$c_2(\infty, t) = 0, \quad t \geq 0 \quad (8)$$

The solution to (1) through (8) is given in Chambré *et al.* [1985]

$$\frac{c_1(r, t)}{c_s} = f(r) + e^{-\lambda t} \int_0^\infty \frac{e^{-D_1 \eta^2}}{1 + \left(\frac{\lambda}{D_1 \eta^2}\right)} I(r, \eta) d\eta, \quad R_0 \leq r \leq R_1, t \geq 0 \quad (9)$$

where

$$H(\eta) = [\epsilon_1' \eta \cos(\eta b) + \alpha \sin(\eta b)]^2 + [\beta \epsilon_2' \eta \sin(\eta b)]^2$$

$$f(r) = \frac{R_0}{r} \frac{1 + \delta(\frac{r}{R_1})}{1 + \delta(\frac{R_0}{R_1})}, \quad I(r, \eta) = - \left(\frac{2R_0\epsilon'_1\epsilon'_2\beta}{\pi r} \right) \frac{\eta \sin(\eta[r - R_0])}{H(\eta)}$$

$$\alpha = \frac{\epsilon'_2 - \epsilon'_1}{R_1}, \quad \beta = \sqrt{K'_2/K'_1}, \quad \delta = \frac{\epsilon'_2 - \epsilon'_1}{\epsilon'_2}$$

$$b = R_1 - R_0, \quad K'_\ell = K_\ell/\sigma_\ell, \quad \epsilon'_\ell = \epsilon_\ell/\sigma_\ell, \quad \ell = 1, 2$$

If $\lambda = 0$ in (9), we obtain the solution for a stable species.

ALGORITHM

This program computes the concentration of the species at the interface of the two layers and mass transfer rates at inner surface of the first layer and at the interface of the two layers. The transient solution is calculated using the D01AJF routine from the NAG library with the help of the exact steady-state solution.

INPUT

There are ten (10) input lines, all in free format. The first 9 lines are the physical properties of the system, described below. Each line gives 3 pieces of data: the initial value of this parameter, the increment multiplier of this parameter, and the number of times the calculations will be repeated using the different values of this parameter. For example, if the input line for the diffusion coefficient is "1e-5 10.0 3", which means the initial value used for D_f is 1×10^{-5} cm²/s, the increment multiplier is 10, and the number of evaluations is 3. Hence the entire set of calculations will be repeated three times, for values of D_f 1×10^{-5} cm²/s, 1×10^{-4} cm²/s, and 1×10^{-3} cm²/s, respectively.

The ten lines of input are for

Df - diffusion coefficient (cm²/sec)

K1 - retardation coefficient of inside layer

K2 - retardation coefficient of outside layer

E1 - porosity of inside layer

E2 - porosity of outside layer

R0 - radius of waste sphere (cm)

XL - insider layer thickness (cm)

T12 - species half life (years)

T - time (years).

To do calculations for 10^6 years, the ninth line should read

1., 10., 7.,

The last line has three (3) input data, EPS - relative error bound, IBUG - debug option, and CUT - upper limit of integral. Numerical experiments showed that for an upper limit of the integral in (9) of 20, the relative error bound for the integral is 10^{-6} . We have used values of CUT from 20 to 100. In practice, we often use

1.e - 4, 0., 20.,

as input for the last line.

The program currently reads from the keyboard as a default. In the DEC/VMS system the command

ASSIGN inputfile FOR005

will cause inputfile to be the input.

OUTPUT

The program currently sends output to the screen.

The output is in three parts. First, input data is repeated. Second, a set of numbers for
•mass flux from the waste sphere/solubility of the species (cm^3/s),
•mass flux at the interface of the layers/solubility of the species (cm^3/s),
•concentration at the interface of the layers,
all as a function of time, but WITHOUT radioactive decay.

The next set of lines are for the species. First, the half life read in is written out. Then comes a series of lines. In order, they are

- time (years),
- mass flux from the waste sphere/solubility of the species (cm^3/s),
- maximum error in the line just above,
- work space,
- number of subintervals used in numerical integration for the result above,
- mass flux at the interface of the layers/solubility of the species (cm^3/s),
- maximum error in the line just above,
- work space,
- number of subintervals used in numerical integration for the result above,
- concentration at the interface of the layers/solubility of the species,
- maximum error in the line just above,
- work space,
- number of subintervals used in numerical integration for the result above,
- The mass transfer coefficient (m/yr).

