Causes of Calculation Instabilities and Their Countermeasures for KIVA, a Three-Dimensional Program for Combustion and Fluid Flow in I. C. Engines

Brent CRARY, Kazushige KIKUTA, Takemi CHIKAHISA
and Tadashi MURAYAMA

(Received August 31, 1992)

Abstract

This paper presents modifications made to the KIVA simulation code to prevent instabilities which cause termination of calculations. These instabilities are mainly caused by Eulerian (timestep) controlled calculations. Alternative calculation methods based on equilibrium concepts, as opposed to timestep control, are employed without adverse effect on computation time. The paper discusses methods for the prevention of the " Temperature Overflow " error. The alternative methods control the evaporation rate and amount of heat transfer of the fuel droplet, and also prevent overflows in limits of the internal energy of cells from calculations of mass and enthalpy transfer. Other modifications such as control of the rate of chemical reaction and determination of the droplet locations in the grid space are also briefly discussed.

1. Introduction

KIVA is a program used widely for 3-D numerical simulation of combustion and fluid flow in internal combustion engines. However, there are still many problems with instability, and calculation can not always be completed for various conditions. For example, when the estimation of mass and enthalpy transport are not appropriately calculated, the internal energy of cells becomes too high, causing a " Temperature Overflow " error, which terminates the calculation. Instability could be removed by decreasing the calculation timestep to be very small, but this increases calculation time significantly, so alternate methods are necessary.

This paper presents some of the causes of instability and the ways in which the program was modified to correct the problems.

2. Causes of Instability and Their Countermeasures

There are many causes of instability which are mainly related to the calculations of evaporation and heat transfer of fuel droplets and the diffusion of mass and energy. Depending on the case, the cell internal energy or mass density becomes negative, resulting in " Temperature Overflow " error.

Derpartment of Mechnical Engineering, Faculty of Engineering, Hokkaido University.

The following outlines the causes of instability and their countermeasures. A summary is shown in Table 1 at the end of the report.

2. 1 Problems and Solutions Related to the Fuel Droplet

2. 1. 1 Control of heat transfer between the cell and the fuel droplet.

In KIVA, the amount of heat transfer between the cell and the fuel droplet is calculated using the Euler method, so when the heat capacity of the cell is very small compared to the droplet, the cell internal energy changes significantly for a very small error of heat transfer amount, resulting in a calculation overflow error.

To solve this problem, the equilibrium temperature of the cell and the droplet is first calculated from the total amount of heat transfer, and the droplet temperature (or the cell) is controlled not to exceed this temperature. The concept of the equilibrium temperature is as shown in Fig. 1, for the case when the cell temperature is higher than the droplet temperature.

The equilibrium temperature is calculated by the following method: First, the changes in cell and droplet temperature are as follows,

$$\Delta T_g = \frac{-dQ}{C_g}, \text{ and } \Delta T_d = \frac{dQ}{C_d} \tag{1}$$

where Q is the amount of heat transfer, C is the heat capacity, and the subscripts "g" and "d" represent the cell and droplet, respectively. From Fig. 1, the equilibrium temperature, Teq, is:

$$T_{eq} = (T_g + \Delta T_g) = (T_d + \Delta T_d) \tag{2}$$

Substituting (ΔT_g) and (ΔT_d) of Eq.(1) into (2), and solving for (dQ), the amount of heat transfer is obtained:

$$\therefore\ dQ = \frac{T_g - T_d}{(1/C_g + 1/C_d)} \tag{3}$$

The equation for (dQ) is then substituted back into equation (2) to obtain the equilibrium temperature of the drop (or cell):

