DOE/PC/89774--T6

NOV 25 1992

DOE/PC/89774--T6

Effects of Catalytic Mineral Matter on CO/CO₂
Temperature and Burning Time for Char Combustion

DE93 007562

Prof. John P. Longwell
Prof. Adel F. Sarofim

Chun-Hyuk Lee
Prof. Changhe Chen

Quarterly Progress Report No. 12
July-September, 1992
Reference Grant # DE-FG22-89PC89774

Prepared for

Jeffrey C. Bogdan
U.S. Department of Energy
Pittsburgh Energy Technology Center
P.O. Box 10940, MS 921-165
Pittsburgh, Pennsylvania 15236-0940

by

Massachusetts Institute of Technology
Department of Chemical Engineering
Cambridge, Ma 02139

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.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Introduction

The high temperature oxidation of char is of interest in a number of applications in which coal must be burned in confined spaces. These include: the conversion of oil-fired boilers to coal using coal-water slurries, the development of a new generation of pulverized-coal-fired cyclone burners, the injection of coal into the tuyeres of blast furnaces, the use of coal as a fuel in direct-fired gas turbines in large-bore low-speed diesels, and entrained flow gasifiers. In addition, there is a need to better understand the temperature history of char particles in conventional pulverized-coal-fired boilers in order to better understand the processes governing the formation of pollutants and the transformation of mineral matter.

The temperature of a char particle burning in an oxygen containing atmosphere is the product of a strongly coupled balance between particle size and physical properties, heat transfer from the particle, surface reactivity, CO/CO₂ ratio and gas phase diffusion in the surrounding boundary layer and within the particle. In addition to its effects on burning rate, particle temperature has major effects on ash properties and mineral matter vaporization. Measurements of the temperature of individual burning char particles have become available in recent years and have clearly demonstrated large particle to particle temperature variations which depend strongly on particle size and on particle composition. These studies, done with pulverized coal, do not allow direct determination of the CO/CO₂ ratio produced at the char surface or the catalytic effects of mineral matter in the individual char particles and it has generally been assumed that CO is the only product of the carbon-oxygen reaction and that CO₂ is formed by subsequent gas phase reaction. More recent work, however, has pointed out the need to take CO₂ production into consideration in order to account for observed particle temperatures.

The importance of the CO/CO₂ ratio of carbon oxidation products is illustrated by examination of the heats of reaction for formation of these two products

The heat released by formation of CO_2 is a factor of 3.5 higher than for CO so the temperature of a particle will depend strongly on the CO/CO_2 ratio produced. If gas diffusion through the boundary layer is fast, increased direct production of CO_2 produces a higher temperature and a higher burning rate. If the supply of oxygen to the surface is limited by diffusion through the boundary layer, production of CO_2 consumes half as much carbon as production of CO so carbon consumption rate is reduced even though temperature may be somewhat higher.

Because ignition of char particles, which is dependent upon the CO_2/CO ratio, is an important clue for carbon oxidation kinetics, a few theoretical approaches to model a single particle oxidation have been made. But most of them assumed the infinitely thin reaction zone at the particle surface. These approaches can not explain pore diffusion limitation, structural change, or reaction at low temperatures inside the particle. There is a need to develop a complete model which combines both solid and gas phase reactions, heat and mass transfer. In this report, modeling results from SCOM (Single Carbon Oxidation Model) and an alternative approach for boundary gas phase reactions are reported.

Results

The calculation of SCOM (Single Carbon Oxidation Model) was performed on Cray-Y/MP of University of Nevada, Las Vegas Supercomputer center. The average CPU times consumed were around 130 minutes. The solutions were obtained during the entire carbon oxidation reaction. The reactions considered were $\text{CO} + \text{O} = \text{CO}_2$, $\text{CO} + \text{O}_2 = \text{CO}_2 + \text{O}$, and $2\text{O} = \text{O}_2$. The total summation of the errors (73 nodes and 4 species) was

$10^{-9} \sim 10^{-10}$ at low temperatures, but reached 10^{-3} at 2000 K. The summation of temperature errors was $10^{-6} \sim 10^{-8}$.

