APPENDIX A

This appendix contains the listing of the Igor (Wavemetrics, Inc.) Macros used in the acquisition of the data in this work. The original Igor Macros written by Warren (1994) were significantly modified to meet the needs of this research. The new macros automatically name each file to identify their data contents following the naming convention used by the data management system designed in this work. Thanks are expressed to Stewart Graham who contributed in developing part of the Single-Shot data acquisition macro.

```
Menu "Macros"
		"Startup/1"
		"OpenShutter/2"
		"CloseShutter/3"
		"Tweek/4"
		"ColdAir/5"
		"DyeProfile/6"
		"XenonLamp/7"
		"AvgData/8"  // This macro is not listed here because it was not used during acquisition
		"SSData/9"
End

Macro Startup (phi_i,sw_i,temp,p ress)
		variable temp=67, press=640.0
		string phi_i, sw_i, thepath
		prompt phi_i, "Equivalence ratio:",popup,"0.65;0.80;1.00;1.10;1.2 "
		prompt sw_i, " Swirl:", popup, "high;medium;low"
    prompt temp, "Enter the ambient temperature (F):"
		prompt press, "Enter the ambient pressure (mm of Mercury):"

//		Setup detector temperature.  Needs set up at least 30 minutes prior to taking data.
    variable detectTemp = -5  //Lowest Temperature attainable without special cooling, if room temperature < 25 C.
					//See OMA manual.
		oma "dt " + num2istr ( detectTemp )

//  Setup the contents of startup file
		string/g basefile, datapath
		string  logFile,date,time
		variable logFileno = 32
			silent 1
			slow 0
		date = secs2date(datetime,1)
		time = secs2time(datetime,0)
    basefile = sw_i[0] + "sPhi"+ phi_i
// Convert T and P to K and atm respectively
    temp = (temp - 32)/1.8 + 273
    press = press/760
//		select the path to be saved
		newPath/o/m = "SELECT THE FOLDER where you want to save data"  thepath
		pathinfo thepath
//  Save the selected path to "datapath" which is used by the OMA XOP		to dump the collected data
  datapath = s_path
		logFile = "C_log.txt"  // "C_" stands for Calibration
  open/P= thepath logFileno as logFile
		fprintf logFileno,  "Date: %s\rTime: %s\rSwirl: %s\rPhi: %s\r" date,time,sw_i, phi_i
		fprintf logFileno, "Ambient Temperature (K): %5.1f\rAmbient Pressure (atm): %5.2f\r" temp, press
		close logFileno
EndMacro


Macro OpenShutter ()
		silent 1
		slow 0
		//	Initialize Auto Shutter
		oma "ddra 8"
		//	Open Shutter
		oma "port.a 8"
EndMacro

Macro CloseShutter ()
		silent 1
		slow 0
```

```
	//	Initialize Auto Shutter
	oma "ddra 8"
	//	Open Shutter
	oma "port.a 0"
EndMacro

Macro Tweek ()
	silent 1
	slow 0

	oma "an"
	oma "SS; FA 0; NORM 1024; FA 0; ES"
//	oma "smin.et"
	oma "et .10"
	oma "ryo; dp 0; do 1; add; add; loop; stop; exit"
	oma "clr 1"
	oma "mem 1;run"

	oma "save first=1 last=1 numshots=1 wave <TWEEKWAVE>"
	Spectra_Window ( "TWEEKWAVE" )
	do
		oma "clr 1"
		oma "mem 1;run"
		oma "save first=1 last=1 numshots=1 wave <TWEEKWAVE> quiet"
	while (1)
EndMacro

Macro ColdAir ( nShots, fileName )
	variable nShots = 400
	string fileName = "C_air.dat"  // "C_" stands for Calibration
	prompt nShots, "Enter the number of shots to be collected:"
	prompt fileName, "Enter the name of the file to be saved:"

	silent 1
	slow 0

	//	Setup scan setup (an) and exposure time
	oma "an"
	oma "smin.et"

	//	Clear memories and program OMA for average data
	oma "clr 1; clr 2"
	oma "ryo; dp 1; do " + num2istr ( nShots) + ";  on.trig; add; add; loop; stop; exit "

	OpenShutter()
	//  Take CARS spectra
	oma "mem 1;run"

	CloseShutter()

	//	Collect and subtract the background
	oma "mem 2;run"
	oma "sub 2 1"

	//	Save and display data
	oma "save avg first=1 last=1 numShots=" + num2istr (nShots) + " '"+fileName+"' wave <COLD_SPECTRA>"
	COLD_SPECTRA = COLD_SPECTRA / nShots
	Spectra_Window ("COLD_SPECTRA")

	OpenShutter()
EndMacro
```

```
Macro DyeProfile ( nShots, fileName, dye )
variable nShots = 400,  dye = 1
string fileName = "C_dyeProfile"
prompt nShots, "Enter the number of shots to be collected:"
prompt fileName, "Enter the base name of the file to be saved (dye type will be added later):"
prompt dye, "Dye:",popup, "All;  N2 side;  O2 side"

	silent 1
	slow 0

	if ( dye == 1)
		fileName = fileName + ".dat"  //Collect dye profile for both sides of the spectrometer
	endif
	if ( dye == 2)
		fileName = fileName + "_n2.dat"
	endif
	if ( dye == 3)
		fileName = fileName + "_o2.dat"
	endif

	//	Setup scan setup (an) and exp osure time
	oma "an"
	oma "smin.et"

	//	Clear necessary memory and program OMA for average data
	oma "clr 1; clr 2"
	oma "ryo; dp 1; do " + num2istr ( nShots) + ";  on.trig; add; add; loop; stop; exit "

	OpenShutter()

	//	Collect data
	oma "mem 1;run"

	CloseShutter()

	//	Collect background
	oma "mem 2;run"
	oma "sub 2 1"

	//	Save data
	oma "save ss first=1 last=1 numShots=" + num2istr (nShots) + " '"+fileName+"' wave <DYE_PROFILE>"

	DYE_PROFILE = DYE_PROFILE / nShots

	Spectra_Window ( "DYE_PROFILE" )

	OpenShutter()

EndMacro

Macro XenonLamp ( fileName)
string fileName =  "C_xen.dat"  // "C_" stands for Calibration
prompt fileName, "Enter the filename to be saved:"
variable numScans = 20

	silent 1
	slow 0

	// Note: the OMA command "an" must be issued by itself because it is intercepted by the OMA program
	oma "an"
	oma "et .1"
	oma "clr 1; clr 2"
```

```
        oma "ryo; dp 1; do " + num2istr(numScans) + "; add; add; loop; stop; exit"

        oma "mem 1; run"
    beep
    doAlert 1, "Are you ready to collect Xe background? "
    if (V_flag == 1)
    endif
        //          Collect and subtract the background
        oma "mem 2;run"
        oma "sub 2 1"

        oma "save avg first=1 last=1 numShots=" + num2istr (numScans) + " '"+fileName+"' wave <XENON>"

    XENON = XENON/numScans
        Spectra_Window ( "XENON" )

//  Get alignment-useful information on Xe peaks
    printf "\t\t\tPeak 1\t\t\t\t\tPeak 2\t\t\t\t\tPeak3\t\t\t\t\tPeak4\t\t\t\t\tPeak5\r"
        findPeaks(xenon,5)

EndMacro


Macro SSData ( nShots, scanSetup, fileName, axi, rdi, updn, inout)
        variable nShots = 1000, detectTemp = 2,  updn = 1,inout = 1,axi=1,rdi=1
        string scanSetup = "SS; FA 0; norm 1024; FA 0; ES"
        string fileName=basefile
        prompt nShots, "Enter the number of single shots to be collected:"
        prompt scanSetup, "Enter the scan setup:"
        prompt fileName, "Enter the base name of the file to be saved:"
        prompt axi, "Enter the begining axial position:", popup,"  5; 10; 20; 30; 40; 50; 60; 70; 80;
                                        90;100;112;125;138;150;162;175;188;200;212;225;238;250"
        prompt rdi, "Enter the begining radial position:", popup," -6; -3;   0;  3;  6;  9;  12;  15;  18;  21;  24;  27;  30;
                                                                32"
        prompt updn, "Direction of axial sample locations:", popup, " up; down"
        prompt inout, "Direction of radial sample locations:", popup, " wall; center"
        make/o rad={99,-6,-3,0,3,6,9,12,15,18,21,24,27,30,32},
             ax={99,5, 10, 20, 30, 40, 50, 60, 70, 80, 90,100,112,125,138,150,162,175,188,200,212,225,238,250}

        variable extnum, extcnt, pointcnt = 1, i = 1,xa,bee,honey,flag,bypass
        variable nBGShots
        string  ext,axii,radd

//  Variables dependent on mapping -->  numrad = # of radial locations; numaxi = # of axial locations
        variable numrad = 14, numaxi = 23

        silent 1
        slow 0
        flag=0
        if (((rdi==1) %& (updn==2) %& (inout==2))  %| ((rdi==numrad) %& (updn==1) %& (inout==1)))
          print "hello"
          flag=1
        endif
        bypass=0

        if (updn ==1)
          bee=1
        else
          bee=-1
        endif

        if (inout ==1)
```

```
		  honey=1
		else
		  honey=-1
		endif

//		Don't need more than 200 background shots so...
		if ( nShots > 200 )
		  nBGShots = 200
		else
		  nBGShots = nShots
		endif

//		Setup the scan setup, and exposure time
		oma scanSetup
		oma "smin.et"

		do
//		Initialize the extension control variables
		  radd=num2str(rad(rdi))
		  axii=num2str (ax(axi))
		  if (ax(axi) < 10)
		    ext = "_z00"+axii
		  endif
		  if ((ax(axi) >= 10) %& (ax(axi) < 100))
		    ext = "_z0"+axii
		  endif
		  if (ax(axi) >= 100)
		    ext = "_z"+axii
		  endif
		  if((rad(rdi) >= 0) %& (rad(rdi) < 10))
		    ext = ext + "_r0"+radd
		  else
		    ext = ext + "_r"+radd
		  endif

		  OpenShutter()

//		Program the OMA to take individual shots
		  oma "ryo; dp 0; do " + num2istr (nShots) +  "; on.trig; add; add; inc.mem; loop; stop; exit"

//		Clear the OMA memory and set the memory counter to one
		  oma "clr.all"
		  oma "mem 1"

		  beep
		  doAlert 1, "Are you ready to collect the SS CARS signal to be saved in " + fileName +ext+".dat?"
		  if (V_flag == 1)
//		Go!!
		    oma "run"

//		Save the data onto disk and into wave  '"
		  oma "save ss avg first=1 last=" + num2istr (nShots) +" numShots= 1 '"+fileName+ ext
		                                          +".dat' wave <SINGLE_AVERAGE>"

		    CloseShutter()

//		Program the OMA to average shots in one memory location (Note: there is no inc.mem)
		    oma "ryo; dp 1; do " + num2istr ( nBGShots) + ";  on.trig; add; add; loop; stop; exit "

//		Only need one location so clear it and set the memory counter
		    oma "clr 1; mem 1"
```

```
//          Don't need user intervention with Auto Shutter
              oma "run"

//          Save the data and get the curve
              oma "save avg first=1 last=1 numShots=" + num2istr (nBGShots) + "'"+fileName+ext
                                                              + ".bg' wave <SINGLE_BG>"

//          Normalize Curve

              Wavestats /Q SINGLE_AVERAGE
              Make/N=(V_npnts)/O SS_DATA

              SS_DATA = SINGLE_AVERAGE/nShots - SINGLE_BG/nBGShots

              if ( i == 1)
                Spectra_Window ( "SS_DATA" )
                i = 2
              endif

            else

              doAlert 1, "Do you want to skip to next point?"
              if (V_flag == 2)
                killwaves/a/z
                abort
              endif

            endif


//          Increment the extension


            rdi+=honey

            if (flag==1)
              axi +=bee
              if ((axi==0) %| (axi==numaxi + 1))
                killwaves/a/z
                abort "Thank You, Come Again!!"
              endif

              flag=0
              bypass=1

               if (honey==1)
                 rdi=numrad
                 honey=-1
               else
                 honey=1
                 rdi=1
               endif
            endif

            if(((rdi==1)  %|  (rdi==numrad)) %& (bypass==0))
              flag =flag+1
            endif
            if (bypass==1)
              bypass=0
            endif
//  this is wrote in case you start taking data at 33 going up or -6 going down
            if (rdi==0)
```

```
			rdi=1
			honey=1
			axi=axi+bee
		endif
		if (rdi==numrad + 1)
			rdi=numrad
			honey=-1
			axi=axi+bee
		endif
	while(1)

EndMacro

Window Spectra_Window(waveName) : Graph
string waveName, list
variable type

	PauseUpdate; Silent 1		| building window...

	if ( WinType ("Spectra_Window") == 1 )
		DoWindow /K Spectra_Window
	endif

	SetScale/P x 1,1,"", $waveName
	Display /W=(76,203,756,503) $waveName
	label bottom "Pixel Number";DelayUpdate
	label left "Intensity"

EndMacro

Window Diagnostic_Volume() : G
	PauseUpdate; Silent 1
	//		Make new waves to elimate numerical integration errors caused by small number of points
	make /o /n=1000 new_intensity, new_location, cdf, a5, a95
	new_intensity = intensity
	new_location = location
	a5 = 5
	a95 = 95
	duplicate /o new_intensity cdf
	integrate cdf
	wavestats cdf
	cdf*=100/v_max

	Display /W=(5,42,491,313) new_INTENSITY vs new_LOCATION
	Append cdf,a5,a95 vs new_LOCATION
	Modify rgb(Cdf)=(65535,3831,26947),rgb(a5)=(7149,65535,62124)
	Modify rgb(a95)=(7149,65535,62124)
	Modify grid=2
	Modify minor(bottom)=1
	Modify sep(bottom)=2
EndMacro
```

APPENDIX B

This appendix lists the most important code developed in this research:

FMCARS B-1
DatPre B-9
F4CSTX B-13
IPDARS B-25
VOIGT FUNCTION IN CARSFT B-28
MSCSTS B-32
StatLib B-33

FMCARS is the program that handles the logistics of the multi-species CARS data reduction process for one or more CARS measurement files. FMCARS first calls DatPre to pre-process the CARS data, and then calls F4CSTX to reduce the pre-processed CARS data. F4CSTX is a modified version of the code FTCARS (Palmer, 1989). For brevity, the code for F4CSTX does not include supporting code such as graphics, I/O and mathematical subroutines. Modifications are noted by the comment lines starting as "Cbyu". The new subroutines COCHI, CO2CHI, N2CHI, and O2CHI were based on the original CHICAL subroutine.

IPDARS calculates the Raman Shift for the IPDA from a Xe spectrum. VOIGT FUNCTION IN CARSFT shows the code added to CARSFT (Palmer, 1989) to calculate the instrument functions. MSCSTS was used to compute the statistics at each data location.

A complete listing of all the code used in this work is found in the accompanying CD.

FMCARS

```
      PROGRAM FMCARS
C*******  Fit Multi-specie CARs Spectra   ********
C  Version 1.0
C  Programmer:  Daniel V. Flores      June, 1999
C
C  FMCARS is designed to process multiple single-shot data files of
C  Multi-specie CARS spectra using the subroutine F4CSTX. The CARS signatures
C  of N2, CO, CO2 and O2 can be looked for in the spectra recorded by the BYU
C  optics lab CARS instrument.
C  FMCARS reports for each data file on a shot-by-shot basis the Temperature,
C  measured mole fractions, and measures of the fit goodness.
C
C  Required INPUT files (filename extension in parenthesis):
C       DATA:
C          list of data files to be processed,
C          experiment log data ('.log'), background ('.bg'),
C          single-shot data ('.dat'), dye profile ('.dye'),
C          natural gas composition ('ngas.yi')
C       THEORY:
C          Fitting libraries for each species: t****.libN2, x.libN2 and so on.
C          File 'fmcars.par' --> Contains Fit-parameter control information.
C
C  FMCARS can also work on an interactive basis, allowing the choice of species
C  to fit, ploting on TEK window, as well as access to specific shot number in
C  the single-shot data file
C
C  Required SUBROUTINE files:
C       datPre.f f4cstx.f crslib.f flush.c stepit.f
C       To access interactive capabilities, f4debg.f is required and the
C       "D" commented statements fmcars.f must be activated.  The HP-f77 linker
C       will activate them by using the flag -D. Otherwise these "D" comments
C       have to be removed manually.
C
      INTEGER IPDAPX, MAXSTR, NCO2GE, NO2GUE
      PARAMETER (IPDAPX = 1024, MAXSTR = 80, NO2GUE = 3, NCO2GE = 5)
C  IMPORTANT NOTE!:  IPDAPX parameter must match the ND parameter in F4CSTX
C                 subroutines.