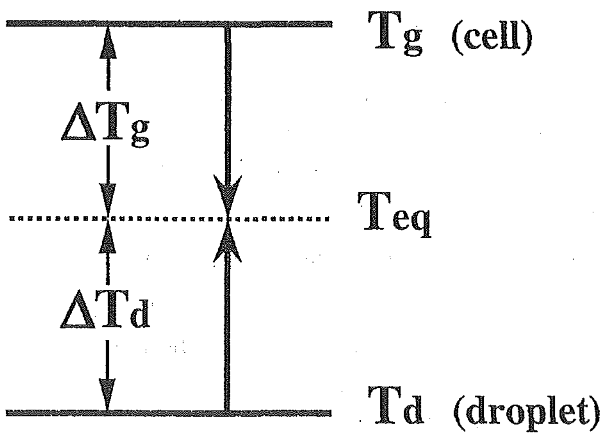

Fig. 1 Concept of equilibrium temperature between cell and droplet

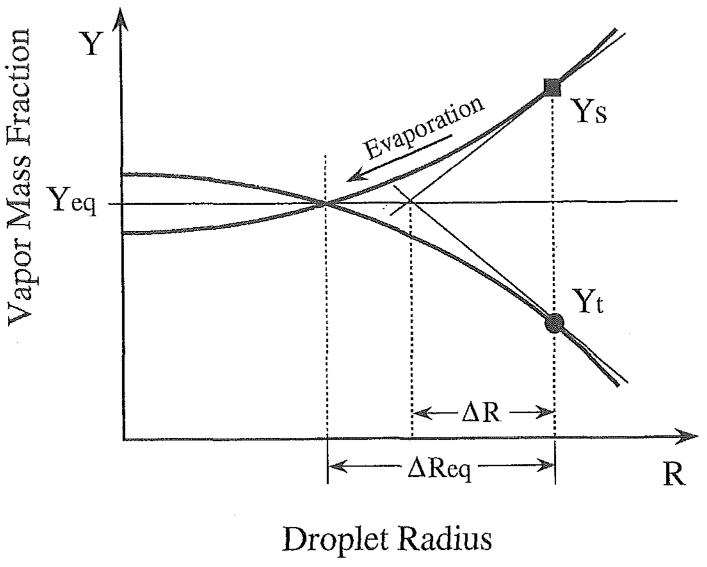

Fig. 2 Concept of evaporation control

$$\therefore\ T_{eq}=(T_d+\Delta T_d)=(T_d+dQ/C_d) \tag{4}$$

The equilibrium temperature is used to limit the temperature change of the drop (and cell). The algorithms for this temperature control and the equilibrium temperature are in the EVAP subroutine, in Appendix A.

2. 1. 2 Control of fuel droplet evaporation

When the cell temperature is high, the fuel droplet evaporates too fast in one calculation cycle, and in the next cycle the fuel vapor pressure in the cell is higher than the vapor pressure of the droplet surface, resulting in calculation instability because the program can not handle condensation conditions. Another problem is that fuel vapor in the cell suddenly increases creating a strong flow of gas to the surroundings, which causes the calculation of the Courant number to exceed the stability region. The sources of this error are mainly from the calculation for the amount of evaporation, which is dependent on the Euler method, and also on the inappropriate handling of the droplet at flash boiling condition.

As a countermeasure for this, the amount of droplet evaporation is determined from the calculation of the equilibrium state between the droplet surface pressure and the cell fuel vapor pressure; and the evaporation rate was controlled to not exceed this evaporation amount. Fig. 2 is a diagram of this concept. The abscissa is the change in droplet radius from evaporation, and the ordinate is the mass fraction of fuel vapor. Y_s is the vapor mass fraction of the droplet surface, and Y_t is the mass fraction of the cell. As the droplet radius decreases, cell vapor pressure increases and the droplet vapor pressure decreases due to the temperature drop by evaporation.

The change in droplet radius ΔR, which roughly corresponds to the equilibrium state, is calculated from the gradient of each mass fraction, so ΔR was used as the limit of the radius change for calculation safety. The method of this estimate is summarized in the following equations.