Two different oxygen concentrations were tested. The Fig.1 is a plot of temperature and conversion versus time when the oxygen partial pressure is 0.7 atm. The maximum temperature rise is approximately 2300K, and the duration time of ignition (high temperature plateau) is about 40 milli-seconds.

Fig.1. Temperature and conversion history of a spherocarb
($P_{O_2} = 0.7$ atm, $d_0 = 180 \mu\text{m}$)

The conversion rate, which is the slope of the conversion curve during the reaction, is almost constant, and the value is about 22 sec^{-1} . The significance of a constant value of

the conversion rate is that a carbon consumption rate is mainly dependent on the initial density and diameter of a particle.

$$r_c = -\frac{1}{A_{ex}} \frac{dm}{dt} = \frac{1}{A_{ex}} m_o \frac{dX}{dt} \equiv slope * \frac{\rho_o d_o}{6}$$

Although the external surface shrinks as the reaction goes on, the above relationship is approximately accurate up to a certain conversion. When the oxygen partial pressure is 0.8 atm, the shape of the ignition-extinction curve is almost identical to Fig.1. The maximum temperature rise is almost same, but the duration time of the ignition is about 2 ms longer. The conversion rate is about 22 sec^{-1} , and the particle conversion is close to one.

Fig.2. Temperature and mass fraction distribution (T=1300, 1400K)

Spatial distributions of temperature and mass fraction (oxygen partial pressure 0.8 atm) in the boundary layer are in Fig.2 and Fig.3. Oxygen radical concentration is too small to be plotted, and the other components are shown. A symbol of cross is given to represent temperatures, and symbols are for each component (solid-1300K, open-1400K surface temperature). Oxygen partial pressure is 0.8 atm. The ignition temperature is about 1350K, and the changes around that temperature are quite significant. In Fig.2, surface oxygen mass fraction drops from 0.6 to 0.3, and also CO and CO₂ almost doubles its concentration. But there's no notable particle diameter shrinkage.

Fig.3. Temperature and mass fraction distribution (T=1600, 2000K)

Results (solid-1600K, open-2000K surface temperature) from Fig.3 show that oxygen and nitrogen distributions are almost same although there is a little particle shrinkage ($r_0=0.009\text{cm}$). CO_2/CO ratio drop significantly as the temperature increases, which is opposite to experimental results reported earlier. The cause of this behavior is mainly due to the lack of gas phase CO oxidation. Therefore, CO_2/CO ratio are directly related to CO_2 and CO production of the heterogeneous reaction, which generates less CO_2 as the temperature goes up. To get CO oxidation in the gas phase boundary layer, we may need to incorporate more elementary reactions. But the computation time currently used for one ignition-extinction calculation is about 2 Cray CPU hours, which is very long, and will be longer if we incorporate full reaction kinetics. Actually we don't need temperature and mass fraction distribution through whole time span, there's a need to develop a gas phase model which calculates distributions for only one surface temperature. A model incorporating full reaction set is under development using the time dependent results of SCOM as an initial guess.

Fig.4 Schematic diagram of full gas phase reaction simulation in the boundary layer

Future Program

A full gas phase reaction model will be completed. And analysis based on the results from both SCOM and gas phase model will be made and the effects of important parameters will be studied. Professor Chen has recently joined the project and has undertaken to continue the experimental measurements of CO_2/CO ratio using the electrodynamic balance.