      CHARACTER*(MAXSTR) BGFIL, DATFIL, DYEFIL, FILIST
      CHARACTER CHOICE*3, ERRMES*(MAXSTR)
      INTEGER FITFLG, IFIRSC, LENTH, LISNUM, NSPECT, STRBEG, STREND
      INTEGER SAVFLG
      INTEGER UNINUM
D     INTEGER DESNUM
D     LOGICAL EXTNUM
      LOGICAL FIRSFT, FIRSBD, GOODFT, LTRAN0, MORESP
      REAL IBG(IPDAPX), IDYE(IPDAPX), PIXLRS(IPDAPX), SICARS(IPDAPX)
      REAL ARERN2, ARERCO, ARERO2, ARRCO2, AREFIT
      REAL CCO2GE(NCO2GE), CCOGUE, CN2GUE, CO2GUE, EQRAT, TEMP,
     1     TGUESS
      REAL CHNRCO, CNRCO2, CHNRN2, CHNRO2, CNRTOT, CHIFIT, MINSX2,SDIFX2
      REAL XCO, XCO2, XN2, XO2
      REAL WBCO, WECO, WBCO2, WECO2, WBN2, WEN2, WBO2, WEO2
      INTEGER N2PEAK, COPEAK, O2PEAK, CO2PEK , GUESS

      DATA DYEFIL/'C_dyeProfile.dat'/, LISNUM/11/, UNINUM/15/
      DATA CO2GUE/35.0/
      DATA CCO2GE/90.0, 900.0, 2300.0, 5000.0, 8000.0/

      CALL LOGO

      CALL RDPAR(TGUESS, CN2GUE)
      CCOGUE = CN2GUE

      CHOICE = 'ALL'

D     CALL CHOLIB(CHOICE)

      CALL LIBGET(CHOICE)

      CALL OPFILE(DYEFIL, UNINUM, MAXSTR,
     1           'Name of dye profile file?', 'OLD')

      CALL AVGDIN(DYEFIL, IDYE, ERRMES)

      IF(ERRMES.NE.'O.K.') THEN
        PRINT*,'Poblems with ', DYEFIL(IFIRSC(DYEFIL):LENTH(DYEFIL)),
     1      ' ', ERRMES(IFIRSC(ERRMES):LENTH(ERRMES))
        STOP
      ENDIF

      CALL GETRS(PIXLRS)

      CALL WAVLIM(CHOICE, WBCO, WECO, WBCO2, WECO2, WBN2, WEN2,
     1            WBO2, WEO2)

      CALL REALOG(EQRAT)

C Set XN2 to average value from combustion mass balance reaction progress.
```

```
      IF(EQRAT.GT.0.7) XN2 = 0.73
      IF(EQRAT.LT.0.7) XN2 = 0.74
      PRINT*, 'XN2 = ', XN2

      CALL OPFILE(FILIST, LISNUM, MAXSTR,
     1               'Name of file listing raw data files', 'OLD')

10    READ(LISNUM,'(A)', END = 40) DATFIL

      IF(DATFIL.EQ. ' ') GOTO 10

      STRBEG = IFIRSC(DATFIL)

      STREND = LENTH(DATFIL)

      BGFIL = DATFIL(STRBEG:INDEX(DATFIL,'.dat')) // 'bg'

      CALL AVGDIN(BGFIL, IBG, ERRMES)

      IF(ERRMES.NE.'O.K.') THEN
        PRINT*, 'Background file Problem: ',
     1         ERRMES(IFIRSC(ERRMES):LENTH(ERRMES))
        PRINT*, 'SKIPPING ', DATFIL(STRBEG:STREND)
        GOTO 10
      ENDIF

      NSPECT = 0

      FIRSFT = .TRUE.

      FIRSBD = .TRUE.

      MORESP = .TRUE.

      PRINT*,'Working on file ', DATFIL(STRBEG:STREND)

D     DESNUM = 0
D     EXTNUM = .FALSE.
D     PRINT*,' '
D15   CALL GETNUM(NSPECT, DESNUM, EXTNUM, MORESP)
D     IF(.NOT.MORESP) GOTO 10

20    CALL SSDPRE(DATFIL(STRBEG:STREND), IBG,IDYE,SICARS,NSPECT,PIXLRS,
     1          WBN2, WEN2, WBCO, WECO, WBO2, WEO2, WBCO2, WECO2,
     2           MORESP, ERRMES, N2PEAK, COPEAK, O2PEAK, CO2PEK)
```

```
D     IF(EXTNUM.AND.NSPECT.NE.DESNUM) GOTO 20

      IF(INDEX(ERRMES,'Opening').GT.0) THEN
        PRINT*, ERRMES(1:LENTH(ERRMES)),'; SKIPPING ',
     1          DATFIL(STRBEG:STREND)
        GOTO 10
      ELSEIF(ERRMES .NE. 'O.K.') THEN
        CALL DIAGNS(1, .FALSE., 'none', ERRMES, GOODFT)
        GOTO 30
      ENDIF

C Re-Initialize Temperature and (totalNon-resChi)/Xi.
      TEMP = TGUESS
      CHNRN2 = CN2GUE
      CHNRCO = CCOGUE
      GOODFT = .TRUE.


D     IF(CHOICE.EQ.'ALL'.OR. CHOICE.EQ.'DEF' .OR. CHOICE.EQ.'N2') THEN
D       PRINT*,' Shot Number ', NSPECT

      IF(N2PEAK.GT.0) THEN
C Set maximum fitting delta of Chinr/Xn2 = 1. Worked best for my samples
C This fitting parameter determined whether STEPIT continued scanning:
C If "too large" or "too low" it returns bad fits.
      CALL CHAPAR(4, 6, 1.0)
      CALL F4CSTX(WBN2, WEN2, IPDAPX, PIXLRS, SICARS, 'N2 ', .FALSE.,
     1            TEMP,  CHNRN2, ARERN2, FITFLG, LTRAN0, SDIFX2)
      ELSE
C Bad shot: skip it.
        ERRMES = 'N2 spectrum peak counts are .LE. 0'
        CALL DIAGNS(FITFLG, LTRAN0, 'N2', ERRMES, GOODFT)
        GOTO 30
      ENDIF
      IF(FITFLG.LE.0 .OR. LTRAN0) THEN
        CALL DIAGNS(FITFLG, LTRAN0, 'N2', ERRMES, GOODFT)
        GOTO 30
      ENDIF

D     ENDIF

C     IF(CHOICE.EQ.'ALL' .OR. CHOICE.EQ. 'CO') THEN
C       PRINT*,' Shot Number ', NSPECT

C Remove comment specifier to also fit CO. Do the same in LIBGET subroutine
C found in f4cstx.f
```

```
C     IF(COPEAK.GT.0) THEN
C     CALL F4CSTX(WBCO, WECO, IPDAPX, PIXLRS, SICARS, 'CO ', .TRUE.,
C    1          TEMP,  CHNRCO, ARERCO, FITFLG, LTRAN0, SDIFX2)
C     ELSE
C       CHNRCO = -1
C     ENDIF
C     IF(FITFLG.LE.0 .OR. LTRAN0) THEN
C       CHNRCO = -1
C     ENDIF

C     ENDIF

D     IF(CHOICE.EQ.'ALL' .OR. CHOICE.EQ. 'O2') THEN
D       PRINT*,' Shot Number ', NSPECT

      IF(O2PEAK.GT.0) THEN
C Set maximum fitting delta of Chinr/Xo2 = 50. Worked best for my samples
        CALL CHAPAR(4, 6, 50.0)
        CHNRO2 = CO2GUE
      CALL F4CSTX(WBO2, WEO2, IPDAPX, PIXLRS, SICARS, 'O2 ', .TRUE.,
     1          TEMP,  CHNRO2, ARERO2, FITFLG, LTRAN0, SDIFX2)
      ELSE
C Set (Total Chinr)/Xi to -1 when there was no CARS signal for species i
        CHNRO2 =  -1
      ENDIF
      IF(FITFLG.LE.0 .OR. CHNRO2.EQ.0.0) THEN
        CHNRO2 =  -1
      ENDIF

D     ENDIF

D     IF(CHOICE.EQ.'ALL' .OR. CHOICE.EQ. 'CO2') THEN
D       PRINT*,' Shot Number ', NSPECT

      IF(CO2PEK.GT.0) THEN
C Set maximum fitting delta of Chinr/Xco2 = 50. Worked best for my samples
        CALL CHAPAR(4, 6, 50.0)
        DO 22 GUESS=1, NCO2GE
          CHIFIT = CCO2GE(GUESS)
      CALL F4CSTX(WBCO2, WECO2, IPDAPX, PIXLRS, SICARS, 'CO2', .TRUE.,
     1          TEMP,  CHIFIT, AREFIT, SAVFLG, LTRAN0, SDIFX2)
          IF(GUESS.EQ.1 .OR. SDIFX2.LT.MINSX2) THEN
            MINSX2 = SDIFX2
            ARRCO2 = AREFIT
            CNRCO2 = CHIFIT
            FITFLG = SAVFLG
          ENDIF
22      CONTINUE
      ELSE
        CNRCO2 = -1
      ENDIF
      IF(FITFLG.LE.0 .OR. CNRCO2.EQ.0.0) THEN
        CNRCO2 = -1
      ENDIF

D     ENDIF

ChinrTot/Xi Uncomment to print (Total Chinr)/Xi instead of Xi
C     XN2 = 1.0

      CNRTOT = CHNRN2*XN2
      IF(CHNRO2.GT.0) THEN
        XO2 = CNRTOT/CHNRO2
      ELSE
        XO2 = 0
      ENDIF
      IF(CNRCO2.GT.0) THEN
        XCO2 = CNRTOT/CNRCO2
      ELSE
        XCO2 = 0
      ENDIF

ChinrTot/Xi Uncomment to print (Total Chinr)/Xi instead of Xi
C     XO2 = CHNRO2
C     XCO2 = CNRCO2

30    CALL PRNRES(DATFIL, ERRMES, FIRSFT, FIRSBD, GOODFT, MORESP,
     1          NSPECT, CNRTOT, TEMP, XN2, XCO, XCO2, XO2,
     2          N2PEAK, COPEAK, O2PEAK, CO2PEK,
     3          ARERN2, ARERCO, ARERO2, ARRCO2)

D     CALL WRIPSS(PIXLRS, SICARS, DATFIL, NSPECT)
C  Reset all parameters to initial guesses input by user in fmcars.par
      CALL RSVPAR

D     IF(MORESP.AND.EXTNUM) GOTO 15

      IF(MORESP) GOTO 20

      GOTO 10

40    CLOSE(LISNUM)
```

```
      PRINT*, 'PROGRAM TERMINATED NORMALLY....'

      END

      SUBROUTINE CHAPAR(RUNPAR, FITVAR, NEWVAL)
Cbyu  ******** CHAnge the value of Run PARameters ******
      INTEGER NP, NPP
      PARAMETER (NP = 30, NPP = 31)
      COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
     1  DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
     2  MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
     3  KW

      REAL FITPAR, XMAX, XMIN, DELTX, DELMN, ERR, FOBJ
      INTEGER  NPARAM, NTRAC, MATRX, MASK, NFMAX, NFLAT, JVARY, NXTRA,
     1       KFLAG, NOREP, KERFL, KW
      INTEGER RUNPAR
      REAL FITVAR, NEWVAL

      IF(RUNPAR.EQ.4) DELTX(FITVAR) = NEWVAL

      RETURN
      END

      SUBROUTINE CHOLIB(CHOICE)
C     ******  CHOose the LIBraries to be read *******
      CHARACTER*3 CHOICE, PROMPT(6)*20
      INTEGER OPTION

      DATA PROMPT/'FTCARS Default', 'CO only', 'CO2 only', 'N2 only',
     1        'O2 only' , 'All four species'/

      OPTION = 0

      PRINT*, '********  CHOOSE LIBRARIES TO BE READ  ********'
      CALL MENU(0, 5, OPTION, PROMPT)

      IF(OPTION.EQ.0) THEN
        CHOICE = 'DEF'
      ELSEIF(OPTION.EQ.1) THEN
        CHOICE = 'CO '
      ELSEIF(OPTION.EQ.2) THEN
        CHOICE = 'CO2'
      ELSEIF(OPTION.EQ.3) THEN
        CHOICE = 'N2 '
      ELSEIF(OPTION.EQ.4) THEN
        CHOICE = 'O2 '
      ELSE
        CHOICE = 'ALL'
      ENDIF

      RETURN
      END

      SUBROUTINE DIAGNS(FITFLG, LTRAN0, SPECIE, ERRMES, GOODFT)
C     ***** DIAGNoSe species fitting output and/or data file access *****
C     ERRMES will be printed to the *.fitBad file.
      CHARACTER*(*) SPECIE, ERRMES
      INTEGER FITFLG, LENTH
      LOGICAL GOODFT, LTRAN0

      GOODFT = .FALSE.

      IF(ERRMES.NE.'O.K.') THEN
        ERRMES = ERRMES(1:LENTH(ERRMES))//
     1       '. NO FIT WAS ATTEMPTED; SKIPPING REST OF FILE'
        RETURN
      ENDIF

      IF(FITFLG.LE.0) THEN
        ERRMES = 'ABNORMAL exit from stepit for'
      ELSEIF(LTRAN0) THEN
        ERRMES = 'All transitions were ZERO for'
      ENDIF

      ERRMES = ERRMES(1:LENTH(ERRMES))//' '//SPECIE//'; file was O.K.'

      RETURN
      END

      SUBROUTINE GETNUM(NSPECT, DESNUM, EXTNUM, MORESP)
C     *****  GET the record NUMber of the sp ectra to be  *****
C     *****  extracted from single-shot data file         *****
      CHARACTER ANS*1, NUMBER*5
      INTEGER NSPECT, DESNUM, SSUNIN
      INTEGER IFIRSC, LENTH
      LOGICAL EXTNUM, MORESP
C  SSUNIN is the unit number of data file in sub SSDPRE
      PARAMETER (SSuniN = 10)

      IF(.NOT.EXTNUM) THEN
```

```
      PRINT*,'Extract specific spectra? <Y>'
      READ(5,'(A)') ANS
      IF(ANS.EQ.' '.OR. ANS.EQ.'Y'.OR.ANS.EQ.'y') THEN
       EXTNUM = .TRUE.
      ELSE
       RETURN
      ENDIF
     ENDIF

     DESNUM = NSPECT + 1
     WRITE(NUMBER,'(I5)') DESNUM
     PRINT*,'Record Number to be extracted? <',
    1        NUMBER(IFIRSC(NUMBER):LENTH(NUMBER)),'>'
     PRINT*,' "A Zero or Negative # end the current file" '
     READ(5,'(A)') NUMBER
     IF(NUMBER.NE.' ') READ(NUMBER,'(I)') DESNUM

     IF(DESNUM.GT.0.AND.DESNUM.LT.(NSPECT+1)) THEN
      REWIND SSUNIN
      NSPECT = 0
     ELSEIF(DESNUM.LE.0) THEN
      MORESP = .FALSE.
      CLOSE(SSUNIN)
     ENDIF

     RETURN
     END

     SUBROUTINE GETRS(pixlRS)
C    *****  GET the Raman Shift of the ipda *****
C    The filename is IpdaRS and is fixed because the ipda dispersion remains
C    constant unless the spectrometer settings are changed manually.

     INTEGER ipdaPx
     PARAMETER (ipdaPx = 1024)
     CHARACTER*10  filNam, headng
     INTEGER i, uniNum, IFIRSC, LENTH
     REAL pixlRS(ipdaPx)
     DATA filNam /'IpdaRS'/, uniNum/15/

     OPEN(UNIT = uniNum, FILE= filNam(IFIRSC(filNam):LENTH(filNam)),
    $     STATUS = 'UNKNOWN')

     READ(uniNum, *) headng
     DO 10 i=1,ipdaPx
      READ(uniNum, *) pixlRS(i)
```



```
 10   CONTINUE

      CLOSE(uniNum)

      RETURN
      END

      SUBROUTINE LOGO
C     ***** Print the LOGO  *****
C     Modified from CARSFT subroutine.    Programmer Daniel V. Flores
      CHARACTER*8 UPDATE
      DATA UPDATE/'1.0'/
C
C::::::::::::::::::::::::::::::::::::::::::::::::::: end declarations
C
      PRINT *, ' '
      PRINT *, ' '
      PRINT *, ' '
      PRINT *,
     1'   FFFFFFF MMM      MMM    CCCCC     AAAA     RRRRRR     SSSS '
      PRINT *,
     1'   FF      MM M     M MM   CC    C   AA  AA    RR   RR   SS   S'
      PRINT *,
     1'   FF      MM  M   M  MM  CC        AA    AA   RR   RR   SS     '
      PRINT *,
     1'   FFFF    MM   M M   MM  CC        AAAAAAAA   RRRRRR     SSSS '
      PRINT *,
     1'   FF      MM    M    MM  CC        AA    AA   RR  RR        SS'
      PRINT *,
     1'   FF      MM         MM   CC    C  AA    AA   RR   RR   S   SS'
      PRINT *,
     1'   FF      MM         MM    CCCCC   AA    AA   RR    RR   SSSS '
      PRINT *, ' '
      PRINT '(T29, A, A)' , 'Version ' , UPDATE
      PRINT *,
     1'================================================================'
      PRINT *, ' '
      END

      SUBROUTINE PRNRES(DATFIL, ERRMES, FIRSFT, FIRSBD, GOODFT, MORESP,
     1           NSPECT, CNRTOT, TEMP, XN2, XCO, XCO2, XO2,
     2            N2PEAK, COPEAK, O2PEAK, CO2PEK,
     3            ARERN2, ARERCO, ARERO2, ARRCO2)


      COMMON /PARAMS/ PRESS, YAMAX
```

```
      REAL PRESS, YAMAX
      LOGICAL FIRSBD, FIRSFT, GOODFT, MORESP
      CHARACTER*(*) DATFIL, ERRMES
      CHARACTER*80 FITFIL, FTBFIL
      INTEGER FBUNUM, FTUNUM, NSPECT
      INTEGER IFIRSC, LENTH, NUMBAD
      REAL CNRTOT, TEMP
      REAL ARERN2, ARERCO, ARERO2, ARRCO2, XN2, XCO, XCO2, XO2
      REAL PERCNT
      INTEGER N2PEAK, COPEAK, O2PEAK, CO2PEK

      DATA FBUNUM, FTUNUM /50, 60/, NUMBAD /0/


C     PRINT*, 'ERRMES = ', ERRMES
      IF(GOODFT .AND. ERRMES.EQ.'O.K.') THEN
       IF(FIRSFT) THEN
         FITFIL = DATFIL(IFIRSC(DATFIL):INDEX(DATFIL,'.dat')) // 'fit'
         OPEN(UNIT=FTUNUM, FILE=FITFIL, STATUS='UNKNOWN')

         WRITE(FTUNUM,'(1X, A, F7.4)') 'N2 mole fraction =', XN2
         WRITE(FTUNUM,'(1X, A, F6.3)') 'Pressure (atm)   =', PRESS
         WRITE(FTUNUM,1000) '# Shot', 'CHINRT', 'T(K)', 'XCO', 'XCO2',
     1                 'XO2', 'N2PEAK', 'COPEAK', 'O2PEAK', 'CO2PEK',
     2                   'ARERN2', 'ARERCO', 'ARERO2', 'ARRCO2'
         FIRSFT = .FALSE.