In Appendix C, the example output has been annotated. If more than one iteration has been requested for any one of the parameters, then the entire output shown in Appendix C is repeated as many times as requested.

RESOURCE REQUIREMENTS

UCB-NE-101 is written in FORTRAN and calls both the IMSL and NAG libraries. It operates on DEC/VAX and DEC/UNIX machines.

VERIFICATION & BENCHMARKING

UCB-NE-101 is an implementation of (9). It was verified through hand calculations. The correctness of UCB-NE-101 was also checked this way. Eq. (9) was given to several workers and they were asked to write computer programs independently. The outputs of the various programs are then compared or benchmarked. They gave identical results.

ACKNOWLEDGEMENTS

UCB-NE-101, was originally known as MASLAM. The author is Dr. H. C. Lung who kindly reviewed this *User's Manual* in draft form. Work was supported in part by the U. S. Department of Energy under contract DE-AC03-76SF00098.

REFERENCES

P. L. Chambré, T. H. Pigford, W. W.-L. Lee, J. Ahn, S. Kajiwar, C. L. Kim, H. Kimura, H. Lung, W. J. Williams and S. J. Zavoshy, *Mass Transfer and Transport in a Geologic Environment*, Report LBL-19430, 1985.

XBL 845-6908

Fig. 1. Geometry of spherical waste form with backfill layer.

APPENDIX A: CODE LISTING

program ucbn0101

```

c UCBNE101 - mass transfer from canister surface
c
c This program computes the concentration at the backfill-rock
c interface and mass transfer rates at canister surface and at
c backfill-rock interface from a spherical canister surrounded
c by a backfill layer with radioactive decay
c
c Transient solution using MAG DOLAUF routine with the help
c of exact steady state solution
c
c For early time, the semi-infinite medium assumption will be
c used and the single-region solution will be applied
c
c This program also computes the results without decay
c
c If the half life of the radionuclide is longer than or equal
c to 1E13 years it will be treated as non-decay nuclide
c
c Input parameters:
c
c There are ten (10) input lines, all in free format. The first 9
c lines are the physical properties of the system, described below.
c Each line has 3 input data: the initial value of this parameter,
c the increment multiplier of this parameter, and the number of
c evaluations used for this parameter. For example, the input line
c for diffusion coefficient is "1e-5 10.0 3", which means the initial
c value used for Df is 1e-5 cm**2/sec, the increment multiplier is
c 10, and the number of evaluations for Df is 3. Hence the values
c used for Df will be 1e-5, 1e-4, and 1e-3, respectively.
c
c   Df - diffusion coefficient (cm**2/sec)
c   K1 - retardation coefficient of backfill
c   K2 - retardation coefficient of rock
c   X1 - porosity of backfill
c   X2 - porosity of rock
c   R0 - radius of waste (cm)
c   XL - backfill thickness (cm)
c   T12 - half life (years)
c   T - time (years)
c
c The last line has three (3) input data, described below.
c
c   EPS - relative error bound
c   IBUG - debug option
c   CUT - upper limit of integral
c
c Output data:
c
c The output data will be mass transfer rates at both sides of
c backfill, the concentration at backfill-rock interface, and
c the mass transfer coefficient.
c
c -----
c
c N.C.Lung          11/06/85