Appendix

```
C=====
c  This program calculates and concentration & temperature
c  distribution using heat balance equation in the boundary layer.
c  This program requires at least CHEMKIN II 2.6 & TRANFIT 1.9 for
c  calculating thermodynamic and multicomponent transport data.
C-----
c  made by Chun-hyuk Lee ,MIT, Oct, 92
C=====

      implicit real (a-h,o-z)

c
c  parameters for chemkin
c
      PARAMETER (LINKCK=25, LINKTP=35, LTRAN=31, LOUT=6, KDIM=15, NO=4,
1      LENICK=2000, LENRCK=2000, LENCCK=100,
2      LENIMC=100, LENRMC=6000)
C-----
c  LENIMC=4*KK+NLITE
c  LENRMC=(19+2*NO+NO*NLITE)*KK+(15+NO)*KK**2
c  LENICK, LENRCK, LENCCK: from interpreter output
c  NO: order of polynomial fit, default=4
c  NLITE: # of species of molecular weight < 5
c  KDIM: >= KK number of species
C-----
      common/mat1/a(200,3),b(200),t(200),x(200),yik(15,200),det,nm(2)
      common/mat2/to
      common/mat3/iband
      common/mat4/axis,xfact
      common/mat5/nxx,nxel,nntol,nodtol,nrhs,ncomp
      common/dat1/conv,eff,rc,xco,d,qcon
c      common/dat2/dtold
      common/dat3/rkg,po2
      common/dat4/rho(200),vr(200),vi(15,200),ri(15,200),cpg(200),
#      cpi(15,200),rhamda(200),hi(15,200),xi(15),yi(15)
      common/dat5/viy(15,200),vit(15,200),vc(200)
C-----
      common/par1/ICKWRK(2000),RCKWRK(2000)
      common/par2/KK,P,RU,RUC,WT(15),WDOT(15),HMS(15),
#      CPBMS,CPMS(15),RHOG,xik(15,200)
      common/tdat1/IMCWRK(100),RMCWRK(6000)
      common/tdat2/EPS(15),SIG(15),DIP(15),POL(15),ZROT(15),
1      NLIN(15),DTH(15),COND,DM(15),Dkm(15,15)
C
      LOGICAL IERR
      CHARACTER CCKWRK(100)*16, KSYM(15)*16
C-----
C
c  *****open CHEMKIN & TRANSPORT link files
      OPEN (LINKCK,STATUS='OLD', FORM='UNFORMATTED', FILE='cklink')
      OPEN (LINKTP,STATUS='OLD', FORM='UNFORMATTED', FILE='tplink')
c
c  ***** open output & solution files *****
```

```

c
  open (3,file='g.out')
c   open (4,status='old',file='iguess')
c-----
c   iguess: initial guess file
c   g.out: mesh, T, Yik for KG plot
c-----
C
C   INITIALIZE CHEMKIN
C
  CALL CKINIT(LENICK, LENRCK, LENCCK, LINKCK, LOUT, ICKWRK,
1    RCKWRK, CCKWRK)
  CALL CKINDX(ICKWRK, RCKWRK, MM, KK, II, NFIT)
  IF (KK .GT. 15) THEN
    WRITE (LOUT, 1000) 15
    STOP
1000  format(/,1x,'15 needs to be increased! at least',/)
  ENDIF
C
  CALL CKSYMS(CCKWRK, LOUT, KSYM, IERR)
  CALL CKWT(ICKWRK, RCKWRK, WT)
  CALL CKRP(ICKWRK, RCKWRK, RU, RUC, PATM)
  P = PATM
c
c   Initialize Transport
c
  call MCINIT(LINKTP,LOUT,LENIMC,LENRMC,IMCWRK,RMCWRK)
  call MCPRAM(IMCWRK,RMCWRK,EPS,SIG,DIP,POL,ZROT,NLIN)
c
c   initialization of data for solid & gas subroutines
c-----gas phase ncomp(max)=15 nxe1(max)=199
  nxe1=72
  nrhs=1
  ncomp=KK
  axis=1.0
c   xfact=2.35
  xfact=2.5
  to=298.15
  po2=0.8
c
c   ***** initialize temperature and mass fractions for the gas phase *****
c
  call init
c
c   call gas subroutine for temperature and mass fraction distribution
c
  temp=t(1)
  call gas(temp)

c
c   print solutions
c
c-----
  write(3,1008) temp