       ENDIF

       WRITE(FTUNUM, 2000) NSPECT, CNRTOT, TEMP, XCO, XCO2, XO2,
     1                  N2PEAK, COPEAK, O2PEAK, CO2PEK,
     2                  ARERN2, ARERCO, ARERO2, ARRCO2
      ELSE
       IF(FIRSBD) THEN
         FTBFIL = DATFIL(IFIRSC(DATFIL):INDEX(DATFIL,'.dat'))//'fitBad'
         PRINT*,'  BAD FITS see ', FTBFIL(IFIRSC(FTBFIL):LENTH(FTBFIL))
     1        ,' for details'
         OPEN(UNIT=FBUNUM, FILE=FTBFIL, STATUS='UNKNOWN')

         WRITE(FBUNUM, '(1X, A, 4X, A)') '# Shot', 'ERROR MESSAGE'
         FIRSBD = .FALSE.
         NUMBAD = 0
       ENDIF

       NUMBAD = NUMBAD + 1
       WRITE(FBUNUM, '(2X, I4, 5X, A)') NSPECT,
     1     ERRMES(IFIRSC(ERRMES):LENTH(ERRMES))
```

```
      ENDIF

      IF(.NOT.MORESP) THEN
       CLOSE(FTUNUM)
       IF(NUMBAD.GT.0) THEN
         WRITE(FBUNUM,'(1X)')
         WRITE(FBUNUM,*) '# of Discarded shots = ', NUMBAD
         PERCNT = (1.0 - REAL(NUMBAD)/REAL(NSPECT))*100.0
         WRITE(FBUNUM,'(1X, A, F6.1)')
     1       'Percent of good shots = ', PERCNT
         CLOSE(FBUNUM)
       ENDIF
      ENDIF

1000  FORMAT(1X, A, 3X, A, 8X, A, 8X, A, 8X, A, 8X, A, 6X, 5(A, 3X),
     1    2X, 3(A, 5X))
2000  FORMAT(2X, I4, 3X, G12.6, 1X, G11.5, 3(1X, G10.4), 1X, 4(I7, 2X),
     1    4(1X, G10.4))
      RETURN
      END

      SUBROUTINE RDPAR(TGUESS, CN2GUE)
C
C.... THIS SUBROUTINE READS RUN PARAMETERS AND FIT PARAMETERS FROM
FILE.
C
      INTEGER NP, NPP
      PARAMETER (NP = 30, NPP = 31)
      COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
     1  DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
     2  MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
     3  KW

      COMMON /SAVPAR/ SAVPAR(NP)
      REAL SAVPAR

      REAL FITPAR, XMAX, XMIN, DELTX, DELMN, ERR, FOBJ, TGUESS
      REAL CN2GUE
      CHARACTER PARNAM*10, HEADNG*80
      INTEGER  NPARAM, NTRAC, MATRX, MASK, NFMAX, NFLAT, JVARY, NXTRA,
     1       KFLAG, NOREP, KERFL, KW
      INTEGER DUM, I, IFIXED(NP), J, NFIXED, UNINUM
      DATA PARNAM/'fmcars.par'/, UNINUM/15/
C
      OPEN (UNINUM, FILE = PARNAM, STATUS = 'OLD', ERR = 300)
      REWIND UNINUM
```

```
C
C.... READ NUMBER OF FITTING PARAMETERS, NUMBER OF FIXED FITTING
C.... PARAMETERS.
C
      READ(UNINUM,*, ERR = 350) HEADNG
      READ(UNINUM,*, ERR = 350) HEADNG
      READ (UNINUM, *, ERR = 350) NPARAM, NFIXED
C
C.... READ FITTING PARAMETER, MAXIMUM VALUE, MINIMUM VALUE, INITIAL
C.... DELTA DURING FIT, AND FINAL DELTA DURING FIT.
C
      READ(UNINUM,*, ERR = 350) HEADNG
      READ(UNINUM,*, ERR = 350) HEADNG
      DO 100 I = 1, NPARAM
        READ (UNINUM,*, ERR = 350) DUM, FITPAR(I), XMAX(I), XMIN(I),
     1      DELTX(I), DELMN(I)
Cbyu  save the nominal values for reseting when needed.(see sub RSVPAR)
        SAVPAR(I) = FITPAR(I)
100   CONTINUE
C
      READ(UNINUM,*, ERR = 350) HEADNG
      READ(UNINUM,*, ERR = 350) HEADNG
      READ (UNINUM, *, ERR = 350) (IFIXED(I), I = 1, NFIXED)
C
C  Setup MASK array for sub STEPIT.

      DO 200 I = 1, NPARAM
        DO 150 J = 1, NFIXED
          IF (IFIXED(J) .EQ. I) THEN
            MASK(I) = 1
            GO TO 200
          ENDIF
150     CONTINUE
        MASK(I) = 0
200   CONTINUE

      GO TO 400
300   WRITE (*, '(1X/1X, 2A)') ' Error opening parameter file ', PARNAM
      PRINT*, 'Make sure it exists in the current directory...'
      STOP
350   PRINT*, ' Error in reading parameter file ', PARNAM
      STOP
400   CLOSE (UNINUM)
C
C Set guess temperature from value specified in "fmcars.par"
      TGUESS = FITPAR(5)
      CN2GUE = FITPAR(6)

      RETURN
      END

      SUBROUTINE REALOG(PHI)
C     *****  REAd the Igor's data acquisition LOG file      *****
C     *****  to obtain Equivalence Ratio (PHI) and PRESSure *****
      COMMON /PARAMS/ PRESS, YAMAX
      CHARACTER filNam*80, STRING*80
      INTEGER IERR, IFIRSC, LENTH, NFOUND, NSTART, uniNum
      LOGICAL FNDPHI, FNDPRE
      REAL PHI, PRESS, YAMAX

      DATA uniNum/15/, filNam/'C_log.txt'/

      FNDPHI = .FALSE.
      FNDPRE  = .FALSE.

10    PRINT*, 'Enter name of log file: <',
     1      filNam(IFIRSC(filNam):LENTH(filNam)), '>'
      READ(5,'(A)') STRING

      IF(STRING.NE.' ') filNam = STRING

      OPEN(UNIT = uniNum, FILE= filNam, STATUS= 'OLD', ERR= 20)
      GOTO 30
20    PRINT'(1X, 3A, / )', 'ERROR IN OPENING ', filNam(1:LENTH(filNam))
     1                 , ' - TRY AGAIN!'
      GOTO 10

30    READ(uniNum, '(A)', END=100) STRING

      NSTART = 0
      IF(INDEX(STRING, ':').GT.0) NSTART = INDEX(STRING,':') + 1
      IF(INDEX(STRING, '=').GT.0) NSTART = INDEX(STRING,'=') + 1
      IF(NSTART.EQ.0) GOTO 30

      IF(INDEX(STRING, 'Phi').GT.0) THEN
       CALL INTERP(0, STRING(NSTART:80),-1, NFOUND, Phi, IERR)
       IF(IERR.EQ.0 .AND. NFOUND.EQ.1) FNDPHI = .TRUE.
       PRINT*, 'Phi = ', Phi
      ENDIF

      IF(INDEX(STRING, 'Pressure').GT.0) THEN
       CALL INTERP(0, STRING(NSTART:80),-1, NFOUND, PRESS, IERR)
```

```
      IF(IERR.EQ.0 .AND. NFOUND.EQ.1) FNDPRE = .TRUE.
      PRINT*, 'PRESS = ', PRESS
     ENDIF

     GOTO 30

100   CONTINUE

     CLOSE(uniNum)

     IF(.NOT.FNDPHI) THEN
      PRINT*, 'Enter the Equivalence Ratio: '
      READ*, Phi
     ENDIF

     IF(.NOT.FNDPRE) THEN
      PRINT*, 'Enter the ambient Pressure (atm): '
      READ*, PRESS
     ENDIF

     RETURN
     END

     SUBROUTINE RSVPAR
C    *****  Revert to SaVed PARameters  *****
C Advisable when 'All transitions are 0' condition was run into.(see sub FITCAL)

     INTEGER NP, NPP
     PARAMETER (NP = 30, NPP = 31)
     COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
    1  DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
    2  MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
    3  KW

     COMMON /SAVPAR/ SAVPAR(NP)
     REAL SAVPAR

     REAL FITPAR, XMAX, XMIN, DELTX, DELMN, ERR, FOBJ
     INTEGER  NPARAM, NTRAC, MATRX, MASK, NFMAX, NFLAT, JVARY, NXTRA,
    1        KFLAG, NOREP, KERFL, KW
     INTEGER I

     DO 10 I = 1, NPARAM
       FITPAR(I) = SAVPAR(I)
10   CONTINUE
     RETURN
     END

     SUBROUTINE WAVLIM(CHOICE, WBCO, WECO, WBCO2, WECO2, WBN2, WEN2,
    1                  WBO2, WEO2)
C    *****  Establish WAVe number LIMits for species  *****
C    It is assumed that the CARS libraries were generated using the
C    useful spectral range over the IPDA.
     INTEGER NLP, NL
     PARAMETER (NLP = 1000, NL = 50)
     COMMON /COLIB/ NCOLIB, TCOLIB(NL), NCOSPC, XCO(NLP),
    1   YCO(NLP, NL, 2)
     COMMON /CO2LIB/ NCO2LB, TCO2LB(NL), NCO2SP, XCO2(NLP),
    1   YCO2(NLP, NL, 2)
     COMMON /N2LIB/ NN2LIB, TN2LIB(NL), NN2SPC, XN2(NLP),
    1   YN2(NLP, NL, 2)
     COMMON /O2LIB/ NO2LIB, TO2LIB(NL), NO2SPC, XO2(NLP),
    1   YO2(NLP, NL, 2)

     CHARACTER*3 CHOICE
     INTEGER NCOLIB, NCOSPC, NCO2LB, NCO2SP, NN2LIB, NN2SPC,
    1       NO2LIB, NO2SPC
     REAL TCOLIB, XCO, YCO, TCO2LB, XCO2, YCO2, TN2LIB, XN2, YN2,
    1    TO2LIB, XO2, YO2
     REAL WBCO, WECO, WBCO2, WECO2, WBN2, WEN2, WBO2, WEO2

     IF(CHOICE.EQ.'CO '.OR.CHOICE.EQ.'ALL') THEN
      WBCO = XCO(1)
      WECO = XCO(NCOSPC)
     ENDIF

     IF(CHOICE.EQ.'CO2'.OR.CHOICE.EQ.'ALL') THEN
      WBCO2 = XCO2(1)
      WECO2 = XCO2(NCO2SP)
     ENDIF

     IF(CHOICE.EQ.'O2 '.OR.CHOICE.EQ.'ALL') THEN
      WBO2 = XO2(1)
      WEO2 = XO2(NO2SPC)
     ENDIF

     IF(CHOICE.EQ.'N2 '.OR.CHOICE.EQ.'ALL' .OR. CHOICE.EQ.'DEF') THEN
      WBN2 = XN2(1)
      WEN2 = XN2(NN2SPC)
     ENDIF
```

```
      RETURN
      END

      SUBROUTINE WRIPSS(PIXLRS, SICARS, datFil, NSPECT)
      INTEGER ipdaPx, uniNum
      PARAMETER(ipdaPx = 1024, uniNum = 15)
      CHARACTER*(*) datFil, ANS*1, OUTFIL*18
      INTEGER I, NSPECT, IFIRSC, LENTH, L1, L2, L3, L4
      REAL PIXLRS(ipdaPx), SICARS(ipdaPx)


      PRINT*, 'Write pre-processed single-shot to a file in FTCARS'
     1        , 'format? <N>'
      READ(5, '(A)') ANS
      IF(ANS.EQ.' ' .OR. ANS.EQ.'N' .OR. ANS.EQ.'n') RETURN

      OUTFIL = 'singleShot0000.dat'
      IF(NSPECT.LE.9999) THEN
        L1 = INT(NSPECT/1000)
        L2 = INT((NSPECT -1000*L1)/100)
        L3 = INT((NSPECT -1000*L1-100*L2)/10)
        L4 = INT(NSPECT -1000*L1-100*L2-10*L3)
        OUTFIL(11:11) = CHAR(L1+48)
        OUTFIL(12:12) = CHAR(L2+48)
        OUTFIL(13:13) = CHAR(L3+48)
        OUTFIL(14:14) = CHAR(L4+48)
       PRINT*, '   Writing to file ', OUTFIL, '...'
       PRINT*, ''
      ELSE
       PRINT*, 'NSPECT is greater than 9999: writing to ', OUTFIL
      ENDIF

      OPEN(UNIT=UNINUM, FILE = OUTFIL, STATUS='UNKNOWN')

      WRITE(UNINUM,'(1X, A, I4, 2A)') 'Single-shot #', NSPECT,
     1        ' from data file: ', datFil(IFIRSC(datFil):LENTH(datFil))

      WRITE(UNINUM,'(1X, A)') 'Wavenumber        Data         DUMMY'

C  Process and record FIT-READY average data file.
C  The chosen file format is readable by both CARSFT and FTCARS
      DO 10 I= 1, IPDAPX
       WRITE(UNINUM,'(1X, G14.8, 2X, G14.8, 2X, F3.1)')
     1        PIXLRS(I), sIcars(I), 0.0
10    CONTINUE
```



```
      CLOSE(UNINUM)

      RETURN
      END
```

DatPre

```
C  PROGRAMS written to PROCESS data files obtained via the CARS data acquisition
C  system (Warren, 1994) used in the Optics Laboratory at BYU, as of 1998.
C
C  SUBROUTINES:
C     SSDPRE
C     AvgDIn
C     VerFil
C
C  FUNCTIONS:
C     ImaPer
C     NonLin
C     SqDpNo
C
C   PROGRAMER:  Daniel V. Flores   Nov. 1998
C
C   REFERENCES:
C
C   Boyack, Kevin W., "A Study of Turbulent Nonpremixed Jet Flames of CO/N2
C         Using Coherent Anti-Stokes Raman Spectroscopy," Ph.D. Dissertation
C         Brigham Youn University, April, 1990.
C
C   Warren, David L., "Combustion Studies of a Swirled, Non-Premixed Flame
C         Using Advanced Diagnostics," M.S. Thesis, Brigham Young University,
C         April, 1994.
C-----------------------------------------------------------------------------

      SUBROUTINE SSDPRE(datFil, yBg, Idye, sIcars, nSpect, PIXLRS,
     1                WBN2, WEN2, WBCO,WECO, WBO2, WEO2, WBCO2, WECO2,
     2                 moreSp,errMes, N2PEAK, COPEAK, O2PEAK, CO2PEK)

C     ********* Single-Shot Data Pre-processing  ***********
C
C READS in one raw CARS OMA measurement at a time from a file. An example of
C      the assumed file-format is found in the TestFiles subdirectory.
C RETURNS the square root of a CARS spectrum ready to be reduced by the
C      FTCARS code from the SANDIA NAT'L LABS.
C
```

```
C  ***IMPORTANT!--> SPECTRUM implies that background noise has already been
C                 substracted from the OMA measurement.
C
C  SUBROUTINE ARGUMENTS
C   datFil = name of the file containing the raw CARS measurements
C      yBg = Contains background noise for the data in datFil.
C     Idye = Contains one dye profile spectrum.
C   sIcars = Contains the square root of a corrected CARS spectrum. This is
C            ready to be reduced by FTCARS.
C   nSpect = Number of read Spectra.  Set to zero before calling SShotI for
C            the first time to signal the OPENing of datFil.
C   moreSp = Indicates whether there are more spectra to read in.
C   errMes = Indicates if an error ocurred while reading data.
C
C  IMPORTANT LOCAL VARIABLES
C   ipdaPx = PARAMETER: Number of pixels in the OMA'S IPDA.
C     Iraw = A pixel raw intensity value.
C   Ibgfre = Contains one uncorrected CARS spectrum.
C    yPrev = Contains the previous y. Used for image persistance
C            in the OMA's IPDA.