From Fig. 2, the calculation of the equilibrium vapor mass fraction Y_{eq} is as follows.

$$Y_{eq}=Y_s+\frac{dY_s}{dR}\Delta R=Y_t+\frac{dY_t}{dR}\Delta R \tag{5}$$

The amount of droplet evaporation ΔR is then calculated as

$$\therefore\ \Delta R=\frac{Y_s-Y_t}{(dY_t/dR)-(dY_s/dR)} \tag{6}$$

The derivatives of Y_s and Y_t in the denominator of Eq.(6) are calculated as shown below in Eq.(7). Here, P_v is the vapor pressure of the droplet surface and T is the temperature of the droplet.

$$\left(\frac{dY_s}{dR}=\frac{dY_s}{dP_v}\cdot\frac{dP_v}{dT}\cdot\frac{dT}{dR}\right),\ \text{and}\ \left(\frac{dY_t}{dR}=\frac{\Delta Y_t}{\Delta R}\right) \tag{7}$$

The actual program changes are included in the EVAP subroutine in Appendix A.

2. 2 Problems and Solutions for Mass and Energy Diffusion

Depending on the calculation condition, the mass and enthalpy transfer between cells is not correct, causing the cell internal energy to become negative or extremely high, resulting in "Temperature Overflow." To solve this problem, two modifications were added to correct errors in mass and energy transfer.

2. 2. 1 Switching of the Differentiation Model of Convection

In KIVA, two models are optionally used for solving the amount of transport of mass and enthalpy by convection. One model is the Full Donor Cell model (FDC), a method which solves the transport amount as being uniform in the cell. Calculations with FDC are the most stable, but a weakness is that diffusion is too strong. The other method is the Interpolated Donor Cell model (IDC), a method which solves the transport amount by applying a weighting corresponding to the fluid speed to the information of both the object cell and the cell in the transport direction. A characteristic of the IDC model is that although the diffusion speed is suppressed, calculation stability is not good in comparison to FDC, especially when there are large differences in gradients between cells.

Based on this, the characteristics of each model are used for the modification. The IDC model is used as a rule, but when internal energy is extremely high or negative, or the chemical species goes negative in a cell, the model is switched to the more stable FDC method for one calculation cycle for that cell. The model is then switched back to IDC after one calculation cycle.

2. 2. 2 Application of Robin Hood algorithm to cell internal energy

The Robin Hood algorithm (RH), is a method already used in KIVA for preventing chemical species mass from becoming negative in a cell. To prevent the overflow of internal energy, the RH method was also applied not only to mass diffusion, but also to energy diffusion. That is, if there is a "trouble cell" with internal energy that exceeded the upper or lower limit, the excess or insufficient energy is either borrowed or donated by the six surrounding cells. The upper and lower limits of internal energy are the values defined in KIVA. The RH algorithm is in the STATE subroutine in Appendix B.

The calculations of the algorithm are as follows. First, if there is a " trouble cell " which has for example, negative internal energy, then energy is donated by the six surrounding cells to make the internal energy of the trouble cell equal to the the lower limit. Only the cells with energy greater than the minimum value are available for donating energy. The amount of energy that each cell donates to the trouble cell is weighted so that cells with higher energy donate more energy. Thus, the amount to be donated to the trouble cell by each surrounding cell with internal energy greater than the minimum value is:

$$(E_{don})_i=\frac{E_i-E_{min}}{\sum_{i=1}^{6}(E_i-E_{min})}\bullet(E_{min}-E_{tr}) \tag{8}$$

Here, Ei is the internal energy of each of the six surrounding cells, Emin is the lower limit value, and Etr is internal energy of the trouble cell.

Likewise, the amount of energy to be received *(Erec)* by each cell from the trouble cell *(Etr)* that has exceeded the upper limit *(Emax)*, is:

$$(E_{rec})_i=\frac{E_{max}-E_i}{\sum_{i=1}^{6}(E_{min}-E_i)}\bullet(E_{tr}-E_{max}) \tag{9}$$

2. 3 Other Problems and Solutions.

There is a problem with chemical reaction rate being too fast, and the chemical species becomes negative, resulting in " Temperature Overflow." To solve this, the upper limit of reaction speed for chemical species to not go negative is determined, and the chemical reaction speed is controlled to not exceed that upper limit. The algorithm for this method is in the CHEM subroutine in Appendix C.