```

```

do 95 i=1,nodtol
  write(3,1005) x(i),t(i),yik(1,i),yik(2,i),yik(3,i),yik(5,i)
95 continue

```

```

1005 format(1x,f11.6,1x,f10.2,4(1x,f11.6,1x))
1008 format(/,1x,'Tp= ',f12.4,1x)
  stop
  end

```

```

c
c-----Initialize T(i) & Yik(ic,i)
c
  subroutine init
  implicit real (a-h,o-z)
  common/mat1/a(200,3),b(200),t(200),x(200),yik(15,200),det,nm(2)
  common/mat2/to
  common/mat5/nnx,nxel,nntol,nodtol,nrhs,ncomp
  open (4,status='old',file='iguess')
  nnx=nnx+1
  yio=1.0e-12
c
c ***** initial condition for species ***
c
  do 705 ic=1,ncomp
    do 705 kk=1,nnx
      yik(ic,kk)=yio
705 continue
  do 706 i=1,nnx
    read (4,1010) x(i),t(i),yik(1,i),yik(2,i),yik(3,i),yik(5,i)
706 continue
1010 format(1x,f11.6,1x,f10.2,4(1x,f11.6,1x))
  return
  end

```

```

c-----
c   This sub-program calculates the temperature and concentration
c   distribution in the boundary layer.
c   - Galerkin FEM is used in this modeling.
c   * linear basis function is used.
c   - For the thermodynamic and transport data, the package from
c   Sandia National Lab has been used
c   MADE BY Chun-hyuk Lee , Oct 1992
c-----

```

```

SUBROUTINE gas(tp)
  implicit real (a-h,o-z)
  common/mat1/a(200,3),b(200),t(200),x(200),yik(15,200),det,nm(2)
  common/mat2/to
  common/mat3/iband
  common/mat4/axis,xfact
  common/mat5/nnx,nxel,nntol,nodtol,nrhs,ncomp
  common/dat1/conv,eff,rc,fco,r,qcon
c   common/dat2/dtold

```

```

common/dat3/rkg,po2
common/dat4/rho(200),vr(200),vi(15,200),ri(15,200),cpg(200),
#   cpi(15,200),rhamda(200),hi(15,200),xi(15),yi(15)
common/dat5/viy(15,200),vit(15,200),vc(200)
c-----
common/par1/ICKWRK(2000),RCKWRK(2000)
common/par2/KK,P,RU,RUC,WT(15),WDOT(15),HMS(15),
#   CPBMS,CPMS(15),RHOG,xik(15,200)
common/tdat1/IMCWRK(100),RMCWRK(6000)
common/tdat2/EPS(15),SIG(15),DIP(15),POL(15),ZROT(15),
1   NLIN(15),DTH(15),COND,DM(15),Dkm(15,15)
c
c-----set dimension for local variables
c   ncomp(max)=15, nxel(max)=199
c-----
dimension ttmp(200),wtm(200)
dimension xa(3),ya(3)
c
c-----set iteration conditions
error=1.0e-9
niter=15
evaluate=1.0e-12
c
c   ***** calculate various parameters *****
c
nnx=nxel+1
nuc=nnx+1
nlc=nnx+1
iband=3
noddol=nnx
nntol=nxel
c
c   ***** iteration loop *****
c
ipass=0
jpass=0
summax=0.0
iter=0
300  continue
iter=iter+1
write(6,2006) iter
c
c   ***** form the matrix a and the vector b *****
c   ***** iterate over components *****
c
sum1=0.0
sum2=0.0
c
c   ---- call CHEMKIN & TRANSPORT subroutines ----
c
do 77 ir=1,nnx
do 771 ic=1,ncomp
yi(ic)=yik(ic,ir)
771  continue