C   N2PEAK = Peak counts of N2 background-free spectrum
C   COPEAK = Peak counts of CO background-free spectrum
C   O2PEAK = Peak counts of O2 background-free spectrum
C   CO2PEK = Peak  counts of CO2 background-free spectrum
C--------------------------------------------------------------------------

      INTEGER ipdaPx, SSuniN
      PARAMETER(ipdaPx = 1024, SSuniN = 10)
      CHARACTER*(*) datFil, label*50, errMes
      INTEGER pixel, nSampl, nSpect
      LOGICAL moreSp
      REAL Ibgfre, Icars, Iraw, sIcars(ipdaPx), Idye(ipdaPx),
     1     yPrev(ipdaPx), yBg(ipdaPx)
      REAL ImaPer, NonLin, SqDpNo
      INTEGER N2PEAK, COPEAK, O2PEAK, CO2PEK
      REAL PIXLRS(ipdaPx), WBN2, WEN2, WBCO, WECO, WBO2, WEO2, WBCO2,
     1               WECO2

      IF (nSpect.EQ.0) THEN
        OPEN(UNIT=SSuniN, FILE=datFil, STATUS='OLD', ERR=15)
        READ(SSuniN,'(A)', ERR = 20, END=25) label
        READ(SSuniN,'(A)', ERR = 20, END=25) label
        READ(SSuniN,*, ERR = 20, END=25) nSampl
        IF(nSampl.NE.1) GOTO 30
        DO 9 pixel = 1, ipdaPx
          READ(SSuniN,*, ERR = 20, END=25) Iraw
          yPrev(pixel) = Iraw  - yBg(pixel)
9       CONTINUE
        REWIND SSuniN
        READ(SSuniN,'(A)', ERR = 20, END=25) label
        READ(SSuniN,'(A)', ERR = 20, END=25) label
      ENDIF

      READ(SSuniN,*, ERR = 20, END=25) nSampl

      IF(nSampl.NE.1) GOTO 30

      nSpect = nSpect + 1
      N2PEAK = 0
      COPEAK = 0
      O2PEAK = 0
      CO2PEK = 0

      DO 10 pixel = 1, ipdaPx
        READ(SSuniN,*, ERR = 20, END=25) Iraw

        Ibgfre = Iraw  - yBg(pixel)

        Icars = NonLin(Ibgfre -  ImaPer(yPrev(pixel)))

        sIcars(pixel) = SqDpNo(Icars, Idye(pixel))

        yPrev(pixel) = Ibgfre

        IF(PIXLRS(pixel).GT.WBCO2.AND.PIXLRS(pixel).LT.WECO2
     1     .AND.Ibgfre.GT.CO2PEK) CO2PEK = NINT(Ibgfre)
        IF(PIXLRS(pixel).GT.WBO2.AND.PIXLRS(pixel).LT.WEO2
     1     .AND.Ibgfre.GT.O2PEAK) O2PEAK = NINT(Ibgfre)
        IF(PIXLRS(pixel).GT.WBCO.AND.PIXLRS(pixel).LT.WECO
     1     .AND.Ibgfre.GT.COPEAK) COPEAK = NINT(Ibgfre)
        IF(PIXLRS(pixel).GT.WBN2.AND.PIXLRS(pixel).LT.WEN2
     1     .AND.Ibgfre.GT.N2PEAK) N2PEAK = NINT(Ibgfre)

10    CONTINUE

      READ(SSuniN,'(A)', ERR = 20, END=25) label

      IF(label(1:1).EQ.'$') THEN
        moreSp = .TRUE.
        errMes = 'O.K.'
      ELSE
        moreSp = .FALSE.
```

```
      CALL VerFil(label, errMes)
     ENDIF

     GOTO 40

 15   errMes = 'Error while Opening File'
     GOTO 35

 20   errMes = 'Error while Reading File'
     GOTO 35

 25   errMes = 'Error: Incomplete File'
     GOTO 35

 30   errMes = '# of Samples not equal to 1!'

 35   moreSp = .FALSE.

 40   IF(.NOT.moreSp) CLOSE(SSuniN)
     RETURN
     END

     SUBROUTINE AvgDIn(datFil, y, errMes)
C     *********  Average Data Input ***********
C READS data files of cumulative intensities.
C
C FILES ARE PROCESSED AS FOLLOWS:
C     (1) Averaging of Background-noise files (".bg" extension in datFil).
C              (Only averaging is performed)
C     (2) Averaging and correction for IPDA non-linearity of Average files
C         whose background has already been substracted by the OMA.
C         Examples are:
C            -- Dye Profile files (".dye" extension in datFil)
C            -- Averaged measurements of Cold Air, and Sample Gases
C               (".dat" extension in datFil)
C
C--------------------------------------------------------------------------

     INTEGER ipdaPx
     PARAMETER(ipdaPx = 1024)
     CHARACTER*(*) datFil, label*50, errMes
     INTEGER pixel, nSampl, uniNum
     LOGICAL proDat
     REAL NonLin, y(ipdaPx)
     DATA uniNum /15/

     OPEN(UNIT=uniNum, FILE=datFil, STATUS='OLD', ERR=15)
     READ(uniNum,'(A)', ERR = 20, END=25) label
     READ(uniNum,'(A)', ERR = 20, END=25) label
     READ(uniNum, *, ERR = 20, END=25) nSampl

     IF(nSampl.LT.2) GOTO 16

     IF(INDEX(datFil,'.bg').GT.0) THEN
       proDat = .FALSE.
     ELSE
       proDat = .TRUE.
     ENDIF

     DO 10 pixel = 1, ipdaPx
       READ(uniNum,*, ERR = 20, END=25) y(pixel)
       y(pixel) = y(pixel)/nSampl
       IF(proDat) y(pixel) = NonLin(y(pixel))
 10   CONTINUE

     READ(uniNum,'(A)', ERR = 20, END=25) label
     CALL VerFil(label, errMes)
     GOTO 30

 15   errMes = 'Error while Opening File'
     GOTO 30

 16   errMes = '# of Samples < 2: Single-shot data?'
     GOTO 30

 20   errMes = 'Error while Reading File'
     GOTO 30

 25   errMes = 'Error: Incomplete File'

 30   CLOSE(uniNum)
     RETURN
     END

     SUBROUTINE VerFil(label, errMes)
C     ********* Subroutine: Verify the end of a data File ***********
C--------------------------------------------------------------------------

     CHARACTER*(*) label, errMes

     IF(label(1:1).EQ.'$') THEN
       errMes = 'More Shots in File'
```

```
      ELSEIF(label(1:1).EQ.'@') THEN
        errMes = 'O.K.'
      ELSE
        errMes = 'End of File problem'
      ENDIF

      RETURN
      END

      FUNCTION ImaPer(Iprev)
C     ******** Correct for Image Persistance of the OMA's IPDA *********
C     Data and algorithm taken from FTDNLL source code (Boyack, 1990)
C          (see Appendix in Boyack's dissertation: SUBROUTINE DATGET)
C-----------------------------------------------------------------------

      REAL cons1, cons2, Imaper, Iprev
      DATA cons1, cons2/0.030095, 0.80575/

      IF (Iprev.GE.0) THEN
        Imaper = cons1*(Iprev**cons2)
      ELSE
        Imaper = -cons1*((-Iprev)**cons2)
      ENDIF

      RETURN
      END

      FUNCTION NonLin(Imeas)
C     ******** Correct for OMA's IPDA Non-Linearity  ************
C     Data and algorithm taken from FTDNLL source code (Boyack, 1990)
C          (see Appendix in Boyack's dissertation: SUBROUTINE DATGET)
C-----------------------------------------------------------------------

      REAL cons1, cons2, NonLin, Imeas
      DATA cons1, cons2 /0.60432, 0.04765/

      IF(Imeas.GT. 0.0 .AND. Imeas.LT.1.0) THEN
        NonLin = Imeas/cons1
      ELSEIF(Imeas.GE.1.0) THEN
        NonLin = Imeas/(cons1 + cons2*LOG(Imeas))
      ELSE
        NonLin = Imeas
      ENDIF

      RETURN
      END
```

```
      FUNCTION SqDpNo(Icars, Idye)
C     ** Square root and Dye-Profile normalization of a CARS intensity datum **
C     Algorithm taken from FTDNLL source code (Boyack, 1990)
C          (see Appendix in Boyack's dissertation: SUBROUTINE DATGET)
C-----------------------------------------------------------------------

      REAL ratio, SqDpNo, Icars, Idye

      IF(Idye.NE.0.0) THEN
        ratio = Icars/Idye
      ELSE
        ratio = Icars
      ENDIF

      IF(ratio.GE. 0.0) THEN
        SqDpNo = SQRT(ratio)
      ELSE
        SqDpNo = -SQRT(-ratio)
      ENDIF

      RETURN
      END
```

F4CSTX

```
      SUBROUTINE F4CSTX (WBEG, WEND, NDAT, XDATA, YDATIN, SPENAM, FIXT,
     1              TFIT,  CHNRXI, ARERAT, FITFLG, LTRAN0, SDIFX2)
Cbyu  ****  Fit 4-species' Cars Spectra for Temperature and mole fraction X
Cbyu  This program is a modified version of the FTCARS program from SANDIA
Cbyu  National Labs. (see last version information below).
Cbyu  F4CSTX interprets CARS spectra of N2, CO, O2 and CO2 and returns:
Cbyu    Temperature (K):  Optionally fitted or held constant as that from N2
Cbyu    CHINR: Ratio of total CHINR to specie mole fraction (see in-code docu-
Cbyu           mentation)
Cbyu  Programmer:  Daniel V. Flores
Cbyu  Date:  Feb.,  , 1999
Cbyu  INPUT:
Cbyu       WBEG   :  Begining wave number to include in fit.
Cbyu       WEND   :  Ending wave number to include in fit.
Cbyu       NDAT   :  Number of data points in data arrays YDATIN and XDATA
Cbyu       XDATA  :  Raman Shift (cm-1) corresponding to array points in YDATIN
Cbyu       YDATIN :  Preprocessed array of CARS data intensities
Cbyu                 (see DATPRE subroutine)
Cbyu       FIXT   :  Logical variable = .TRUE. if temperature is to remain
Cbyu                 fixed during least-squares fit.
Cbyu       SPENAM :  Name of gas whose CARS spectra is to be interpreted.
Cbyu  OUTPUT:
Cbyu       TFIT   :  Best-fit value of temperature (K)
Cbyu       CHNRXI :  Best-fit value of the Ratio of total nonresonant suscept-
Cbyu                 ibility (at STP) of gas sample to the mole fraction of
Cbyu                 the specie I. (in units of 1E-18 cc/ergs)
Cbyu       ARERAT :  RATio of the AREas under theoretical and data curves.
Cbyu       FITFLG :  KFLAG value from stepit, greater than 0 if normal exit.
Cbyu       LTRAN0 :  Logical variable = .TRUE. if All transitions are 0 in
Cbyu                 any specie's CHI subroutine. Its value is set using LZERO
Cbyu                 (see COMMON /L0/)
Cbyu       SDIFX2 : Sum of the square of differences between thoery and data
Cbyu  Notes from FTCARS start here:
C     DATE REVISED:  07/13/2001
C
C    This program was issued by Sandia Laboratories, a prime contractor
C    to the United States Department of Energy.  Neither the United
C    States, nor the United States Department of Energy, nor any of
C    their employees, nor any other their contractors, subcontractors,
C    nor their employees, make any warranty, express or implied, or
C    assumes any legal liability or responsibility for the accuracy,
C    completeness, or usefulness of any information, apparatus,
C    product or process disclosed, or represents that its use would
C    not infringe privately owned rights.
C
C***********************************************************************
C     CREATE THE CARS LIBRARY SPECTRA AS DESCRIBED IN SUBROUTINE
C     LIBSPC.
C***********************************************************************
      PARAMETER (NP = 30, NPP = 31, NLP = 1000, NL = 50, ND = 1024)
      COMMON /PARAMS/ PRESS, YAMAX
      COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
     1  DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
     2  MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
     3  KW
      COMMON /FITINF/ NPTS,  YDATA(ND), SHOTNS
      COMMON /FTPNTS/ XDATA1(ND), XPTS(ND), X1COMM
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
      COMMON /FITSAV/ XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV
      COMMON /PLOTN/ Y1D(ND), Y2D(ND), Y3D(ND), XPLZER(2), PLTZER(2),
     1  PLTOFF(2), AREATH, AREADT, DATMAX
      COMMON /SPECIE/ ITSNAM
      COMMON /L0/ LZERO
C
      DIMENSION XDATA(ND), YDATIN(ND), YCAL(ND)
C
      LOGICAL LTRAN0, FIXT, SHOTNS, LZERO
      CHARACTER*3 ITSNAM, SPENAM
      INTEGER FITFLG
      REAL SDIFX2
D     CHARACTER*80 ITITLE, ANS*1
D     CHARACTER*14 VARS(20)
D     DATA VARS/ 'Horiz. shift', 'Horiz. expan.', 'Vert. shift',
D    1   'Intens. exp.', 'Temperature', 'Nonres. susc.', 14*' ' /
D     DATA ITERM, ITMVT, ITMTK, ILN03 / 0, 5, 1, 10/


CbyuBeg Set values of parameters to do fit as desired...
      NTRAC = -1
      MATRX = 0
      NFMAX = 1500
      KW = 6
      ITSNAM = SPENAM
      LTRAN0 = .FALSE.
      LZERO = .FALSE.

Cbyu  Always use Shot-noise statistics while fitting experimental spectra
      SHOTNS = .TRUE.
```

```
Cbyu  Set Temperature and nonresonant susceptibility guess values
Cbyu  (if FIXT = .TRUE., this Temperature value will remain fixed during fit.)
      FITPAR(5) = TFIT
      FITPAR(6) = CHNRXI
      IF(FIXT) THEN
        MASK(5) = 1
      ELSE
        MASK(5) = 0
      ENDIF

CbyuEnd

      CALL CPYPAR (FITPAR, XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
      CALL WSET (WBEG, WEND, NDAT, XDATA, XOFF, XEXPND, IMIN,
     1   IMAX, NPTS)
C
C.... COMPACT (X, Y) DATA POINTS.
Cbyu  i.e., extract appropriate spectral range from IPDA array intensity data.
C
      CALL DATCAL (XDATA, YDATIN, IMIN, IMAX, XDATA1, YDATA)
      X1COMM = XDATA1(1)
C
C.... MUST RECALL NORM FOR NONRATIOED DATA TO CORRECT ORIGINAL
C.... NORMALIZATION IF ORIGINAL MAXIMUM DATA POINT IS NOT INCLUDED
C.... IN NEW WAVELENGTH LIMITS.
C
      CALL NORM (NPTS, XDATA1, YDATA, DATMAX)
      IF(DATMAX.LE.0) THEN
C Skip fitting this spectrum: consider it an abnormal exit from STEPIT
        FITFLG = -1
        RETURN
      ENDIF

      CALL
     1   FITRUN (XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV,
     2   XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
      CALL INISAV (XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV)
C
Cbyu  Calculate the CHI curve for the species of interest.
      IF( ITSNAM.EQ.'CO ') THEN
        CALL COCHI (YCAL, 0)
      ELSE IF( ITSNAM.EQ.'CO2') THEN
        CALL CO2CHI (YCAL, 0)
      ELSE IF( ITSNAM.EQ.'N2 ') THEN
        CALL N2CHI (YCAL, 0)
      ELSE IF( ITSNAM.EQ.'O2 ') THEN
        CALL O2CHI (YCAL, 0)
      ENDIF
C
C.... RESCALE DATA AND THEORY.
C
      CALL RESCAL (NPTS, YDATA, YCAL, Y1D, Y2D, Y3D,
     1   YAMAX, AMPL, XPTS, AREATH, AREADT, YOFF)

      TFIT = TEMP
      CHNRXI = CHINR
      FITFLG = KFLAG
      ARERAT = AREATH/AREADT
      SDIFX2 = FOBJ
      IF(LZERO) LTRAN0 = .TRUE.

D     ITITLE = ITSNAM(1:LENTH(ITSNAM))//' FIT RESULTS'
D     IF(ITSNAM.NE.'CO2') THEN
D     PRINT'(2X, (3A))', '*****************   ', ITITLE(1:LENTH(ITITLE))
D    1              , '   *****************'
D     ELSE
D     PRINT'(2X, (3A))', '*****************   ', ITITLE(1:LENTH(ITITLE))
D    1              , '  *****************'
D     ENDIF
D     WRITE (*, '(1X, A, I6, A, I6, A, I6, A/)')
D    1   ' Data points kept:  IMIN = ', IMIN, ', IMAX = ', IMAX,
D    2   ' (', NPTS, ' data points)'
D     CALL PRNPAR (ITERM, ITMVT, ITMTK, 'NO', NPARAM,
D    1   MASK, FITPAR, WBEG, WEND, PRESS, AREATH, AREADT,
D    2   .TRUE., KFLAG, NOREP, KERFL, FOBJ, 'NO', 'NO', VARS,
D    3   .FALSE., YAMAX, AMPL, DATMAX)

D     IF(.NOT.LZERO) THEN
C  Otherwise garbage will go to the TEK term
c     PRINT*,'Plot Theory vs Data? <Y>'
c      READ(5,'(A)') ANS
c      IF(ANS.EQ.' '.OR. ANS.EQ.'Y' .OR. ANS.EQ.'y')
D        CALL PLTRUN (NPTS, .FALSE., XPTS, Y1D, Y2D, Y3D, YAMAX, WBEG,
D    2      WEND, ITITLE, ITERM, ITMVT, ITMTK, 'NO', 'NO', 'NO')
D     ELSE
D       PRINT '(1X, / A)', 'Press RETURN to continue....'
D       READ(5,'(A)')
D     ENDIF
```

```
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE COCHI (YCAL, JVARY)
C
      PARAMETER (NLP = 1000, NL = 50, ND = 1024)
      COMMON /COLIB/ NLIB, TLIB(NL), NSPECT, XSPECT(NLP),
     1   YSPECT(NLP, NL, 2)
      COMMON /PARAMS/ PRESS, YAMAX
      COMMON /FITINF/ NPTS, YDATA(ND), SHOTNS
      COMMON /FTPNTS/ XDATA1(ND), XPTS(ND), X1COMM
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
      COMMON /FITSAV/ XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV
      COMMON /L0/ LZERO
      LOGICAL LZERO
C
      DIMENSION YCAL(*), SQTCHI(ND), CHISAV(ND), YKEEP(ND),
     1   DIFOLD(ND)
      LOGICAL NWXOFF, NWXEXP, NWYOFF, NEWTEM, NEWCHI, NWCHNR,
     1   SHOTNS
      SAVE CHISAV, XSAVE, XSAVEX
C
      IF(CHINR.EQ.0.0) RETURN

      NWXOFF = (XOFF .NE. XOFFSV)
      NWXEXP = (XEXPND .NE. XEXPSV)
      NWYOFF = (YOFF .NE. YOFFSV)
      NEWTEM = (TEMP .NE. TEMPSV)
      NWCHNR = (CHINR .NE. CHNRSV)
      NEWCHI = (NWXOFF .OR. NWXEXP .OR. NWYOFF .OR. NEWTEM .OR.
     1         NWCHNR)
C
      IF (NWXOFF .OR. NWXEXP) THEN
C
        DO 200 I = 1, NPTS
          DIFOLD(I) = XDATA1(I)-X1COMM
          XPTS(I) = X1COMM+XOFF+DIFOLD(I)*XEXPND
200     CONTINUE
C
      ENDIF
C
C.... CALCULATE SUSCEPTIBILITY.