```

```

real*8 sum,low,up,abe,df0,deldf,k10,delk1,k20,delk2,k11,k22
real*8 d,cl,c2,c3,t,eps,b,year,flux0(20),e10,dele1,e20,dele2
real*8 df,k1,k2,e1,e2,r0,x1,ak21,gamma,r1,pie,r00,delr0,cut
real*8 time(20),x10,delx1,t0,delt,mxe0(20),con(20),mxe(20)
real*8 t12,delt12,t120,t1,e,x0,x1,tr(20,6),hm(20),coeff,arg
real*8 flux01(20),flux1(20),flux11(20),con1(20),mxe1(20)
real*8 d1,d2,r01,a11,a12,col,co2,den,sm0,sml,smc,sh,ch,alg2
common /para/ d,cl,c2,c3,b,e,r1
data year/3.1536d7/, ahe/0.0/
pi=3.14159265358979
alg2=dlog(2d0)/year

c
c read df,k1,k2,e1,e2,r0,x1,t1/2, and time
c
    low=0.0
    read (5,*) df0,deldf,ndf
    read (5,*) k10,delk1,nk1
    read (5,*) k20,delk2,nk2
    read (5,*) e10,dele1,ne1
    read (5,*) e20,dele2,ne2
    read (5,*) r00,delr0,nr0
    read (5,*) x10,delx1,nx1
    read (5,*) t120,delt12,nt12
    read (5,*) t0,delt,nt

c
c read eps, ibug, and upper cut
c
    read (5,*) eps,ibug,cut
    write (6,90) eps,ibug,cut

c
    df=df0/deldf
    do 8 idf=1,ndf
    df=df*deldf
    k1=k10/delk1
    do 7 ik1=1,nk1
    k1=k1*delk1
    k2=k20/delk2
    do 6 ik2=1,nk2
    k2=k2*delk2
    x1=x10/delx1
    do 5 ix1=1,nx1
    x1=x1*delx1
    d=x1
    r0=r00/delr0
    do 4 ir0=1,nr0
    r0=r0*delr0
    r1=r0*x1
    coeff=3.1536e5/(4.0*pi*e1*r1)
    r01=r0/r1
    e1=e10/dele1
    do 3 ie1=1,ne1
    e1=e1*dele1
    k11=k1
    if (dabs(e1-1.0).lt.1e-4) k11=1.0
    d1=df/k11
    ts=d**2/d1/120.0

c
    ts=1e12*year
    cl=e1
    x0=4.0*pi*e1*df*r0**2
    x1=x0/r01
    e2=e20/dele2
    do 2 ie2=1,ne2
    e2=e2*dele2
    k22=k2
    if (dabs(e2-1.0).lt.1e-4) k22=1.0
    d2=df/k22

```

```

      sk21=dsqrt(k22/k11)
      c2=(a2-e1)/(r1)
      c3=sk21*a2
      gamma=(e1-e2)/a2
      if (lbug.gt.0) write (6,100) d,c1,c2,c3
      write (6,80) df,k11,k22,e1,e2,r0,x1
c
c
c  steady state parts of flux and concentration without decay
c
      sum=1.0/r0/(1.0+gamma*r0/r1)*x0
      ssc=r0/r1*(1.0+gamma)/(1.0+gamma*r0/r1)
c
c  beginning calculations
c
      t=t0/delt
      do 1 it=1,nt
      t=t+delt
      time(it)=t
      ty=t*year
      b=d1*ty
      if (ty.lt.ts) then
c
c  early time, use single region approx.
c
c  set concentration and flux at b-r interface to zero
c
      flux0(it)=x0*(1.0/dsqrt(pie*b)+1.0/r0)
      flux1(it)=0.0
      con(it)=0.0
      else
c
c  later time, use exact solution
c
      up=dsqrt(cut/b)
c
c  calculate flux at waste surface without decay
c
      error=eps
      icode=1
      call intgr1(f,0.0,low,up,error,sum,icode,n)
      if (n.gt.1e6) go to 1000
      mne0(it)=0.0
      if (sum.gt.0.0) mne0(it)=error/sum
      tr(it,1)=2.0*e1*a2*sk21/pie*sum*x0
      flux0(it)=ssm+tr(it,1)
      wsg0(it)=a
c
c  calculate flux at backfill-rock interface without decay
c
      error=eps
      icode=1
      call intgr1(h,0.0,low,up,error,sum,icode,n)
      if (n.gt.1e6) go to 1000
      mne1(it)=0.0
      if (sum.gt.0.0) mne1(it)=error/sum
      tr(it,3)=2.0*e1*a2*sk21/pie*sum*x1
      flux1(it)=ssm+tr(it,3)
      wsg1(it)=n
c
c  calculate concentration at b-r interface without decay
c
      error=eps
      icode=1
      call intgr1(g,0.0,low,up,error,sum,icode,n)
      if (n.gt.1e6) go to 1000
      mne2(it)=0.0