```

```

      call CKYTX(yi,ICKWRK,RCKWRK,xi)
      do 772 ic=1,ncomp
        xik(ic,ir)=xi(ic)
772      continue

77      continue
c
c
      do 88 ir=1,nnx
c-----
        do 881 ic=1,ncomp
          yi(ic)=yik(ic,ir)
881      continue
        call CKYTX(yi,ICKWRK,RCKWRK,xi)
c-----
        call CKMMWY(yi,ICKWRK,RCKWRK,wtmm)
        wtm(ir)=wtmm
c-----
        call CKRHOY(P,T(ir),yi,ICKWRK,RCKWRK,RHOG)
        rho(ir)=RHOG
c      vr(ir)=x(1)**2*rc/(x(ir)**2*rho(ir))
c      vr(ir)=rc*fco*RU*T(ir)/(24.0*P*RHOG*x(ir)**2)*rho(1)*x(1)**2
        vr(ir)=0.0
        call CKCPBS(T(ir),yi,ICKWRK,RCKWRK,CPBMS)
        cpg(ir)=CPBMS
        call CKWYP(P,T(ir),yi,ICKWRK,RCKWRK,WDOT)
        call CKCPMS(T(ir),ICKWRK,RCKWRK,CPMS)
        call CKHMS(T(ir),ICKWRK,RCKWRK,HMS)
        do 883 ic=1,ncomp
          ri(ic,ir)=WT(ic)*WDOT(ic)
          cpi(ic,ir)=CPMS(ic)
          hi(ic,ir)=HMS(ic)
883      continue
        call MCMCDT(P,T(ir),xi,IMCWRK,RMCWRK,ICKWRK,RCKWRK,DTH,COND)
        rhamda(ir)=COND
        call MCADIF(P,T(ir),xi,RMCWRK,DM)
c
c      calculate dT/dr
c
        delx=1.0e-5
        if (ir.eq.1) then
          xa(1)=x(1)
          xa(2)=x(2)
          xa(3)=x(3)
          ya(1)=t(1)
          ya(2)=t(2)
          ya(3)=t(3)
          xin=xa(1)+delx
          call POLINT(xa,ya,3,xin,yout,dy)
          delt=yout-ya(1)
        elseif (ir.eq.nnx) then
          xa(1)=x(nnx-2)
          xa(2)=x(nnx-1)
          xa(3)=x(nnx)

```

```

ya(1)=t(nnx-2)
ya(2)=t(nnx-1)
ya(3)=t(nnx)
xin=xa(3)-delx
call POLINT(xa,ya,3,xin,yout,dy)
delt=ya(3)-yout
else
  xa(1)=x(ir-1)
  xa(2)=x(ir)
  xa(3)=x(ir+1)
  ya(1)=t(ir-1)
  ya(2)=t(ir)
  ya(3)=t(ir+1)
  xin=xa(2)+delx
  call POLINT(XA,YA,3,xin,yout,dy)
  delt=yout-ya(2)
endif

sumvi=0.0
do 884 ic=1,ncomp
  if (xi(ic).lt.evalue) then
    xi(ic)=evalue
  endif
  if (yi(ic).lt.evalue) then
    yi(ic)=evalue
  endif
c
c   calculate dYi/dr
c
  if (ir.eq.1) then
    xa(1)=x(1)
    xa(2)=x(2)
    xa(3)=x(3)
    ya(1)=xik(ic,1)
    ya(2)=xik(ic,2)
    ya(3)=xik(ic,3)
    xin=xa(1)+delx
    call POLINT(xa,ya,3,xin,yout,dy)
    delxik=yout-ya(1)
  elseif (ir.eq.nnx) then
    xa(1)=x(nnx-2)
    xa(2)=x(nnx-1)
    xa(3)=x(nnx)
    ya(1)=xik(ic,nnx-2)
    ya(2)=xik(ic,nnx-1)
    ya(3)=xik(ic,nnx)
    xin=xa(3)-delx
    call POLINT(xa,ya,3,xin,yout,dy)
    delxik=ya(3)-yout
  else
    xa(1)=x(ir-1)
    xa(2)=x(ir)
    xa(3)=x(ir+1)
    ya(1)=xik(ic,ir-1)