C
      IF ((JVARY .EQ. 1) .OR. (JVARY .EQ. 2)) THEN
        CALL CHIDEL (NPTS, XPTS, XOFF-XSAVE, XEXPND-XSAVEX, DIFOLD,
     1      CHISAV, SQTCHI)
      ELSEIF (NEWCHI) THEN
        CALL CHICAL (NEWTEM, PARPR, NLIB, TLIB, NSPECT,
     1      XSPECT, YSPECT, PRESS, YOFF, TEMP, CHINR, XPTS, NPTS,
     2      SQTCHI)
      ENDIF
C
      IF (JVARY .EQ. 0) THEN
C
        DO 500 I = 1, NPTS
          CHISAV(I) = SQTCHI(I)
500     CONTINUE
C
        XSAVE = XOFF
        XSAVEX = XEXPND
      ENDIF
C
      YAMAX = 0.
      YKEEP(1) = SQTCHI(1)
      IF (YKEEP(1) .GT. 0.) YAMAX = YKEEP(1)
C
      DO 800 I = 2, NPTS
        YKEEP(I) = SQTCHI(I)
        IF (YKEEP(I) .GT. YAMAX) YAMAX = YKEEP(I)
800   CONTINUE
C
      IF (YAMAX .EQ. 0.) THEN
D        IF(.NOT.LZERO)
D    1     PRINT *, ' All transition amplitudes are 0 in COCHI'
Cbyu     STOP
        LZERO = .TRUE.
        RETURN
      ENDIF
C
      DO 900 I = 1, NPTS
        YCAL(I) = YKEEP(I)/(AMPL*YAMAX)
900   CONTINUE
C
      XOFFSV = XOFF
      XEXPSV = XEXPND
      YOFFSV = YOFF
      TEMPSV = TEMP
      CHNRSV = CHINR
C
```

```
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE CO2CHI(YCAL, JVARY)
C
      PARAMETER (NLP = 1000, NL = 50, ND = 1024)
      COMMON /CO2LIB/ NLIB, TLIB(NL), NSPECT, XSPECT(NLP),
     1   YSPECT(NLP, NL, 2)
      COMMON /PARAMS/ PRESS, YAMAX
      COMMON /FITINF/ NPTS, YDATA(ND), SHOTNS
      COMMON /FTPNTS/ XDATA1(ND), XPTS(ND), X1COMM
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
      COMMON /FITSAV/ XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV
      COMMON /L0/ LZERO
      LOGICAL LZERO
C
      DIMENSION YCAL(*), SQTCHI(ND), CHISAV(ND), YKEEP(ND),
     1   DIFOLD(ND)
      LOGICAL NWXOFF, NWXEXP, NWYOFF, NEWTEM, NEWCHI, NWCHNR,
     1   SHOTNS
      SAVE CHISAV, XSAVE, XSAVEX
C
      IF(CHINR.EQ.0.0) RETURN

      NWXOFF = (XOFF .NE. XOFFSV)
      NWXEXP = (XEXPND .NE. XEXPSV)
      NWYOFF = (YOFF .NE. YOFFSV)
      NEWTEM = (TEMP .NE. TEMPSV)
      NWCHNR = (CHINR .NE. CHNRSV)
      NEWCHI = (NWXOFF .OR. NWXEXP .OR. NWYOFF .OR. NEWTEM .OR.
     1         NWCHNR)
C
      IF (NWXOFF .OR. NWXEXP) THEN
C
        DO 200 I = 1, NPTS
          DIFOLD(I) = XDATA1(I)-X1COMM
          XPTS(I) = X1COMM+XOFF+DIFOLD(I)*XEXPND
200     CONTINUE
C
      ENDIF
C
C.... CALCULATE SUSCEPTIBILITY.
C
      IF ((JVARY .EQ. 1) .OR. (JVARY .EQ. 2)) THEN
        CALL CHIDEL (NPTS, XPTS, XOFF-XSAVE, XEXPND-XSAVEX, DIFOLD,
     1      CHISAV, SQTCHI)
      ELSEIF (NEWCHI) THEN
        CALL CHICAL (NEWTEM, PARPR, NLIB, TLIB, NSPECT,
     1      XSPECT, YSPECT, PRESS, YOFF, TEMP, CHINR, XPTS, NPTS,
     2      SQTCHI)
      ENDIF
C
      IF (JVARY .EQ. 0) THEN
C
        DO 500 I = 1, NPTS
          CHISAV(I) = SQTCHI(I)
500     CONTINUE
C
        XSAVE = XOFF
        XSAVEX = XEXPND
      ENDIF
C
      YAMAX = 0.
      YKEEP(1) = SQTCHI(1)
      IF (YKEEP(1) .GT. 0.) YAMAX = YKEEP(1)
C
      DO 800 I = 2, NPTS
        YKEEP(I) = SQTCHI(I)
        IF (YKEEP(I) .GT. YAMAX) YAMAX = YKEEP(I)
800   CONTINUE
C
      IF (YAMAX .EQ. 0.) THEN
D       IF(.NOT.LZERO)
D    1     PRINT *, ' All transition amplitudes are 0 in CO2CHI'
Cbyu    STOP
        LZERO = .TRUE.
        RETURN
      ENDIF
C
      DO 900 I = 1, NPTS
        YCAL(I) = YKEEP(I)/(AMPL*YAMAX)
900   CONTINUE
C
      XOFFSV = XOFF
      XEXPSV = XEXPND
      YOFFSV = YOFF
      TEMPSV = TEMP
      CHNRSV = CHINR
C
```

```
      RETURN
      END
C
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C
      SUBROUTINE N2CHI(YCAL, JVARY)
C
      PARAMETER (NLP = 1000, NL = 50, ND = 1024)
      COMMON /N2LIB/ NLIB, TLIB(NL), NSPECT, XSPECT(NLP),
     1   YSPECT(NLP, NL, 2)
      COMMON /PARAMS/ PRESS, YAMAX
      COMMON /FITINF/ NPTS, YDATA(ND), SHOTNS
      COMMON /FTPNTS/ XDATA1(ND), XPTS(ND), X1COMM
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
      COMMON /FITSAV/ XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV
      COMMON /L0/ LZERO
      LOGICAL LZERO
C
      DIMENSION YCAL(*), SQTCHI(ND), CHISAV(ND), YKEEP(ND),
     1   DIFOLD(ND)
      LOGICAL NWXOFF, NWXEXP, NWYOFF, NEWTEM, NEWCHI, NWCHNR,
     1   SHOTNS
      SAVE CHISAV, XSAVE, XSAVEX
C
      IF(CHINR.EQ.0.0) RETURN

      NWXOFF = (XOFF .NE. XOFFSV)
      NWXEXP = (XEXPND .NE. XEXPSV)
      NWYOFF = (YOFF .NE. YOFFSV)
      NEWTEM = (TEMP .NE. TEMPSV)
      NWCHNR = (CHINR .NE. CHNRSV)
      NEWCHI = (NWXOFF .OR. NWXEXP .OR. NWYOFF .OR. NEWTEM .OR.
     1        NWCHNR)
C
      IF (NWXOFF .OR. NWXEXP) THEN
C
        DO 200 I = 1, NPTS
          DIFOLD(I) = XDATA1(I)-X1COMM
          XPTS(I) = X1COMM+XOFF+DIFOLD(I)*XEXPND
200     CONTINUE
C
      ENDIF
C
C.... CALCULATE SUSCEPTIBILITY.
C
```

```
      IF ((JVARY .EQ. 1) .OR. (JVARY .EQ. 2)) THEN
        CALL CHIDEL (NPTS, XPTS, XOFF-XSAVE, XEXPND-XSAVEX, DIFOLD,
     1      CHISAV, SQTCHI)
      ELSEIF (NEWCHI) THEN
        CALL CHICAL (NEWTEM, PARPR, NLIB, TLIB, NSPECT,
     1      XSPECT, YSPECT, PRESS, YOFF, TEMP, CHINR, XPTS, NPTS,
     2      SQTCHI)
      ENDIF
C
      IF (JVARY .EQ. 0) THEN
C
        DO 500 I = 1, NPTS
          CHISAV(I) = SQTCHI(I)
500     CONTINUE
C
        XSAVE = XOFF
        XSAVEX = XEXPND
      ENDIF
C
      YAMAX = 0.
      YKEEP(1) = SQTCHI(1)
      IF (YKEEP(1) .GT. 0.) YAMAX = YKEEP(1)
C
      DO 800 I = 2, NPTS
        YKEEP(I) = SQTCHI(I)
        IF (YKEEP(I) .GT. YAMAX) YAMAX = YKEEP(I)
800   CONTINUE
C
      IF (YAMAX .EQ. 0.) THEN
D       IF(.NOT.LZERO)
D    1    PRINT *, ' All transition amplitudes are 0 in N2CHI'
Cbyu    STOP
        LZERO = .TRUE.
        RETURN
      ENDIF
C
      DO 900 I = 1, NPTS
        YCAL(I) = YKEEP(I)/(AMPL*YAMAX)
900   CONTINUE
C
      XOFFSV = XOFF
      XEXPSV = XEXPND
      YOFFSV = YOFF
      TEMPSV = TEMP
      CHNRSV = CHINR
C
```

```
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE O2CHI(YCAL, JVARY)
C
      PARAMETER (NLP = 1000, NL = 50, ND = 1024)
      COMMON /O2LIB/ NLIB, TLIB(NL), NSPECT, XSPECT(NLP),
     1   YSPECT(NLP, NL, 2)
      COMMON /PARAMS/ PRESS, YAMAX
      COMMON /FITINF/ NPTS, YDATA(ND), SHOTNS
      COMMON /FTPNTS/ XDATA1(ND), XPTS(ND), X1COMM
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
      COMMON /FITSAV/ XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV
      COMMON /L0/ LZERO
      LOGICAL LZERO
C
      DIMENSION YCAL(*), SQTCHI(ND), CHISAV(ND), YKEEP(ND),
     1   DIFOLD(ND)
      LOGICAL NWXOFF, NWXEXP, NWYOFF, NEWTEM, NEWCHI, NWCHNR,
     1   SHOTNS
      SAVE CHISAV, XSAVE, XSAVEX
C
      IF(CHINR.EQ.0.0) RETURN

      NWXOFF = (XOFF .NE. XOFFSV)
      NWXEXP = (XEXPND .NE. XEXPSV)
      NWYOFF = (YOFF .NE. YOFFSV)
      NEWTEM = (TEMP .NE. TEMPSV)
      NWCHNR = (CHINR .NE. CHNRSV)
      NEWCHI = (NWXOFF .OR. NWXEXP .OR. NWYOFF .OR. NEWTEM .OR.
     1        NWCHNR)
C
      IF (NWXOFF .OR. NWXEXP) THEN
C
        DO 200 I = 1, NPTS
          DIFOLD(I) = XDATA1(I)-X1COMM
          XPTS(I) = X1COMM+XOFF+DIFOLD(I)*XEXPND
200     CONTINUE
C
      ENDIF
C
C.... CALCULATE SUSCEPTIBILITY.
C
      IF ((JVARY .EQ. 1) .OR. (JVARY .EQ. 2)) THEN
        CALL CHIDEL (NPTS, XPTS, XOFF-XSAVE, XEXPND-XSAVEX, DIFOLD,
     1     CHISAV, SQTCHI)
      ELSEIF (NEWCHI) THEN
        CALL CHICAL (NEWTEM, PARPR, NLIB, TLIB, NSPECT,
     1     XSPECT, YSPECT, PRESS, YOFF, TEMP, CHINR, XPTS, NPTS,
     2     SQTCHI)
      ENDIF
C
      IF (JVARY .EQ. 0) THEN
C
        DO 500 I = 1, NPTS
          CHISAV(I) = SQTCHI(I)
500     CONTINUE
C
        XSAVE = XOFF
        XSAVEX = XEXPND
      ENDIF
C
      YAMAX = 0.
      YKEEP(1) = SQTCHI(1)
      IF (YKEEP(1) .GT. 0.) YAMAX = YKEEP(1)
C
      DO 800 I = 2, NPTS
        YKEEP(I) = SQTCHI(I)
        IF (YKEEP(I) .GT. YAMAX) YAMAX = YKEEP(I)
800   CONTINUE
C
      IF (YAMAX .EQ. 0.) THEN
D       IF(.NOT.LZERO)
D    1     PRINT *, ' All transition amplitudes are 0 in O2CHI'
Cbyu     STOP
        LZERO = .TRUE.
        RETURN
      ENDIF
C
      DO 900 I = 1, NPTS
        YCAL(I) = YKEEP(I)/(AMPL*YAMAX)
900   CONTINUE
C
      XOFFSV = XOFF
      XEXPSV = XEXPND
      YOFFSV = YOFF
      TEMPSV = TEMP
      CHNRSV = CHINR
C
```

```
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE CHICAL (NEWTEM, PARPR, NLIB, TLIB,
     1   NSPECT, XSPECT, YSPECT, PRESS, YOFF, TEMP, CHINR, XPTS, NPTS,
     2   SQTCHI)
C
C.... THE INTERPOLATION ROUTINE USED HERE TO FIT TEMPERATURE AND
C.... NONRESONANT SUSCEPTIBILITY IS BASED ON THE PAPER R. J. HALL AND
C.... L. R. BOEDEKER, APPL. OPT. 23, 1340 (1984).  NOTE THAT THEY USE
C.... THE RATIO OF THE MOLE FRACTION OF THE RESONANT SPECIES TO THE
C.... OVERALL NONRESONANT SUSCEPTIBILITY.  HENCE, WHEN GENERATING
C.... LIBRARY SPECTRA WITH CODES SUCH AS CARSFT, THE PARTIAL PRESSURE
C.... OF THE RESONANT SPECIES OF INTEREST SHOULD BE SET EQUAL TO 1.
C.... (HALL AND BODECKER'S MODEL IS INCONVENIENT FOR TREATING
MULTIPLE
C.... RESONANT SPECIES, SO THAT THE PARTIAL PRESSURE OF OTHER
C.... RESONANT GASES SHOULD BE SET TO ZERO WHEN GENERATING LIBRARY
C.... SPECTRA; THE NONRESONANT SUSCEPTIBILITY OF OTHER GASES WILL
C.... BE INCLUDED IN THE OVERALL NONRESONANT SUSCEPTIBILITY.)  ALSO,
C.... VERTICAL OFFSETS OR SHIFTS CAN BE INCLUDED ONLY IF THE
C.... NONRESONANT SUSCEPTIBILITY IN FTCARS IS SET TO ZERO AND
C.... ACCOUNTED FOR DIRECTLY IN THE LIBRARY SPECTRA CALCULATIONS.
C.... FITTING VARIABLES ARE:
C....    XOFF - HORIZONTAL SHIFT
C....    XEXPND - HORIZONTAL EXPANSION
C....    YOFF - VERTICAL SHIFT
C....    AMPL - INTENSITY EXPANSION (OPERATES ON DATA)
C....    TEMP - TEMPERATURE IN KELVINS
C....    CHINR  - NONRESONANT SUSCEPTIBILITY AT STP
C
      PARAMETER (NLP = 1000, NL = 50, ND = 1024)
      DIMENSION SAVCHI(ND, 2), PARPR(*), XPTS(*), TLIB(*),
     1   XSPECT(*), YSPECT(NLP, NL, *), SAVSUS(ND), SQTCHI(*)
      LOGICAL NEWTEM
      SAVE SAVCHI, SAVSUS
      DATA T0 / 273.15 /
C
C.... CORRECT CHINR FOR TEMPERATURE AND PRESSURE.  NOTE THAT QUOTED
C.... VALUES ARE TYPICALLY FOR STP.  ALSO NOTE THAT CORRECTIONS
SHOULD
C.... BE MADE FOR NONZERO ANGLES BETWEEN PUMP, PROBE, AND ANALYZER
C.... (FACTOR TNRES IN SUBROUTINE NONRES IN CARSFT); IF THESE ANGLES
C.... ARE NONZERO, FITTED RESULT FOR CHINR MUST BE ADJUSTED BY HAND
C.... ACCORDINGLY.  CALCULATION ASSUMES MOLE FRACTION OF RESONANT
SPECIES
C.... IS UNITY.  TO DETERMINE ACTUAL MOLE FRACTION, DIVIDE KNOWN
VALUE OF
C.... NONRESONANT SUSCEPTIBILITY OF GAS MIXTURE AT STP BY FITTED
VALUE AT
C.... STP (I.E., CHINR).  UNCERTAINTIES IN MOLE FRACTION CAN BE LARGE,
C.... ESPECIALLY IF RESONANT SUSCEPTIBILITY IS MUCH LARGER THAN
NONRESONANT
C.... BACKGROUND.
C
      CHIUSE = CHINR*PRESS*(T0/TEMP)
C
      IF (NEWTEM) THEN
        IL = 1
        CALL SEARCH (NLIB, TLIB, TEMP, IL, IER)
C
        DO 100 I = 1, NSPECT
          TFAC = (TEMP-TLIB(IL))/(TLIB(IL+1)-TLIB(IL))
          SAVCHI(I, 1) = TFAC*(YSPECT(I, IL+1, 1)-YSPECT(I, IL, 1))+
     1             YSPECT(I, IL, 1)
          SAVCHI(I, 2) = TFAC*(YSPECT(I, IL+1, 2)-YSPECT(I, IL, 2))+
     1             YSPECT(I, IL, 2)
100     CONTINUE
C
      ENDIF
C
CSPON REPLACE THE FIRST LINE IN THE DO LOOP WITH SAVCHI(I, 2)+YOFF.
CSPON COMMENT OUT THE IF..ELSE..ENDIF STATEMENT IN THE DO LOOP.
C
      DO 200 I = 1, NSPECT
        SAVSUS(I) = CHIUSE**2+2.*CHIUSE*SAVCHI(I, 1)+
     1           SAVCHI(I, 2)+YOFF*ABS(YOFF)
        IF (SAVSUS(I) .LT. 0.) THEN
          SAVSUS(I) = 0.