Besides this, there is an error with not being able to find the cell which contains a fuel droplet. In this case, the fuel droplet coordinate is moved a small amount and the calculation is redone. There were also errors which occurred with the mass of a substance other than the fuel becoming zero, causing the evaporation calculation to have division by zero. For this, the equations which caused division by zero is not used, and the necessary values for the calculations which followed were fixed.

3. Summary

In this paper, the origins of the calculation instabilities and their solutions are presented. In general, these solutions prevent overshooting arising from Eulerian calculation methods by setting limit values based on equilibrium concepts. From these modifications, total calculation finishes in normal time.

Table 1 Causes of Instability and Their Countermeasures

TYPE OF ERROR	ORIGIN OF ERROR	PROGRAM MODIFICATION
Cell temperature exceeds overflow limit in evaporation.	Heat capacity of the cell is small compared to that of droplet. (EVAP Subroutine)	The amount of heat transfer is limited such that the cell temperature can not exceed the equilibrium temperature of the cell and droplet.
Condensation occurs in next calculation cycle after evaporation.	Evaporation of the fuel droplet is too fast. (EVAP Subroutine)	The rate of droplet evaporation is limited by the equilibrium mass vapor fraction of the droplet surface and the cell.
Internal energy of cell exceeds overflow limit.	The transfer of mass and energy are not calculated correctly with IDO model. (DIFF Subroutine)	a) The diffusion model is switched to FDC for the related cells. b) Energy is borrowed from or donated to neighboring cells.
Species mass becomes negative from chemical reaction.	Chemical reaction rate is too fast for the calculation timestep. (CHEM Subroutine)	Chemical reaction rate is limited by the reaction rate of the most unstable species.

APPENDIX A: EVAP SUBROUTINE CHANGES

The following are the modifications made to the EVAP subroutine. Line numbers at the right correspond to the original program listing in the KIVA Manual. ***Changes to the program are represented by bold, oblique print.*** *Omitted parts of the program are represented by "• •".*

```
      SUBROUTINE EVAP                                                line 1952
                         •   •
                         •   •
      DO 90 N=1,NP                                                   line 1972
      IFLAGT=0                                                (statement added)
      I4=I4P(N)                                                      line 1973
                         •   •
                         •   •
      CVLIQ=0.01*(ELIQ(ITB+2)-ELIQ(ITB+1))
C     IF(RADSQ.EQ.0.) GO TO 80                     (comment mark)   line 2030
      AIRLAM=AIRLA1*TG*SQTG/(TG+AIRLA2)
                         •   •
                         •   •
 75   RSUM=RDROP+RADPP(N)                          ("75" added)     line 2045
      XX1=SIXTH*RSUM*CVLIQ
      XX2=AIRLAM*ANUSSP*DT/(RSUM*RHOP)
C     TP(N)=(HLAT*(RADPP(N)-RDROP)+TTIL*XX2+TDROP*XX1)/(XX1+XX2)  (comment )
      TP(N)=(TTIL*XX2 + TDROP*XX1)/(XX1+XX2)                   (new equation)
                         •   •
                         •   •
      DMASS=PARTRM*(RADPP3-RDROP3)                                   line 2055
C-----------------------------------------------------------------
C     AVOIDING TOO FAST TP INCREASE BY
C           CONTROL OF HEAT TRANSFER BETWEEN CELL AND DROPLET
C-----------------------------------------------------------------
      HCAPI4=CPC(I4)*TOTCM(I4)
      HCAPP=PARTRM*RDROP3*CVLIQ
      DQP=(TTIL-TDROP)/(1.0/HCAPI4+1.0/HCAPP)
      TPEQ=TDROP+DQP/HCAPP
      IF(TTIL.GE.TDROP) THEN
         IF(TP(N).GT.TPEQ) TP(N)=TPEQ
      ELSE
         IF(TP(N).LT.TPEQ) TP(N)=TPEQ
      ENDIF
      TP(N) = TP(N) + (HLAT*(RADPP(N)-RDROP)/(XX1+XX2))
C-----------------------------------------------------------
C     AVOIDING OVER EVAPORATION OF DROPLET,
C        CONTROL CHANGE IN RADIUS BY EQUILIBRIUM VAPOR
C        PRESSURE BETWEEN CELL AND DROPLET
C-----------------------------------------------------------
      IF(IFLAGT.EQ.1) GO TO 85
      VAPMT=VAPM(I4)-DMASS
      TOTCMT=TOTCM(I4)-DMASS
      YTNT=VAPMT/TOTCMT
      IF(Y1TIL.GE.Y1STAR) GO TO 85
      IF(YTNT.LT.Y1STAR) GO TO 85
      RADPT=0.99*RDROP
      DRT=-0.01*RDROP
      RADPT3=RADPT**3
      DMAST=PARTRM*(RADPT3-RDROP3)
      VAPMT=VAPM(I4)-DMAST
      TOTCMT=TOTCM(I4)-DMAST
      YTNT=VAPMT/TOTCMT
      DYTDR=(YTNT-Y1TIL)/DRT
      TPT=(HLAT*(RADPT-RDROP))/XX1 + TDROP
      IF(TPT.LT.250.) TPT=250.
      IF(TPT.GE.TDROP) GO TO 85
      TB=TPT*0.1
      ITB=INT(TB)
      FR=TB-FLOAT(ITB)
      PVT=FR*PVAP(ITB+2) + (1.0-FR)*PVAP(ITB+1)
      YSNT=MW(1)/(MW(1)+WAIR*(P(I4)/DMIN1(P(I4),PVT)-1.0))
      DYSDR=(YSNT-Y1STAR)/DRT
      DYDR=DYSDR-DYTDR
      IF(ABS(DYDR).LT.1.0E-10) GO TO 85
      DRT=(Y1TIL-Y1STAR)/DYDR
      RADPT=RDROP+DRT
      RADPP(N)=RADPT
      IF(RADPP(N).LT.0.0) RADPP(N)=0.0
      IF(RADPP(N).GT.RDROP) RADPP(N)=RDROP
      IFLAGT=1
      GO TO 75
   85 TOTCM(I4)=TOTCM(I4)-DMASS                 ("85" added)        line 2056
                        .   .
                        .   .
      END                                                           line 2070
```