```



```

        if (sum.gt.0.0) mrec(it)=error/sum
        tr(it,5)=-2.0*e1*e2*sk21/pie*sum*r0/r1
        con(it)=ssc+tr(it,5)
        wpsc(it)=n
    end if
1 continue
    write (6,25)
    write (6,20) (time(i),i=1,nt)
    write (6,30) (flux0(i),i=1,nt)
    write (6,10) (flux1(i),i=1,nt)
    write (6,10) (con(i),i=1,nt)
c
c start decay part
c
    t12=t120/delt12
    do 11 it12=1,nt12
        t12=t12*delt12
        write (6,13) t12
c
c if half life .ge. 1E15 years, treated as non-decay
c
        if (t12.ge.1e15) go to 11
c
c otherwise begin calculation with decay
c
        t1=alg2/t12
c
c first calculate the steady state solutions
c
        a11=dsqrt(t1/d1)
        a12=dsqrt(t1/d2)
        co1=a11*e1
        co2=a12*e2+c2
        arg=a11*d
        sh=dsinh(arg)
        ch=dcosh(arg)
        den=co1*sh+co2*sh
        ssc=r01*co1/den
        sm0=x0/r0*(1.0+a11*r0*(co1*sh+co2*sh)/den)
        sm1=x1/r0*(ssc+a11*r0*co2/den)
c
c begin transient calculations
c
        t=t0/delt
        do 12 it=1,nt
            t=t*delt
            ty=t*year
            e=ti*ty
            b=d1*ty
            if (ty.lt.ts) then
c
c early time, use single region approx.
c
c set the concentration and flux at backfill-rock interface
c to zero
c
                mne0(it)=0.0
                wpe0(it)=0
                mne1(it)=0.0
                wpe1(it)=0
                mrec(it)=0.0
                wpsc(it)=0
                isb0(it)=0
                isb1(it)=0
                isbc(it)=0
                flux01(it)=x0*(a11*erf(dsqrt(e))+expe(-e)/dsqrt(pie*b)+1.0/r0)

```

```

        flux11(it)=0.0
        con1(it)=0.0
    else
c
c later time, use exact solution
c
        up=dsqrt(cut/b)
c
c calculate flux at waste surface with decay
c
        icode=1
        error=eps
        call intgr1(fl,0.0,low,up,error,sum,icode,n)
        if (n.gt.1e6) go to 1000
        mxe0(it)=0.0
        if (sum.gt.0.0) mxe0(it)=error/sum
        tr(it,2)=-2.0*e1*e2*sk21/pie*sum*x0
        flux01(it)=mxe0+(tr(it,2)+tr(it,1))*amps(-e)
        wsp0(it)=n
        isb0(it)=icode
c
c calculate flux at backfill-rock interface with decay
c
        icode=1
        error=eps
        call intgr1(hl,0.0,low,up,error,sum,icode,n)
        if (n.gt.1e6) go to 1000
        mxe1(it)=0.0
        if (sum.gt.0.0) mxe1(it)=error/sum
        tr(it,4)=-2.0*e1*e2*sk21/pie*sum*x1
        flux11(it)=mxe1+(tr(it,4)+tr(it,3))*amps(-e)
        wsp1(it)=n
        isb1(it)=icode
c
c calculate concentration at b-r interface with decay
c
        icode=1
        error=eps
        call intgr1(gl,0.0,low,up,error,sum,icode,n)
        if (n.gt.1e6) go to 1000
        msec(it)=0.0
        if (sum.gt.0.0) msec(it)=error/sum
        tr(it,6)=2.0*e1*e2*sk21/pie*sum*r0/r1
        con1(it)=msec+(tr(it,6)+tr(it,3))*amps(-e)
        wsep(it)=n
        isbc(it)=icode
c
c calculate mass transfer coefficient
c
        hm(it)=1e38
        if (con1(it).gt.0.0) hm(it)=flux11(it)/con1(it)*coeff
    end if
12 continue
    write (6,30)
    write (6,20) (time(i),i=1,nt)
    write (6,50) (flux01(i),i=1,nt)
    write (6,10) (mxe0(i),i=1,nt)
    write (6,40) (wsp0(i),i=1,nt)
    write (6,40) (isb0(i),i=1,nt)
    write (6,30) (flux11(i),i=1,nt)
    write (6,10) (mxe1(i),i=1,nt)
    write (6,40) (wsp1(i),i=1,nt)
    write (6,40) (isb1(i),i=1,nt)
    write (6,50) (con1(i),i=1,nt)
    write (6,10) (msec(i),i=1,nt)
    write (6,40) (wsep(i),i=1,nt)