        ELSE
          SAVSUS(I) = SQRT(SAVSUS(I))
        ENDIF
200   CONTINUE
C
      DO 500 I = 1, NPTS
        CALL TERPOL (NSPECT, XSPECT, SAVSUS, XPTS(I), SQTCHI(I), IER)
500   CONTINUE
C
      RETURN
```

```
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE CHIDEL (NPTS, XPTS, XDEL, XDELX, DIFOLD, CHISAV,
     1   SQTCHI)
C
C.... THIS SUBROUTINE CALCULATES THE CHANGE IN SQTCHI WHEN XOFF OR
C.... XEPXND IS VARIED.
C
      DIMENSION XPTS(*), DIFOLD(*), CHISAV(*), SQTCHI(*)
C
      DO 100 I = 1, NPTS-1
        SLOPE = (CHISAV(I+1)-CHISAV(I))/(XPTS(I+1)-XPTS(I))
        SQTCHI(I) = CHISAV(I)+SLOPE*(XDEL+XDELX*DIFOLD(I))
        IF (SQTCHI(I) .LT. 0.) SQTCHI(I) = 0.
100   CONTINUE
C
C.... DO LAST POINT, USING SLOPE FOR SECOND TO LAST POINT.
C
      SQTCHI(NPTS) = CHISAV(NPTS)+SLOPE*XDEL
      IF (SQTCHI(NPTS) .LT. 0.) SQTCHI(NPTS) = 0.
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE CPYPAR (FITPAR, XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
C.... THIS SUBROUTINE REPLACES VALUES OF PHYSICAL PARAMETERS
C.... WITH FITTING VARIABLES.
C
      DIMENSION FITPAR(*)
C
      XOFF = FITPAR(1)
      XEXPND = FITPAR(2)
      YOFF = FITPAR(3)
      AMPL = FITPAR(4)
      TEMP = FITPAR(5)
      CHINR = FITPAR(6)
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE DATCAL (XDATA, YDATIN, IMIN, IMAX, XDATA1, YDATA)
C
      DIMENSION XDATA(*), XDATA1(*), YDATIN(*), YDATA(*)
C
C.... THIS SUBROUTINE COMPACTS DATA POINTS TO WITHIN WAVENUMBER
RANGE.
C
      ICT = 0
C
      DO 100 I = IMIN, IMAX
        ICT = ICT+1
        XDATA1(ICT) = XDATA(I)
        YDATA(ICT) = YDATIN(I)
100   CONTINUE
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE FITCAL
C
      PARAMETER (NP = 30, NPP = 31, ND = 1024)
      COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
     1   DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
     2   MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
     3   KW
      COMMON /FITINF/ NPTS, YDATA(ND), SHOTNS
      COMMON /FITPRM/ XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR
Cbyu
      COMMON /SPECIE/ ITSNAM
      COMMON /L0/ LZERO
      LOGICAL LZERO
C
      CHARACTER*3 ITSNAM
      DIMENSION YCAL(ND)
      LOGICAL SHOTNS
C
      CALL CPYPAR (FITPAR, XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
CbyuBeg
```

```
      IF( ITSNAM.EQ.'CO ') THEN
        CALL COCHI (YCAL, JVARY)
      ELSE IF( ITSNAM.EQ.'CO2') THEN
        CALL CO2CHI (YCAL, JVARY)
      ELSE IF( ITSNAM.EQ.'N2 ') THEN
        CALL N2CHI (YCAL, JVARY)
      ELSE IF( ITSNAM.EQ.'O2 ') THEN
        CALL O2CHI (YCAL, JVARY)
      ELSE
        PRINT*, 'THE SPECIE ', ITSNAM, ' IS NOT CURRENTLY SUPPORTED...'
        STOP
      ENDIF
CbyuEnd

      FOBJ = 0.
Cbyu  LZERO is set to .TRUE. in the CHI subroutines when All transtions are 0.
Cbyu  Quit doing CHI computations through the rest of times stepit calls FITCAL.
      IF(LZERO) RETURN
C
C.... USE SHOT-NOISE STATISTICS AND ASSUME VARIANCE IS EQUAL TO
C.... SQUARE ROOT OF DATA.
C
      IF (SHOTNS) THEN
        DO 100 K = 1, NPTS
          FO = YCAL(K)-YDATA(K)
Cbyu Account for YDATA(K) < or = zero. This treatment is similar to the one
Cbyu found in CARSFT.F, but differs in that negative data values are not used
Cbyu in the shot-noise correction.
          IF(YDATA(K).GT.0) THEN
            FOBJ = FOBJ+FO*FO/YDATA(K)
          ELSE
            FOBJ = FOBJ+FO*FO
          ENDIF
100     CONTINUE
      ELSE
        DO 200 K = 1, NPTS
          FO = YCAL(K)-YDATA(K)
          FOBJ = FOBJ+FO*FO
200     CONTINUE
      ENDIF
C
      IF (NTRAC .EQ. 0) THEN
        IF (JVARY .EQ. 0) THEN
          WRITE (*, '(1X, A, E10.4)')
     1         ' STEPIT varying all free fit parameters, X2 = ', FOBJ
        ELSE
          WRITE (*, '(1X, A, I3, A, E12.6, A, E10.4)')
     1         ' STEPIT varying parameter #', JVARY, ', value = ',
     2         FITPAR(JVARY), ', X2 = ',FOBJ
        ENDIF
      ENDIF
C
      RETURN
      END
C
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C
      SUBROUTINE FITRUN (XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV,
     1  XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
C.... THIS SUBROUTINE CALLS T HE FIT ROUTINE.
C
      PARAMETER (NP = 30, NPP = 31)
      COMMON /CSTEP/ FITPAR(NP), XMAX(NP), XMIN(NP), DELTX(NP),
     1  DELMN(NP), ERR(NP, NPP), FOBJ, NPARAM, NTRAC, MATRX,
     2  MASK(NP), NFMAX, NFLAT, JVARY, NXTRA, KFLAG, NOREP, KERFL,
     3  KW
C
      EXTERNAL FITCAL
C
      CALL INISAV (XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV)
C
      NFLAT = 1
      CALL STEPIT (FITCAL)
C
      CALL CPYPAR (FITPAR, XOFF, XEXPND, YOFF, AMPL, TEMP, CHINR)
C
      IF (KW. EQ. 16) CLOSE (16)
C
      RETURN
      END
C
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C
      SUBROUTINE INISAV (XOFFSV, XEXPSV, YOFFSV, TEMPSV, CHNRSV)
C
C.... THIS SUBROUTINE INITIALIZES SAVED VALUES OF FITTING PARAMETERS
C.... TO UNREALISTIC VALUES SO ALL ROUTINES IN CALCHI WILL BE CALLED.
C
      XOFFSV = 1.32E-7
```

```
      XEXPSV = 1.32E-7
      YOFFSV = 1.32E-7
      TEMPSV = 1.32E-7
      CHNRSV = 1.32E-7
C
      RETURN
      END
C
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C
      SUBROUTINE LIBGET (CHOICE)
C
C.... THIS SUBROUTINE GETS LIBRARY SPECTRA IF REQUESTED.
C
      PARAMETER (NLP = 1000, NL = 50)
      COMMON /COLIB/ NCOLIB, TCOLIB(NL), NCOSPC, XCO(NLP),
     1   YCO(NLP, NL, 2)
      COMMON /CO2LIB/ NCO2LB, TCO2LB(NL), NCO2SP, XCO2(NLP),
     1   YCO2(NLP, NL, 2)
      COMMON /N2LIB/ NN2LIB, TN2LIB(NL), NN2SPC, XN2(NLP),
     1   YN2(NLP, NL, 2)
      COMMON /O2LIB/ NO2LIB, TO2LIB(NL), NO2SPC, XO2(NLP),
     1   YO2(NLP, NL, 2)
      CHARACTER*3 CHOICE
      CHARACTER*40 XLBFIL
      LOGICAL NOLIB
C
      IF(CHOICE.EQ.'DEF') THEN
        XLBFIL = 'x.lib'
        CALL WSPECT  (NN2SPC, XN2, XLBFIL)
        PRINT*, 'Loading the DEFAULT libraries....'
        CALL LIBSPC (NN2LIB, TN2LIB, NN2SPC, YN2, XLBFIL, NOLIB)
        IF(NOLIB) GOTO 200
      ENDIF

C Remove comments to load CO libraries
C     IF(CHOICE.EQ.'CO '.OR.CHOICE.EQ.'ALL') THEN
C       XLBFIL = 'x.libCO'
C       CALL WSPECT (NCOSPC, XCO, XLBFIL)
C       PRINT*, 'Loading libraries for CO....'
C       CALL LIBSPC (NCOLIB, TCOLIB, NCOSPC, YCO, XLBFIL, NOLIB)
C       IF(NOLIB) GOTO 200
C     ENDIF

      IF(CHOICE.EQ.'CO2'.OR.CHOICE.EQ.'ALL') THEN
```

```
        XLBFIL = 'x.libCO2'
        CALL WSPECT (NCO2SP, XCO2, XLBFIL)
        PRINT*, 'Loading libraries for CO2...'
        CALL LIBSPC (NCO2LB, TCO2LB, NCO2SP, YCO2, XLBFIL, NOLIB)
        IF(NOLIB) GOTO 200
      ENDIF

      IF(CHOICE.EQ.'O2 '.OR.CHOICE.EQ.'ALL') THEN
        XLBFIL = 'x.libO2'
        CALL WSPECT (NO2SPC, XO2, XLBFIL)
        PRINT*, 'Loading libraries for O2....'
        CALL LIBSPC (NO2LIB, TO2LIB, NO2SPC, YO2, XLBFIL, NOLIB)
        IF(NOLIB) GOTO 200
      ENDIF

      IF(CHOICE.EQ.'N2 '.OR.CHOICE.EQ.'ALL') THEN
        XLBFIL = 'x.libN2'
        CALL WSPECT (NN2SPC, XN2, XLBFIL)
        PRINT*, 'Loading libraries for N2....'
        CALL LIBSPC (NN2LIB, TN2LIB, NN2SPC, YN2, XLBFIL, NOLIB)
        IF(NOLIB) GOTO 200
      ENDIF

C
      RETURN
 200  PRINT*, 'The expected basename is txxx0.lib', XLBFIL(6:8)
      PRINT*, 'MAKE SURE THE REQUESTED LIBRARIES EXIST IN THE WORKING'
     1        ,' DIRECTORY....'
      STOP
      END
C
C----------------------------------------------------------------------
C----------------------------------------------------------------------
C
      SUBROUTINE LIBSPC (NLIB, TLIB, NSPECT, YSPECT, XLBFIL, NOLIB)
C
C.... THIS SUBROUTINE READS IN LIBRARY SPECTRA.  THERE ARE TWO ENTRIES
C.... PER RECORD.  THE FIRST IS THE REAL PART OF THE RESONANT
C.... SUSCEPTIBILITY.  THE SECOND IS THE SUM OF THE SQUARES OF THE
C.... REAL AND IMAGINARY PART OF THE RESONANT SUSCEPTIBILITY.
C.... NOTE THAT THE READ FORMAT IS 'UNFORMATTED,' SO THAT BINARY
C.... INSTEAD OF ASCII DATA IS EXPECTED.  THE WAVENUMBER IS THE ENTRY
C.... IN THE CORRESPONDING RECORD IN FILE X.LIB.  (THE FILE X.LIB
C.... IS FORMATTED.)
C
      PARAMETER (NLP = 1000, NL = 50)
```

```
      DIMENSION TLIB(*), YSPECT(NLP, NL, *)
      CHARACTER*(*) XLBFIL
      CHARACTER*40 LIBFIL
      LOGICAL NOLIB
C
C.... THE FILENAME OF A LIBRARY SPECTRUM IS TXXXX.YYY, WHERE XXXX IS
C.... THE TEMPERATURE (IT MUST BE A MULTIPLE OF TEN), AND YYY IS THE
C.... EXTENDER SELECTED IN SUBROUTINE WSPECT FOR THE WAVENUMBER
C.... LIBRARY FILE.
Cbyu  This WAVENUMBER file contains the Raman Shift, in cm-1, of the library.
Cbyu  The extender YYY can be up to IMAX-9 long....
C
      NOLIB = .FALSE.
      LIBFIL = 'txxx0.lib'
C
      DO 200 I = 1, 40
        IF (XLBFIL(I:I) .EQ. '.') GO TO 400
200   CONTINUE
      GO TO 500
400   IMAX = MIN(40-I+6, 40)
      LIBFIL(7:IMAX) = XLBFIL(I+1:40)
C
      NLIB = 0
C
      DO 300 JT = 200, 4000, 10
        L1 = INT(JT/1000)
        L2 = INT((JT -1000*L1)/100)
        L3 = INT((JT -1000*L1-100*L2)/10)
        LIBFIL(2:2) = CHAR(L1+48)
        LIBFIL(3:3) = CHAR(L2+48)
        LIBFIL(4:4) = CHAR(L3+48)
C
        OPEN (15, FILE = LIBFIL, STATUS = 'OLD', ERR = 300,
     1      FORM = 'UNFORMATTED')
        NLIB = NLIB+1
        IF (NLIB .GT. NL) THEN
          WRITE (*, '(1X, A, I3, A)') ' More than ', NL,
     1        ' libraries found; increase dimension of nl.'
          STOP
        ENDIF
C
        REWIND 15
C
        DO 100 I = 1, NSPECT
          READ (15) YSPECT(I, NLIB, 1), YSPECT(I, NLIB, 2)
100     CONTINUE
```

```
C
        CLOSE (15)
        TLIB(NLIB) = FLOAT(JT)
300   CONTINUE
C
500   IF (NLIB .LE. 0) THEN
        WRITE (*, '(1X/1X, 4A/)') ' No library intensity',
     1      ' files found with the extender "', LIBFIL(7:LENTH(LIBFIL))
     2      , '"'
        NOLIB = .TRUE.
        RETURN
      ENDIF
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE NORM (N, X, Y, YMAX)
C
C.... FOR UNREFERENCED SPECTRA, THE DATA POINTS ARE NORMALIZED
C.... TO THE LARGEST DATA POINT.
C
      DIMENSION X(*), Y(*)
C
      YMAX = 0.
C
      DO 100 I = 1, N
        IF (Y(I) .GT. YMAX) YMAX = Y(I)
100   CONTINUE
C
      IF (YMAX .LE. 0.) THEN
        PRINT *, ' YMAX = 0. IN NORM'
        RETURN
      ENDIF
C
      DO 200 I = 1, N
        Y(I) = Y(I)/YMAX
200   CONTINUE
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
```

```
C
      SUBROUTINE RESCAL (NPTS, Y1, Y2, Y1D, Y2D, Y3D, YAMAX, AMPL,
     1   XPTS, AREATH, AREADT, YOFF)
C
      DIMENSION Y1(*), Y2(*), Y1D(*), Y2D(*), Y3D(*), XPTS(*)
C
C.... SCALE DATA AND CALCULATION TO QUANTITATIVE AMPLITUDES.
C.... IN FITTING DATA, CALCULATION WAS SCALED TO 1/AMPL AND DATA
C.... WAS SCALED TO 1.  Y3D IS FOR PLOTTING DIFFERENCE BETWEEN
C.... DATA AND THEORY, OFFSET BY YAMAX/9.
C
      DO 100 I = 1, NPTS
        Y1D(I) = Y1(I)*YAMAX*AMPL
        Y2D(I) = Y2(I)*YAMAX*AMPL
        Y3D(I) = Y1D(I)-Y2D(I)-0.11*YAMAX
100   CONTINUE
C
C.... CALCULATE AREA OF THEORY AND DATA MINUS YOFF.
C
      AREATH = 0.
      AREADT = 0.
C
      DO 200 I = 2, NPTS
        AREATH = AREATH+(Y2D(I)+Y2D(I-1)-2.*YOFF)*(XPTS(I)-XPTS(I-1))
        AREADT = AREADT+(Y1D(I)+Y1D(I-1)-2.*YOFF)*(XPTS(I)-XPTS(I-1))
200   CONTINUE
C
      AREATH = AREATH*0.5
      AREADT = AREADT*0.5
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE REVDAT (NPTS, X, Y, X1, Y1)
C
C.... THIS SUBROUTINE PUTS THE LOWEST X VALUE FIRST IN ARRAYS.
C
      PARAMETER (ND = 1024)
      DIMENSION X(*), Y(*), X1(*), Y1(*)
      DIMENSION XREV(ND), YREV(ND), X1REV(ND), Y1REV(ND)
C
      DO 100 I = 1, NPTS
        XREV(I) = X(I)
        YREV(I) = Y(I)
        X1REV(I) = X1(I)
        Y1REV(I) = Y1(I)
100   CONTINUE
C
      DO 200 K = 1, NPTS
        X(K) = XREV(NPTS+1-K)
        Y(K) = YREV(NPTS+1-K)
        X1(K) = X1REV(NPTS+1-K)
        Y1(K) = Y1REV(NPTS+1-K)
200   CONTINUE
C
      RETURN
      END
C
C-----------------------------------------------------------------------
C-----------------------------------------------------------------------
C
      SUBROUTINE WSET (WBEG, WEND, NDAT, XDATA, XOFF, XEXPND, IMIN,
     1   IMAX, NPTS)
C
C.... THIS SUBROUTINE DETERMINES WHICH DATA POINTS TO INCLUDE IN THE
C.... CALCULATION.
C
      DIMENSION XDATA(*)
      LOGICAL WFLAG
C
      XPL1 = XDATA(1)
      IMIN = 1
      IMAX = NDAT
      WFLAG = .FALSE.
C
      DO 100 I = 1, NDAT
        DIFOLD = XDATA(I)-XPL1
        XDI = XPL1+DIFOLD*XEXPND+XOFF
       IF ((XDI .GE. WBEG) .AND. (.NOT. WFLAG)) THEN
          IMIN = I
          WFLAG = .TRUE.