APPENDIX B: STATE SUBROUTINE CHANGES

The following are the modifications made to the STATE subroutine. Line numbers at the right correspond to the original program listing in the KIVA Manual. ***Changes to the program are represented by bold, oblique print.*** *Omitted parts of the program are represented by "• •".*

```
      SUBROUTINE STATE                                          line 4553
C
C      ******************************************************
C                 SIE "ROBIN HOOD" ALGORITHM
C      ******************************************************
C
C      ESTABLISHING CELL VERTICES (TAKEN FROM CCFLUX SUBROUTINE)
C
      DO 150 K=1,NZ
        I4B=(K-1)*NXPNYP
        DO 140 J=1,NY
         I4=I4B+(J-1)*NXP+1
         DO 130 I=1,NX
          IF(F(I4).EQ.0.) GO TO 130
           IF(SIE(I4).LT.SIEMIN .OR. SIE(I4).GT.SIEMAX) THEN
          I41=I4-1
          IF(I41.LE.0) I41=I4
          IF(I.EQ.1 .OR. F(I41).EQ.0.) THEN
            IMJK=I4
          ELSE
            IMJK=I4-1
          ENDIF
          IF(I.EQ.NX .OR. F(I4+1).EQ.0.) THEN
            I1=I4
          ELSE
            I1=I4+1
          ENDIF
          I4NXP=I4-NXP
          IF(I4NXP.LE.0.) I4NXP=I4
          IF(J.EQ.1 .OR. F(I4NXP).EQ.0.) THEN
            IJMK=I4+NXPNYM*ICYL
          ELSE
            IJMK=I4-NXP
          ENDIF
          IF(J.EQ.NY .OR. F(I4+NXP).EQ.0.) THEN
```

```
              I3=I4-NXPNYM*ICYL
            ELSE
              I3=I4+NXP
            ENDIF
            I44=I4-NXPNYP
            IF(I44.LE.0.) I44=I4
            IF(K.EQ.1 .OR. F(I44).EQ.0.) THEN
              IJKM=I4
            ELSE
             IJKM=I4-NXPNYP
            ENDIF
            IF(K.EQ.NZ .OR. F(I4+NXPNYP).EQ.0.) THEN
              I8 =I4
            ELSE
              I8 = I4+NXPNYP
            ENDIF
C *****************************************************************
C       SIE ROBIN HOOD -- USING ALL SURROUNDING CELLS TO CORRECT
C           THE INTERNAL ENERGY VALUE OF A "TROUBLE CELL" WHEN IT
C           EXCEEDS THE UPPER OR LOWER LIMIT
C          USE IF (SIE(I4) < SIEMIN) .OR. (SIE(I4) > SIEMAX )
C ******************************************************************
C
C     APPLY NAMES THE VERTICES OF THE SIX CELLS SURROUNDING THE
C       CURRENT CELL (I4)
C
           NROB(1)=IMJK
           NROB(2)=I1
           NROB(3)=IJMK
           NROB(4)=I3
           NROB(5)=IJKM
           NROB(6)=I8
C
           DO 305 L=1,6
             SSIE(L)=SIE(NROB(L))
  305      CONTINUE
C   ===============================================
C    LOWER LIMIT EXCEEDED:  IF SIE(I4) < SIEMIN
C   ===============================================
C     SET FLAG=1 FOR SURROUNDING CELLS WHICH ARE NOT ABLE TO DONATE
C       ENERGY TO THE "TROUBLE CELL" AT I4
            IF(SIE(I4).LT.SIEMIN) THEN
              DO 306 L=1,6
                ISFLAG(L)=0
                 IF(SSIE(L).LE.SIEMIN) ISFLAG(L)=1