```

```

      write (6,40) (iabc(i),i=1,nt)
      write (6,50) (hm(i),i=1,nt)
11  continue
2   continue
3   continue
4   continue
5   continue
6   continue
7   continue
8   continue
10  format (11(1p11.4))
15  format (///' half life=',1p13.3,' years'//)
20  format (11(1p11.4)/)
25  format (/ ' time, flux0, flux1, and concentration without decay
+          '/')
30  format ('          time (years) // flux (cm**3/sec) // max. error ',
+          '/// work space // cl/c2 // max. error // work space // hm'
+          ' (m/yr)')//)
40  format (11(16.3x))
50  format (/11(1p11.4))
80  format (///12('-----')//1x,'df,k1,k2,c1,c2,r0,n1=',
+7(1p10.3'///)
90  format (1h1,'eps,1bug,cut=',1p10.3,13,1p10.3)
100 format (/ 'd,c1,c2,c3=',4(1p13.6)/)
      go to 1100
1000 continue
      write (6,'(a)') 'Too many subintervals, fortran stop'
1100 continue
      stop
      end

```

c

```

-----
      subroutine integrl(f,omega,a,b,err,ans,ierr,n)
-----

```

```

c  integrl - calculate an integral
c
c
c  where
c
c      f=function of integrand
c      omega=frequency (0 if not sinusoidal function)
c      a,b=lower and upper limits
c      err=relative error bound
c      ans=result of integration
c      ierr=error flag (also served as output - no. of subintervals)
c      n=number of subdivisions used
c
c
c

```

c

```

-----

```

c

```

      implicit double precision (a-h,o-v)
      double precision f
      external f
      integer nw,nlw
      parameter (nw=4000)
      parameter (nlw=nw/8+2)
      double precision a,b,err,ans,omega,eps,om
      double precision aa,hb,abe,ernd,sum,error,work(nw),tmp,tmp1
      integer ierr,n,nn,iw(nlw)

```

c

```

      eps=err
      abe=0.0
      tmp=0.0
      aa=a
      if (omega.ne.0) then
        om=10.0*3.1415926535/omega

```

```

        bb=a+cm
        if (bb.gt.b) bb=b
    else
        bb=b
    end if
    nn=0
    ii=0
10  continue
    errd=eps
    icode=ierr
    call d01ajf(i,aa,bb,abe,errd,sum,error,werk,nw,lv,nlw,icode)
    if (icode.ne.0) then
c
c  not enough accuracy, reduce integral limits
c
        bb=(bb+aa)/2.0
        go to 10
    else
c
c  got it!
c
        nn=nn+1
        ii=ii+lv(1)
        if (tmp.eq.0) go to 20
        if (dabs(sum/tmp).le.eps) go to 30
20  continue
        tmp1=sum
        tmp=tmp+sum
        if (bb.eq.b) go to 30
        if (nn.gt.1e6) go to 30
        aa=bb
        if (ampe.ne.0.0) then
            bb=aa+cm
            if (bb.gt.b) bb=b
        else
            bb=b
        end if
        go to 10
    end if
30  continue
    ans=tmp
    n=nn
    ierr=ii
    err=error
    if (n.gt.1e6) err=-1e30
    return
end
c
c-----
function f(x)
c-----
    common /para/ d,c1,c2,c3,b,e,r1
    double precision x,d,c1,c2,c3,b,e,s,e,r1,ampe
    double precision f
    s=x**2
    c=x*d
    f=a/((c1*x*d*cos(c)+c2*d*sin(c))**2+(c3*x*d*sin(e))**2)*ampe*(-b*x**2)
    return
end
c
c-----
function fl(x)
c-----
    common /para/ d,c1,c2,c3,b,e,r1
    double precision x,d,c1,c2,c3,b,a,c,e,r1,expo
    double precision fl