       ELSE IF (XDI .GT. WEND) THEN
          IMAX = I-1
          GO TO 200
        ENDIF
100   CONTINUE
C
200   CONTINUE
C
```

```
      NPTS = IMAX-IMIN+1
C
      RETURN
      END
C
C------------------------------------------------------------------------
C------------------------------------------------------------------------
C
     SUBROUTINE WSPECT (NSPECT, XSPECT, XLBFIL)
C
C.... THIS SUBROUTINE SETS UP THE WAVENUMBER VECTOR FOR THE
CALCULATED
C.... LIBRARY SPECTRA FROM FILE X.LIB.  THE FORMAT IN X.LIB SHOULD BE A
C.... SINGLE WAVENUMBER PER RECORD IN ASCENDING ORDER.  THE NUMBER
OF
C.... RECORDS IS NSPECT.
Cbyu  This WAVENUMBER file contains the Raman Shift, in cm-1, of the library.
C
     PARAMETER(NLP = 1000)
     DIMENSION XSPECT(*)
      CHARACTER*(*) XLBFIL
C
      OPEN (17, FILE = XLBFIL, STATUS = 'OLD', ERR = 200)
C
      REWIND 17
      NSPECT = 0
300   NSPECT = NSPECT+1
      READ (17, *, END = 400) XSPECT(NSPECT)
     GO TO 300
400   NSPECT = NSPECT -1
      IF(NSPECT.GT.NLP) THEN
       PRINT*, XLBFIL(1:LENTH(XLBFIL)), ' has too many records!'
       PRINT*, 'PARAMETER NLP needs to be increased....'
       PRINT*, 'Terminating the program'
       STOP
      ENDIF
      CLOSE (17)
C
      RETURN
200   WRITE (*, '(1X, 3A/)') ' Error opening library wavenumber ',
     1   'file ', XLBFIL
      STOP
C
      END
```


IPDARS

```
      PROGRAM IPDARS
C        ****** IPDA Raman Shifts  ******
C  Assigns the Raman shift (in cm-1) corresponding to each pixel in the
C  Intensified Photo Diode Array at the BYU's Optics Lab.
C  Programmer:  Daniel V. Flores
C       Date: Dec 15, 1998
C  ASSUMPTIONS: Spectrometer is setup to send the CARS signals of N2, CO,
C             O2 and CO2 simultaneously to the IPDA.
C  VARIABLES:
C       y is an array containing the Intensity values of the Xe spectra
C         recorded in each pixel of the IPDA
C  Required files:
C       fpeaks.f sltc.f
C---------------------------------------------------------------------------
      INTEGER ipdaPx, defPek
      PARAMETER (ipdaPx = 1024, defPek = 4)
      CHARACTER ans*1, filNam*80, errMes*25
      INTEGER O2end, N2beg, nPeaks, Xepkpx(defPek), LENTH, DELPN2
      INTEGER O2beg, CObeg
      LOGICAL prntsc
      REAL lamlN2, lamrN2, lamlO2, lamrO2, y(ipdaPx), pixlRS(ipdaPx)

C  The following wavelengths are in nanometers.
C  Source: "Pen-Ray Rare Gas Spectra", Ultra-Violet Products, Inc.
C  lamrN2 is out of IPDA range, ca 311 pixel-distance from lamlN2
C  (see DELPN2)
      DATA lamlN2, lamrN2, lamlO2, lamrO2 /473.416, 469.70,
     $                              492.315, 491.651/
      DATA DELPN2/311/

      DATA O2beg, O2end/250, 450/
      DATA CObeg, N2beg /500, 750/

C  There are 3 strong peaks over IPDA
      nPeaks = 3
      prntsc = .TRUE.

 5    PRINT*, 'Enter the name of the file containing the Xe spectra:'
      PRINT*, '   (TYPE "done" or simply press RETURN when finished)'
      READ '(A)', filNam

      IF(filNam.EQ.'done'.OR.filNam.EQ.' ') GOTO 40
```

```
      CALL XeDaIn(filNam, y, errMes)

      IF(errMes.NE.'O.K.') THEN
        PRINT*,'Problem in accessing ', filNam(1:LENTH(filNam)),
     $        ': ', errMes
        PRINT*,' '
        GOTO 5
      ENDIF

      CALL FPEAKS(y, ipdaPx, prntsc, Xepkpx, nPeaks)

      CALL CALCRS(Xepkpx(1), Xepkpx(2), lamlO2, lamrO2,
     $            1, O2end, pixlRS)

      CALL FILGAP(O2end + 1, CObeg - 1, pixlRS)

      CALL CALCRS(Xepkpx(3) , Xepkpx(3) + DELPN2, lamlN2, lamrN2,
     $            CObeg, ipdaPx, pixlRS)

      PRINT*,'Modify IPDA''s Raman Shift obtained from Xe spectrum? <N>'
      READ(5,'(A)') ANS
      IF(ANS.EQ.'Y' .OR. ANS.EQ.'y') THEN
        CALL MODRS('CO2', 1, O2beg - 1, pixlRS)
        CALL MODRS('O2', O2beg, O2end, pixlRS)
        CALL MODRS('CO', CObeg, N2beg - 1, pixlRS)
        CALL MODRS('N2', N2beg, ipdaPx, pixlRS)
      ENDIF

      CALL PRNTFL(pixlRS)

      GOTO 5

   40 PRINT '(1X / (A))', 'JOB ACCOMPLISHED!......'

      END



      SUBROUTINE CALCRS(Pl, Pr, laml, lamr, beg, end, pixlRs)
      INTEGER ipdaPx
      PARAMETER (ipdaPx = 1024)
      INTEGER beg, end, pixel, Pl, Pr
      REAL lami, laml, lamr, pmpfrq, pmplam, pixlRs(ipdaPx), slope
C  All wavelengths are in nanometers.
      DATA pmplam/532.0/

      slope = (laml - lamr) / (Pl - Pr)
      pmpfrq = 1/pmplam

      DO 10 pixel = beg, end
        lami = slope*(pixel - Pr) + lamr
        pixlRs(pixel) = (1/lami - pmpfrq)*1.0E7
   10 CONTINUE

      RETURN
      END

      SUBROUTINE FILGAP(beg, end, pixlRS)
      INTEGER ipdaPx
      PARAMETER(ipdaPx = 1024)
      INTEGER beg, end, i
      REAL pixlRS(ipdaPx), filFrq
      DATA filFrq /1999.0/

      DO 10 i=beg, end
        pixlRS(i) = filFrq
   10 CONTINUE

      RETURN
      END

      SUBROUTINE MODRS(SECTN, BEG, END, PIXLRS)
C     ***** MODify the Raman Shift *****

      INTEGER ipdaPx
      PARAMETER (ipdaPx = 1024)
      CHARACTER ANS*1, SECTN*(*)
      INTEGER I, BEG, END
      REAL pixlRS(ipdaPx), XOFF, XEXPND

5     PRINT*, 'Horizontal Shift (cm-1) for ', SECTN, 'section?'
      READ*, XOFF
      PRINT*, 'Horizontal Expansion for ', SECTN, 'section?'
      READ*, XEXPND

      PRINT*, '    ***** Changes to be made: *****'
      PRINT*, 'Horizontal Shift (cm-1) = ', XOFF
      PRINT*, 'Horizontal Expansion    = ', XEXPND
      PRINT*, '---------------------------------------'
      PRINT*, 'Proceed? <Y>'
      READ(5,'(A)') ANS
```

```
      IF(ANS.EQ.' ' .OR. ANS.EQ.'Y' .OR. ANS.EQ.'y') THEN
        CONTINUE
      ELSE
        PRINT*,' '
        GOTO 5
      ENDIF

      DO 10 I= BEG, END
        PIXLRS(I) = PIXLRS(BEG)+XOFF+(PIXLRS(I)-PIXLRS(BEG))*XEXPND
10    CONTINUE

      RETURN
      END

      SUBROUTINE PRNTFL(pixlRS)
      INTEGER ipdaPx
      PARAMETER (ipdaPx = 1024)
      CHARACTER header*61, outFil*10
      INTEGER i, uniNum, IFIRSC, LENTH
      REAL pixlRS(ipdaPx)
      DATA uniNum/15/
      DATA
     1 header/
     2 '*** IPDA''s Raman Shift (cm-1). 1st datum is for pixel #1 ***'/

      outFil = 'IpdaRS'

      OPEN(UNIT = uniNum, FILE= outFil(1:LENTH(outFil)),
     $     STATUS = 'UNKNOWN')

      PRINT*, ' Writing pixel''s Raman Shift to '
     $        , outFil(IFIRSC(outFil):LENTH(outFil))

      WRITE(uniNum,'(1X, A)') Header
      DO 10 i=1,ipdaPx
        WRITE(uniNum, '(1X, G14.8)') pixlRS(i)
 10   CONTINUE

      CLOSE(uniNum)

      RETURN
      END

      FUNCTION IFIRSC(STRING)
C
C-----------------------------------------------------------------------
C  position of first non-blank character in string of arbitrary length
C  if no characters are found, ifirsc is set = 0
C-----------------------------------------------------------------------
C
      CHARACTER* (*)STRING
      INTEGER I, IFIRSC, NLOOP
C
      NLOOP = LEN(STRING)
C
      IF (NLOOP .EQ. 0) THEN
        IFIRSC = 0
        RETURN
      ENDIF
C
      DO 100 I = 1, NLOOP
        IF (STRING(I:I) .NE. ' ') GO TO 120
100   CONTINUE
C
      IFIRSC = 0
      RETURN
120   CONTINUE
      IFIRSC = I
      END

      FUNCTION LENTH(STRING)
C
C-----------------------------------------------------------------------
C  position of last non-blank character in string of arbitrary length
C-----------------------------------------------------------------------
C
      CHARACTER* (*)STRING
      INTEGER I, LENTH, NLOOP
C
      NLOOP = LEN(STRING)
C
      DO 100 I = NLOOP, 1, -1
        IF (STRING(I:I) .NE. ' ') GO TO 120
100   CONTINUE
C
120   CONTINUE
      LENTH = I
      END
```

VOIGT FUNCTION IN CARSFT

```
      SUBROUTINE PRBVOI(NPRB, WLL, WVL, WLR, WVR, WPR, TINSTR)
C     *********  PRoBe instrument function with asymmetric   *********
C     *********  VOIgt line shape                            *********
C    Calculates a non-symmetrical Voigt profile from the Voigt and
C    Lorentzian linewidths for the left and right halves.
C    The shape of the probe instrument function (Intensity vs Wavenumber)
C    is said to follow closely a voigt type distribution.
C    More on this later....Daniel V. Flores 03/17/1999
C  Input:
C      NPRB = Number of points in the probe function
C      WLL = Left-side Lorentzian width (FWHM)
C      WVL = Left-side Voigt width (FWHM)
C      WLR = Right-side Lorentzian width (FWHM)
C      WVR = Right-side Voigt width (FWHM)
C  Output:
C      WPR = Wavenumber array.  Each element is effectevily the difference
C           between the centerline value and the WPR value itself because
C           the centerline value is zero in these calculations.
C      TINSTR = Probe instrument function.  Centerline value equals 1
C  Observations:
C

      DOUBLE PRECISION WVOIGT, DWAVL, DWAVR, WPR(*)
      INTEGER I, IRIGHT, MIDPNT, NPRB, RCOUNT
      REAL TINSTR(*), EXTENT, WLL, WLR, WVL, WVR

C  EXTENT is the number of wavenumbers the line shape will extend to either
C  side from the centerline.  A value of 75 was used by Boyack (see file
C  VOIGT.F in the VAX_CARS_PRGRAMS directory).  EXTENT does have a big impact
C  on the convolved spectrum.  I don't understand yet how, but for the moment
C  I'll use what has worked before. May be EXTENT should also be a fitting
C  variable....                  Daniel 03/31/99

      DATA EXTENT/75/

C  Calculate the mid point array index, MIDPNT, (i.e., centerline)
      MIDPNT = INT(NPRB/2)

C  Calculate X and Y values at far left, centerline and far right.
      WPR(1) = -EXTENT
      WPR(MIDPNT) = 0
      WPR(NPRB) = EXTENT

      TINSTR(1) = WVOIGT(WPR(1), WLL, WVL)
      TINSTR(MIDPNT) = WVOIGT(WPR(MIDPNT), WLL, WVL)
      TINSTR(NPRB) = WVOIGT(WPR(NPRB), WLR, WVR)

C  Calculate the wavenumber spacing for left and right wings
      DWAVL = -WPR(1)/(MIDPNT - 1)
      DWAVR = WPR(NPRB)/(NPRB - MIDPNT)

C  Compute the instrument probe function with a voigt profile

      IRIGHT = MIDPNT - 1
      DO 10 I=2,MIDPNT-1
C  Left wing X and Y
        WPR(I) = WPR(I-1) + DWAVL
        TINSTR(I) = WVOIGT(WPR(I), WLL, WVL)

C  Right wing X and Y
        RCOUNT = IRIGHT + I
        WPR(RCOUNT) = WPR( RCOUNT-1) + DWAVR
        TINSTR(RCOUNT) = WVOIGT(WPR(RCOUNT), WLR, WVR)
 10   CONTINUE

C  Complete right wing
      DO 20 I=2*MIDPNT-1, NPRB-1
        WPR(I) = WPR(I-1) + DWAVR
        TINSTR(I) = WVOIGT(WPR(I), WLR, WVR)
 20   CONTINUE

      END


      FUNCTION WVOIGT(DLAM, WL, WV)
C     ******** Whiting's approximation to the VOIGT function *********
C
C     Approximation to the Voigt function.
C     REF:  Whiting, J. Quant. Spectr. Rad. Trans. 8, 1379 (1968).
C     Includes the second correction, SECCOR, which increases the accuracy of
C     the wings.
C     Programmer:  Daniel V. Flores, 03/17/1999
C  Input:
C        DLAM = Difference between abscissa's centerline and point of interest
C               (Delta LAMbda if abscissa is in wavelengths)
C        WL   = Lorentzian width (FWHM)
C        WV   = Voigt width (FWHM)
```

```
      DOUBLE PRECISION WVOIGT, DLAM, SECCOR
      REAL WL, WV

      WVOIGT = (1-WL/WV)*EXP(-2.772*(DLAM/WV)**2)
     1       + (WL/WV)/(1+4*(DLAM/WV)**2)


      SECCOR = 0.016*(1-WL/WV)*(WL/WV)
     1       * (EXP(-0.4*(ABS(DLAM/WV))**2.25)
     2            - 10/(10 + (ABS(DLAM/WV))**2.25))

      WVOIGT = WVOIGT + SECCOR

      RETURN
      END

C **********   A VARIETY OF SUBROUTINES TO HANDLE THE INTERPHASE ********
C **********   TO CALCULATE THE VOIGT PROFILE IN CARSFT         ********
C    This file contains several subroutines used to manipulate the values
C    of the Lorentzian and Voigt widths in CARSFT.
C    These subroutines make use of other subroutines originally provided with
C    CARSFT (e.g., crslib.f, carsft.f).
C    Programmer:                        Daniel V. Flores 03/31/99

      SUBROUTINE CHKVOI(ERROR)
C ****** CHecK the VOIgt parameters for consistency *******
C   Voigt Parameters common statement
      COMMON /CVOIGT/ LVOIGT, WLL, WVL, WLR, WVR
      LOGICAL ERROR, LVOIGT
      REAL WLL, WVL, WLR, WVR

      IF(WLL.LT.0.0.OR.WLR.LT.0.0.OR.WVL.LE.0.0.OR.WVR.LE.0.0) THEN
        ERROR = .TRUE.
      ELSE IF(WLL.GT.WVL.OR.WLR.GT.WVR) THEN
C Whitings approximation does allow for WL = WV. (see FUNCTION WVOIGT)
        ERROR = .TRUE.
      ELSE
        ERROR = .FALSE.
      ENDIF

      RETURN
      END

      SUBROUTINE DISVOI(WIDNAM, WIDTHS)
C ********  DISplay VOIgt paramet ers *********
      CHARACTER WIDNAM(5)*17
      REAL WIDTHS(4)
      INTEGER I

      PRINT '(/5A1, A, 5A1/)', ('_', I=1,5), ' VOIGT WIDTHS (FWHM) ',
     1                   ('_', I=1,5)
      PRINT 100, (I, WIDNAM(I), WIDTHS(I), I = 1,4)

      PRINT '(" (", I1, ") ", A16)', 5, WIDNAM(5)

      PRINT '(31A1/)', ('_', I=1,31)

 100  FORMAT(' (', I1, ') ', A16, 3X, F7.4)

      END

      SUBROUTINE INPVOI(NLINE, INFO)
C ***** interprets the VOIgt parameters INPut through a parameter file *****
C ***** (e.g., cars.par)                                           *****
C   Voigt Parameters common statement
      COMMON /CVOIGT/ LVOIGT, WLL, WVL, WLR, WVR
      LOGICAL LVOIGT
      CHARACTER* (*)INFO
      REAL WLL, WVL, WLR, WVR
      REAL DAT(4)
      INTEGER IER, NFOUND, NLINE

C    Interpret the Voigt parameters from the character array INFO
      CALL INTERP (NLINE, INFO, -4, NFOUND, DAT, IER)
      IF (IER .NE. 0) RETURN
      IF(NFOUND.EQ.2) THEN
        WLL = DAT(1)
        WVL = DAT(2)
C    Assume symmetric voigt is asked for...
        WLR = WLL
        WVR = WVL
      ELSE IF(NFOUND.EQ.4) THEN
        WLL = DAT(1)
        WVL = DAT(2)
        WLR = DAT(3)
        WVR = DAT(4)
      ELSE
        CALL VARERR ( 'MUST HAVE EITHER 2 OR 4 VALUES' ,
     1    NLINE)
      ENDIF
```

```
      RETURN
      END

      SUBROUTINE MODVOI()
C  ********  MODify VOIgt parameters *********
Cbyu Voigt Parameters common statement
      COMMON /CVOIGT/ LVOIGT, WIDTHS(4)
      COMMON /CFTVOI/ LFTVOI, VOIFIT(4,4), VOINUM(4)
      CHARACTER*2 VOINUM
      LOGICAL LVOIGT, LFTVOI
      REAL WIDTHS, VOIFIT
      CHARACTER STRING*80, WIDNAM(5)*17
      INTEGER I, IER, LENTH, N, NCHANG, NFOUND, NVAR
      REAL CHANG(4), DAT(4)
      LOGICAL ERROR, SHOWAG

      DATA WIDNAM/'Left Lorentzian', 'Left Voigt', 'Right Lorentzian',
     1         'Right Voigt', 'All Widths'/

      ERROR = .FALSE.