  306         CONTINUE
```

```
C    ADD UP SIE VALUES FROM SURROUNDING CELLS WITH SIE   > SIEMIN
C        TO OBTAIN TOTAL ENERGY AVAILABLE  FOR DONATING TO TROUBLE CELL
C
              SENG=0.
              DO 320 L=1,6
                IF(ISFLAG(L).EQ.1) THEN
                  ENG(L)=0.
                  GO TO 320
                ENDIF
                 ENG(L)=(SSIE(L)-SIEMIN)*RO(NROB(L))*VOL(NROB(L))
                SENG=SENG+ENG(L)
  320         CONTINUE
C
C     CALCULATE ENERGY REQUIRED BY CELL I4
C
              DENEGY=(SIEMIN-SIE(I4))*RO(I4)*VOL(I4)
              IF(DENEGY.GT.SENG) THEN
                SIE(I4)=SIEMIN
                GO TO 125
              ENDIF
C
C    CALCULATE AMOUNT DONATED BY EACH CELL (WEIGHTED),
C      THEN FIND THE NEW CELL ENERGY
C
              DO 340 L=1,6
                DENG(L)=DENEGY*ENG(L)/SENG
                 SSIE(L)=SSIE(L) - DENG(L)/(RO(NROB(L))*VOL(NROB(L)))
  340          CONTINUE
C
C    RE-ESTABLISH ENERGY VALUES WITH ORIGINAL CELL VERTEX POSITION
C
              DO 345 L=1,6
                SIE(NROB(L))=SSIE(L)
  345          CONTINUE
            SIE(I4)=SIEMIN
        ELSE
C ===============================================
C    UPPER LIMIT EXCEEDED:   IF SIE(I4) > SIEMAX
C ===============================================
C     SET FLAG=1 FOR SURROUNDING CELLS WHICH ARE NOT ABLE TO
C       RECEIVE ENERGY FROM THE "TROUBLE CELL" I4
C
           DO 350 L=1,6
             ISFLAG(L)=0
             IF(SSIE(L).GT.SIEMAX) ISFLAG(L)=1
  350      CONTINUE
```

```
C      ADD UP SIE VALUES FROM SURROUNDING CELLS WITH  SIE < SIEMAX
C         TO OBTAIN TOTAL ENERGY THAT COULD BE BORROWED FROM TROUBLE CELL
C
            SENG=0.
            DO 360 L=1,6
              IF(ISFLAG(L).EQ.1) THEN
                ENG(L)=0.
                GO TO 360
              ENDIF
               ENG(L)=(SIEMAX-SSIE(L))*RO(NROB(L))*VOL(NROB(L))
              SENG=SENG+ENG(L)
  360       CONTINUE
C
C    CALCULATE EXCESS ENERGY TO BE BORROWED BY SURROUNDING CELLS
C       DENEGY=(SIE(I4)-SIEMAX)*RO(I4)*VOL(I4)
C
            IF(DENEGY.GT.SENG) THEN
              SIE(I4)=SIEMAX
              GO TO 125
            ENDIF
C
C    CALCULATE AMOUNT  BORROWED BY EACH SURROUNDING CELL (WEIGHTED),
C       THEN FIND NEW CELL ENERGY
C
            DO 370 L=1,6
              DENG(L)=DENEGY*ENG(L)/SENG
               SSIE(L)=SSIE(L) + DENG(L)/(RO(NROB(L))*VOL(NROB(L)))
  370       CONTINUE
C
C    RE-ESTABLISH ENERGY VALUES WITH ORIGINAL CELL VERTEX POSITIONS
            DO 380 L=1,6
              SIE(NROB(L))=SSIE(L)
  380       CONTINUE
            SIE(I4)=SIEMAX
          ENDIF
  125     IF(ITFLAG.EQ.0) ITFLAG=1
        ENDIF
  130 I4=I4+1
  140 CONTINUE
  150 CONTINUE
C======================  (END OF SIE ROBIN HOOD) ====================
      DO 70 K=1,NZ
                        •   •
                        •   •
      END
```