```

```

        ya(2)=xik(ic,ir)
        ya(3)=xik(ic,ir+1)
        xin=xa(2)+delx
        call POLINT(XA,YA,3,xin,yout,dy)
        delxik=yout-ya(2)
    endif
c---- Set minimum value of delxik
c    if (delxik.lt.0.0.and.ic.eq.1) then
c        if (delxik.le.0.0.and.ic.eq.1) then
c            delxik=evalue
c        elseif (delxik.eq.0.0.and.ic.eq.1) then
c            delxik=evalue
c            if (tp.ge.1600.) then
c                sumxik=0.0
c                do 8841 jc=2,ncomp
c                    sumxik=sumxik+xik(jc,ir+1)-xik(jc,ir)
c 8841            continue
c                delxik=-sumxik
c                if (delxik.lt.evalue) then
c                    delxik=evalue
c                endif
c            endif
c            elseif (delxik.ge.0.0.and.ic.eq.2) then
c                delxik=-evalue
c            elseif (delxik.ge.0.0.and.ic.eq.3) then
c                delxik=-evalue
c            endif
c----End of delxik
        vi(ic,ir)=-DM(ic)*delxik/(delx*xi(ic))-DTH(ic)*delt/(delx
        #          *rho(ir)*yi(ic)*t(ir))*0.0
        vit(ic,ir)=DTH(ic)*delt/(delx*t(ir))*0.0
        viy(ic,ir)=rho(ir)*DM(ic)
c----Calculation of Vc
        if (ic.ne.ncomp) then
            sumvi=sumvi-yi(ic)*vi(ic,ir)
        else
            go to 884
        endif
        884 continue
c----Correcting Vi
        vinew=sumvi/yi(ncomp)
        vc(ir)=vinew-vi(ncomp,ir)
        vi(ncomp,ir)=vinew
c        vc(ir)=sumvi
c        vc(ir)=0.0
c        do 885 ic=1,ncomp
c            vi(ic,ir)=vi(ic,ir)+vc(ir)
c 885 continue
cc----End of correcting
        if (vi(1,1).gt.1.0e-3) then
            write(6,*) ' Vi O2 is positive ! Vi= ',vi(1,1)
            stop
        endif
        88 continue

```

```

C-----
c   write(6,*) 'rho= ',rho(1),' vr= ',vr(1)
C-----
      if (iter.eq.1) then
        fco=1./(1.+0.02*po2**0.21*exp(6000./(1.987*tp)))
C-----
        pso2=rho(1)*82.056*tp*yik(1,1)/WT(1)
        delp=po2-pso2
        if (delp.lt.1.0e-12) delp=1.0e-12
        vi11=vi(1,1)
c       vi11=-1.*(yik(2,1)*vi(2,1)+yik(3,1)*vi(3,1))/yik(1,1)
        if (abs(vr(1)).gt.abs(vi11)) then
          vio2=evaluate
          write(6,*) ' vr is bigger than Vi O2 '
          write(6,*) 'vr= ',vr(1),' Vi O2= ',vi(1,1)
          endif
          vio2=-1.*(vi11+vr(1))
c       vi12=abs(vi(1,1))
          if (vio2.lt.1.0e-1.and.tp.gt.1700.) vio2=1.0e-1
          rkg=rho(1)*yik(1,1)*vio2*12.011*(1.+fco)/(31.9988*delp)
c       if (rkg.lt.1.0e-10) rkg=1.0e-10
C-----
        rc=rkg*po2
        endif

c
c       ***** solve Energy equation *****
c
c---if temperatue converged already iterate Yik only
c
c       if (jpass.eq.1) go to 411
C-----
      call domit
      call boundt(tp)
      call band
      do 401 i=1,nodtol
        err=t(i)-b(i)
        ttmp(i)=b(i)
        sum2=sum2+sqrt(err**2)
401   continue
411   continue

c
c       ***** solve Species Conservation equations *****
c
c       do 99 ic=1,ncomp
c       do 99 ic=2,ncomp
c       do 99 ic=1,ncomp-1

c
c       call domi(ic)

c
c       ***** insert boundary conditions *****
c
c       call bound(ic)

c
c       ***** solve the system of linear equations *****
c