```

```

a=x**2
c=x*d
f1=a/((c1*x**dexp(c)+c2*dain(c))**2+(c3*x*dain(c))**2)
+ /(1.0+b/a*x**2)*expo(-b*x**2)
return
end

c
-----
function g(x)
-----
common /para/ d,c1,c2,c3,b,a,c,e,r1,expo
double precision x,d,c1,c2,c3,b,a,c,e,r1,expo
double precision g
c=x*d
a=x*dain(c)
g=a/((c1*x**dexp(c)+c2*dain(c))**2+(c3*x*dain(c))**2)*expo(-b*x**2)
return
end

c
-----
function g1(x)
-----
common /para/ d,c1,c2,c3,b,a,c,e,r1,expo
double precision x,d,c1,c2,c3,b,a,c,e,r1,expo
double precision g1
c=x*d
a=x*dain(c)
g1=a/((c1*x**dexp(c)+c2*dain(c))**2+(c3*x*dain(c))**2)
+ /(1.0+b/a*x**2)*expo(-b*x**2)
return
end

c
-----
function h(x)
-----
common /para/ d,c1,c2,c3,b,a,c,e,r1,expo
double precision x,d,c1,c2,c3,b,a,c,e,r1,expo
double precision h
c=x*d
a=x*(x**dexp(c)-dain(c)/r1)
h=a/((c1*x**dexp(a)+c2*dain(a))**2+(c3*x*dain(a))**2)*expo(-b*x**2)
return
end

c
-----
function h1(x)
-----
common /para/ d,c1,c2,c3,b,a,c,e,r1,expo
double precision x,d,c1,c2,c3,b,a,c,e,r1,expo
double precision h1
c=x*d
a=x*(x**dexp(c)-dain(c)/r1)
h1=a/((c1*x**dexp(c)+c2*dain(c))**2+(c3*x*dain(c))**2)
+ /(1.0+b/a*x**2)*expo(-b*x**2)
return
end

c
-----
function expo(x)
-----
double precision expo,x
if (dabs(x).lt.87.0) expo=dexp(x)
if (x.gt.87.0) expo=1e38
if (x.lt.-87.0) expo=0.0
return
end

```

APPENDIX B: SAMPLE INPUT

1.05, 1, 1.
 1.03, 1, 1.
 1.03, 1, 1.
 .21, 1.
 .01, 1, 1.
 .66, 9, 1, 1.
 30, 1, 1.
 570, 1, 1.
 1.00, 10, 0.
 1.0-4, 0, 2.01

eps,ibug,cut= 1.000E-04 0 2.000E+01

df,k1,k2,e1,e2,r0,x1= 1.000E-05 1.000E+03 1.000E+03 2.000E-01 1.000E-02 6.590E+01 3.000E+01

time, flux0, flux1, and concentration without decay

1.0000E+00 1.0000E+01 1.0000E+02 1.0000E+03 1.0000E+04 1.0000E+05 1.0000E+06

1.1131E-01 3.6333E-02 1.2622E-02 4.6905E-03 2.7698E-04 1.5360E-04 1.2895E-04

0.0000E+00 0.0000E+00 1.2593E-06 3.1330E-04 2.4212E-04 1.5325E-04 1.2894E-04

0.0000E+00 0.0000E+00 2.7513E-04 3.4263E-01 9.4944E-01 9.7101E-01 9.7564E-01

half life= 5.73000E+03 years

time (years) // flux (cm**3/sec) // max. error // work space // c1/cs // max. error // work space // hm.

1.0000E+00 1.0000E+01 1.0000E+02 1.0000E+03 1.0000E+04 1.0000E+05 1.0000E+06

TIME

1.1133E-01 3.6374E-02 1.2754E-02 5.1398E-03 1.8761E-03 1.8599E-03 1.8599E-03

FLUX AT WASTE SURFACE

0.0000E+00 0.0000E+00 8.5721E-05 6.9278E-05 2.3199E-10 5.2957E-07 4.3443E-07

0 0 1 1 1 1 1

0 0 80 32 20 4 4

0.0000E+00 0.0000E+00 1.2456E-06 2.9638E-04 2.7579E-04 2.6657E-04 2.6657E-04

FLUX AT BACKFILL/ROCK INTERFACE

0.0000E+00 0.0000E+00 6.8951E-05 3.7774E-05 4.6261E-05 5.9398E-07 4.4230E-07

0 0 1 1 1 1 1

0 0 80 36 16 4 4

0.0000E+00 0.0000E+00 2.7345E-04 3.1899E-01 7.6581E-01 7.6852E-01 7.6852E-01

C1/CS

0.0000E+00 0.0000E+00 6.3083E-05 3.6337E-05 1.7500E-10 5.4578E-07 4.3643E-07

0 0 1 1 1 1 1

0 0 68 28 20 4 4

0.0000E+00 0.0000E+00 1.2430E-02 2.5353E-03 9.8269E-04 9.4649E-04 9.4649E-04

MASS TRANSFER DEFICIENT

FORTRAN STOP