APPENDIX C: CHEM SUBROUTINE CHANGES

The following are the modifications made to the CHEM subroutine. Line numbers at the right correspond to the original program listing in the KIVA Manual. ***Changes to the program are represented by bold, oblique print.*** *Omittted parts of the program are represented by "• •"*

```
      SUBROUTINE CHEM                                              line 907
                              •    •
                              •    •
   50 CONTINUE                                                     line 972
      IF(SPD(I4,2).LT.1.D-08) THEN
      NFLAG=0
         GO TO 51
      ENDIF
      OFR=SPD(I4,1)/SPD(I4,2)
      IF(OFR.GE.0.171 .AND. OFR.LE.1.22 )    THEN
         NFLAG=1
       ELSE
         NFLAG=0
      ENDIF
   51 ROM=SPD(I4,IREF)*RMW(IREF)                 ("51" added)      line 973
                              •    •
                              •    •
      CBOT=FLAM*KF*RP + FLBM*KB*PP                                 line 977
      DOMMAX=ROM/DABS(FLAM-FLBM)                          (equation added)
      DOMEGA(IR)=ROM*DT*(CTOP-CBOT)/((ROM+DT*CBOT)*(FLBM-FLAM))    line 978
      IF(DOMEGA(IR)*(FLBM-FLAM).LT.0.) THEN
         DOMEGA(IR)=DSIGN(1.D0,DOMEGA(IR))*DMIN1(DABS(DOMEGA(IR)),
     1   DABS(DOMMAX))
      ENDIF
      IF(IR.EQ.1 .AND. NFLAG.EQ.1 .AND. TEMP(I4).LT. 1200.) THEN
        DSPD1=DABS(SPD(I4,1)/(MW(1)*FBMAM(1,1)))
        DSPD2=DABS(SPD(I4,2)/(MW(2)*FBMAM(2,1)))
        DOME=DMIN1(DSPD1,DSPD2)*DCHECK
        DOMEGA(IR)=DMAX1(DOMEGA(IR),DOME)
      ENDIF
      DO 60 ISP=1,NSP                                              line 979
                              •    •
                              •    •
      END                                                          line 1008
```