```



```

c
  call band
  do 400 i=1,nodtol
    err=yik(ic,i)-b(i)
    yik(ic,i)=b(i)
    if (yik(ic,i).lt.evalue) yik(ic,i)=evalue
c-----
c   write(6,*) 'yik(',ic,',',i,')= ',yik(ic,i)
c-----
    sum1=sum1+sqrt(err**2)
  400 continue
  99 continue
c-----Calculate yn2
  do 408 ir =1,nnx
    ysum=0.0
    do 409 ic=1,ncomp-1
      ysum=ysum+yik(ic,ir)
  409 continue
    yik(ncomp,ir)=1 0-ysum
    if (yik(ncomp,ir).lt.evalue) then
      yik(ncomp,ir)=evalue
    do 419 ic=1,ncomp-1
      yik(ic,ir)=yik(ic,ir)/ysum-evalue
  419 continue
    endif
  408 continue
c-----End of yn2
  do 402 i=1,nodtol
    t(i)=ttmp(i)
c-----
c   write(6,*) 't(',i,')= ', t(i)
c-----
  402 continue
c
c   ***** check convergence *****
c
  if(ipass.eq.1) go to 600
  esum=sum1+sum2
  write(6,2007)sum1,sum2
  if(esum.le.error) go to 500
  summax=max(esum,summax)
  if(sum2.lt.1.0e-9) jpass=1
  if(iter.lt.niter) go to 300

c
c   ---- set condition if the iteration diverges ----
c   ---- get back to initial guess & solve again ----
c
  if(summax.eq.esum) then
    write(6,2010)
    go to 600
c   stop
c   go to 300
  endif
  write(6,2008)
  go to 600

```

```

500  continue
      write(6,2009)
600  continue
c
c      ***** format *****
c
2006  format(/,1x,'iteration number 'iter) =',i5,/)
2007  format(/,1x,'Yik error =',e15.8,2x,'Temp error =',e15.8,/)
2008  format(/,1x,'!!-- iteration did not converge ---',/,
      #      1x,' -- but decreasing error!!',/)
2009  format(/,1x,'!--- iteration converged -----',/)
2010  format(/,1x,'WARNING! iteration diverges !!!',/)
      return
      end
c
c
      SUBROUTINE POLINT(XA,YA,N,X,Y,DY)
      PARAMETER (NMAX=10)
      DIMENSION XA(N),YA(N),C(NMAX),D(NMAX)
      NS=1
      DIF=ABS(X-XA(1))
      DO 11 I=1,N
          DIFT=ABS(X-XA(I))
          IF (DIFT.LT.DIF) THEN
              NS=I
              DIF=DIFT
          ENDIF
          C(I)=YA(I)
          D(I)=YA(I)
11  CONTINUE
      Y=YA(NS)
      NS=NS-1
      DO 13 M=1,N-1
          DO 12 I=1,N-M
              HO=XA(I)-X
              HP=XA(I+M)-X
              W=C(I+1)-D(I)
              DEN=HO-HP
              IF(DEN.EQ.0.)PAUSE
              DEN=W/DEN
              D(I)=HP*DEN
              C(I)=HO*DEN
12  CONTINUE
          IF (2*NS.LT.N-M)THEN
              DY=C(NS+1)
          ELSE
              DY=D(NS)
              NS=NS-1
          ENDIF
          Y=Y+DY
13  CONTINUE
      RETURN
      END
***** Other subroutines are the same ones in Report #11 *****

```

END

**DATE
FILMED**

3 / 24 / 93