 1    CALL DISVOI(WIDNAM, WIDTHS)

      IF(ERROR) THEN
        PRINT*,'!!! ERROR !!!'
        PRINT*,'Please correct the width values to meet the ',
     1       'following criteria:'
        PRINT*,' Voigt widths must be > 0'
        PRINT*,' Lorentzian widths must be > or = 0'
        PRINT*,' Lorentzian width must be < or = corresp. Voigt width'
        PRINT*, ' '
      ENDIF

 5    PRINT *, 'Numbers of items to be changed ? <none> '
      READ (5, '(A)' ) STRING

      IF (STRING .NE. ' ') THEN

        CALL INTERP (0, STRING, -4, NCHANG, CHANG, IER)
        IF (IER .NE. 0) GO TO 5

C   Check if "All Widths" has been requested (option 5)
        I = 1
 10     IF(I.LE.NCHANG) THEN
          IF(INT(CHANG(I)).EQ.5) THEN
            DO 20 N = 1, 4
```

```
              CHANG(N) = N
 20         CONTINUE
            NCHANG = 4
            I = 4
          ENDIF
          I = I + 1
          GOTO 10
        ENDIF

C   Prompt user for the new values of requested parameters
        DO 40 N = 1, NCHANG
          NVAR = CHANG(N)
 30       STRING = 'New value for ' //
     1    WIDNAM(NVAR)(1:LENTH(WIDNAM(NVAR))) // '? <unchanged> '
          PRINT *, STRING(1:LENTH(STRING))
          READ (5, '(A)' ) STRING
          CALL INTERP (0, STRING, -1, NFOUND, DAT, IER)
          IF (NFOUND .EQ. 1) WIDTHS(NVAR) = DAT(1)
          IF (IER .NE. 0) GO TO 30
 40     CONTINUE

C  Check the width values for consistency
C       CALL CHKVOI(ERROR)
        SHOWAG = .TRUE.
      ELSE
        SHOWAG = .FALSE.
      ENDIF

      IF(ERROR.OR.SHOWAG) GOTO 1

C  Update the fitting parameters for TASK3
      VOIFIT(1,1) = WIDTHS(1)/WIDTHS(2)
      VOIFIT(2,1) = WIDTHS(2)
      VOIFIT(3,1) = WIDTHS(3)/WIDTHS(4)
      VOIFIT(4,1) = WIDTHS(4)

      RETURN
      END

      SUBROUTINE PLTVOI(NPRB, WLL, WVL, WLR, WVR, WPR, TINSTR)
      INTEGER N, NPRB, LENTH
      REAL WLL, WVL, WLR, WVR, TINSTR(*), WPRSGL(NPRB)
      CHARACTER ANS*1, TITLE*80, TMPCHR*6
      DOUBLE PRECISION WPR(*)

      PRINT*,'Plot Instrument Function? <N>'
```

```
      READ(5,'(A)') ANS
      IF(ANS.EQ.'N'.OR.ANS.EQ.'n'.OR.ANS.EQ.' ') GOTO 150

      DO 10 N= 1, NPRB
       WPRSGL(N) = SNGL(WPR(N))
 10   CONTINUE

C  Construct the TITLE..

      WRITE(TMPCHR, '(F6.3)') WLL
      TITLE = '(WLL=' // TMPCHR // ', WVL='
      WRITE(TMPCHR, '(F6.3)') WVL
      TITLE = TITLE(1:LENTH(TITLE)) // TMPCHR // ') | (WLR ='
      WRITE(TMPCHR, '(F6.3)') WLR
      TITLE = TITLE(1:LENTH(TITLE)) // TMPCHR // ', WVR='
      WRITE(TMPCHR, '(F6.3)') WVR
      TITLE = TITLE(1:LENTH(TITLE)) // TMPCHR // ')'

      CALL PLTCHI(WPRSGL, TINSTR, WPRSGL, TINSTR, NPRB,
     1           'Wavenumber (cm-1)', 'Normalized Intensity',
     2            TITLE(1:LENTH(TITLE)), 'VOIGT PROFILE', ' ',
     3            1, .TRUE.)

150   PRINT*,'Continuing with computations...'
      PRINT*,' '

      RETURN
      END

      SUBROUTINE VOIDEF()
C  ******* set VOIgt widths and fitting parameters DEFault values *******
C  Voigt Parameters common statement
      INTEGER NF, NP
      PARAMETER (NF = 26, NP = 30)
      COMMON /CVOIGT/ LVOIGT, WLL, WVL, WLR, WVR
      COMMON /CFTVOI/ LFTVOI, VOIFIT(NF+1:NF+4,4), VOINUM(NF+1:NF+4)
      CHARACTER*2  VOINUM
      LOGICAL LVOIGT, LFTVOI
      REAL WLL, WVL, WLR, WVR, VOIFIT

C  Check CARSFT dimmensions are consistent with those of STEPIT
      IF(NF+4.GT.NP) THEN
       PRINT*, 'Parameter NF ( =', NF, ') + 4 in CARSFT should not'
       PRINT*, 'be greater than parameter NP ( =', NP, 'in STEPIT'
       PRINT*, 'QUITING PROGRAM!'
       STOP
      ENDIF

C  Initialize the Asymmetric Voigt profile parameters
      LVOIGT = .FALSE.
      LFTVOI = .FALSE.
      WLL = 1.1
      WVL = 1.7
      WLR = 1.2
      WVR = 2.2
C  Initialize nominal values of Voigt fitting variables
      VOIFIT(27,1) = WLL/WVL
      VOIFIT(28,1) = WVL
      VOIFIT(29,1) = WLR/WVR
      VOIFIT(30,1) = WVR
C  Initialize limits: minima
      VOIFIT(27,2) = 0.0
      VOIFIT(28,2) = 0.5
      VOIFIT(29,2) = 0.0
      VOIFIT(30,2) = 0.5
C  Initialize limits: maxima
      VOIFIT(27,3) = 1.0
      VOIFIT(28,3) = 4.0
      VOIFIT(29,3) = 1.0
      VOIFIT(30,3) = 4.0
C  Voifit are free during fitting.  They are used only if LFTVOI = .TRUE.
C  Also, set values of VOINUM, used in SUB PARSE of CARSFT.
      DO 120 N= NF+1, NF+4
       VOIFIT(N,4) = -1
       WRITE(VOINUM(N), '(I2)') N
120   CONTINUE

      RETURN
      END

      SUBROUTINE VOIDEL(VOIFIT, XVAR, XMIN, XMAX, DELTX, DELMN,
     1   MASK)
C
C______________________________________________________________________
C
C  establishes minimum and maximum values, and initial and final step
C  sizes for VOIFIT variables.
C______________________________________________________________________
C
      PARAMETER (NF = 26)
      DIMENSION VOIFIT(NF+1:NF+4,4),  XVAR(*), XMIN(*), XMAX(*),
     1  DELTX(*), DELMN(*), MASK(*)
```

```
C  Transfer VOIFIT parameters to STEPIT equivalents. Also set the initial and
C  minimum step sizes.
     DO 110 N = NF+1, NF+4
       XVAR(N) = VOIFIT(N, 1)
       XMIN(N) = VOIFIT(N, 2)
       XMAX(N) = VOIFIT(N, 3)
       DELTX(N) = 0.1
       DELMN(N) = 1.E-5
       IF (VOIFIT(N, 4) .GT. 0.) THEN
         MASK(N) = 1
       ELSEIF (VOIFIT(N, 4) .LT. 0) THEN
         MASK(N) = 0
       ENDIF
110   CONTINUE

     RETURN
     END
```

MSCSTS

```
      PROGRAM MSCSTS
C  *******  Multi-Species Cars STatisticS  ********
      INTEGER MAXSTR
      PARAMETER (MAXSTR=31)
      CHARACTER* (MAXSTR) FITFIL
      INTEGER IFIRSC, LENTH, LISNUM, PSPOS, R, RSPOS, Z, ZSPOS
      INTEGER N2KEPT, O2KEPT, CO2KPT
      LOGICAL LSKIP, LSIEVE
      REAL O2MIN, CO2MAX
      CHARACTER ANS*1
      DATA LISNUM/11/
      DATA O2MIN, CO2MAX /0.0555, 0.0777/

      CALL INIOUT(0,0,0)

      CALL OPFILE('fitList', LISNUM, MAXSTR,
     1              'Name of file listing the CARS fit files', 'OLD')

      PRINT*, 'Arbitrary values assigned when # of sieved samples < 2'
C This problem is most likely to show up in high-temperature regions where CARS
C spectra had very low counts. For my premixed cases, it makes sense to assign
C values near the average mol fractions at complete combustion for the two
C equivalence ratios, 0.65 and 0.80, where O2 is minimum and CO2 is maximum.
C This way I avoid eskewing unreasonably the interpolated data between locations
C with good data and those with bad data
C It also helps to make the values somewhat unique so they are easily identified
      PRINT '(A, F7.4)', '      O2 minimum concentration = ', O2MIN
      PRINT '(A, F7.4)', '     CO2 minimum concentration = ', CO2MAX
      PRINT*, ''

      PRINT*, 'Sieve the data using 4*sigma? <N>'
      READ (5, '(A)') ANS
      IF (ANS.EQ.'' .OR. ANS.EQ. 'N' .OR. ANS.EQ. 'n') THEN
        LSIEVE = .FALSE.
      ELSE
        LSIEVE = .TRUE.
        PRINT*, 'Doing 4*SIGMA data filter... '
      ENDIF

10    READ(LISNUM,'(A)', END = 40) FITFIL
```

```
      PRINT*, 'Processing ', FITFIL(IFIRSC(FITFIL):LENTH(FITFIL)),
     1       '.....'

      ZSPOS = INDEX(FITFIL,'_z')

      RSPOS = INDEX(FITFIL,'_r')

      PSPOS = INDEX(FITFIL,'.fit')

      READ(FITFIL(ZSPOS+2:RSPOS-1),'(I3)') Z

      READ(FITFIL(RSPOS+2:PSPOS-1) ,'(I3)') R

      LSKIP = .FALSE.

      CALL GETDAT(FITFIL, LSKIP, N2KEPT)

      IF(LSKIP) GOTO 10

      CALL CONSIE(O2KEPT, CO2KPT)

      CALL COSTAT(N2KEPT, O2KEPT, CO2KPT, O2MIN, CO2MAX)

      IF (LSIEVE) THEN
        CALL SIEVE(N2KEPT, O2KEPT, CO2KPT)
        CALL COSTAT(N2KEPT, O2KEPT, CO2KPT, O2MIN, CO2MAX)
      ENDIF

      PRINT*, '     Specie   # of Kept Samples    Z    R'
      PRINT*, '     ------   -----------------  ---  ---'
      PRINT'(8X, A, 15X, I7, 3X, I3, 2X, I3)', 'N2', N2KEPT, Z, R
      PRINT'(8X, A, 15X, I7, 3X, I3, 2X, I3)', 'O2', O2KEPT, Z, R
      PRINT'(8X, A, 14X, I7, 3X, I3, 2X, I3)', 'CO2',CO2KPT, Z, R

      CALL INIOUT(1, Z, R)

      GOTO 10

40    CALL INIOUT(-1, 0, 0)

      PRINT*, '***** JOB ACCOMPLISHED! *****'
      END
```


StatLib

```
      SUBROUTINE CONSIE(O2KEPT, CO2KPT)
C **********  CONcentration SIEve  ****************
C Discards unnacceptable concentration values
      INTEGER MAXSAM, O2KEPT, CO2KPT
      PARAMETER (MAXSAM = 1000)
      COMMON /THEDAT/ NUMSAM, CNRTOT(MAXSAM), TEMP(MAXSAM),
XCO(MAXSAM),
     1      XCO2(MAXSAM), XO2(MAXSAM), N2PEAK(MAXSAM),
     2      COPEAK(MAXSAM), O2PEAK(MAXSAM), CO2PEK(MAXSAM),
     3      ARERN2(MAXSAM), ARERCO(MAXSAM), ARERO2(MAXSAM),
     4      ARRCO2(MAXSAM)
      INTEGER NUMSAM
      REAL N2PEAK, COPEAK, O2PEAK, CO2PEK
      REAL CNRTOT, TEMP, XCO, XCO2, XO2, ARERN2, ARERCO, ARERO2, ARRCO2
      COMMON /CONIND/XO2IND(MAXSAM), XCO2ID(MAXSAM)
      REAL XO2IND, XCO2ID

      INTEGER LOWCOU
      REAL HIHO2, HIHCO2, LOWARE, HIHARE

      DATA LOWCOU, HIHO2, HIHCO2 /25, 0.21, 0.12/
      DATA LOWARE, HIHARE /0.9, 1.1/

      INTEGER I, J

C Flush O2 array
      I = 0
      DO 10 J=1,NUMSAM
        IF(XO2(J).LT.HIHO2.AND.O2PEAK(J).GT.LOWCOU
     1    .AND.ARERO2(J).GTLOWARE.AND.ARERO2(J).LT.HIHARE) THEN
          I = I + 1
          XO2(I)= XO2(J)
          O2PEAK(I)= O2PEAK(J)
          ARERO2(I)= ARERO2(J)
          XO2IND(I) = J
        ENDIF
10    CONTINUE
      O2KEPT = I

C Flush CO2 array
      I = 0
      DO 20 J=1,NUMSAM
        IF(XCO2(J).LT.HIHCO2.AND.CO2PEK(J).GT.LOWCOU.AND.
```

```
1    ARRCO2(J).GT.LOWARE.AND.ARRCO2(J).LT.HIHARE) THEN
       I = I + 1
       XCO2(I)= XCO2(J)
       CO2PEK(I)= CO2PEK(J)
       ARRCO2(I)= ARRCO2(J)
       XCO2ID(I) = J
     ENDIF
20   CONTINUE
     CO2KPT = I

     RETURN
     END

     SUBROUTINE COSTAT(N2KEPT, O2KEPT, CO2KPT, O2MIN, CO2MAX)
     INTEGER N2KEPT, O2KEPT, CO2KPT
     REAL O2MIN, CO2MAX
     INTEGER MAXSAM, NSTS
     PARAMETER (MAXSAM = 1000, NSTS = 5)
     COMMON /THEDAT/ NUMSAM, CNRTOT(MAXSAM), TEMP(MAXSAM),
XCO(MAXSAM),
1         XCO2(MAXSAM), XO2(MAXSAM), N2PEAK(MAXSAM),
2         COPEAK(MAXSAM), O2PEAK(MAXSAM), CO2PEK(MAXSAM),
3         ARERN2(MAXSAM), ARERCO(MAXSAM), ARERO2(MAXSAM),
4         ARRCO2(MAXSAM)
     INTEGER NUMSAM
     REAL N2PEAK, COPEAK, O2PEAK, CO2PEK
     REAL CNRTOT, TEMP, XCO, XCO2, XO2, ARERN2, ARERCO, ARERO2, ARRCO2

     COMMON /STATS/ CNTSTS(NSTS), TSTS(NSTS), XCOSTS(NST S),
1         XCO2ST(NSTS), XO2STS(NSTS), N2PSTS(NSTS), COPSTS(NSTS),
2         O2PSTS(NSTS), CO2PST(NSTS), AN2STS(NSTS), ACOSTS(NSTS),
3         AO2STS(NSTS), ACO2ST(NSTS)
     REAL CNTSTS, TSTS, XCOSTS, XCO2ST, XO2STS, AN2STS, ACOSTS, AO2STS,
1    ACO2ST
     REAL N2PSTS, COPSTS, O2PSTS, CO2PST

     LOGICAL LESST2


     CALL MOMENT(CNRTOT, N2KEPT, CNTSTS(1), CNTSTS(2), CNTSTS(3),
1          CNTSTS(4), CNTSTS(5), LESST2)

     CALL MOMENT(TEMP, N2KEPT, TSTS(1), TSTS(2), TSTS(3), TSTS(4),
1          TSTS(5), LESST2)

     CALL MOMENT(N2PEAK, N2KEPT, N2PSTS(1), N2PSTS(2), N2PSTS(3),
1          N2PSTS(4), N2PSTS(5), LESST2)

     CALL MOMENT(ARERN2, N2KEPT, AN2STS(1), AN2STS(2), AN2STS(3),
1          AN2STS(4), AN2STS(5), LESST2)

     CALL MOMENT(XCO, NUMSAM, XCOSTS(1), XCOSTS(2), XCOSTS(3),
1          XCOSTS(4), XCOSTS(5), LESST2)

     CALL MOMENT(COPEAK, NUMSAM, COPSTS(1), COPSTS(2), COPSTS(3),
1          COPSTS(4), COPSTS(5), LESST2)

     CALL MOMENT(ARERCO, NUMSAM, ACOSTS(1), ACOSTS(2), ACOSTS(3),
1          ACOSTS(4), ACOSTS(5), LESST2)

     CALL MOMENT(XO2, O2KEPT, XO2STS(1), XO2STS(2), XO2STS(3),
1          XO2STS(4), XO2STS(5), LESST2)

     IF(LESST2) THEN
       XO2STS(1) = O2MIN
       XO2STS(2) = 0.0
       XO2STS(3) = 0.0
       XO2STS(4) = O2MIN
       XO2STS(5) = O2MIN
       O2PSTS(1) = 0.0
       O2PSTS(2) = 0.0
       O2PSTS(3) = 0.0
       O2PSTS(4) = 0.0
       O2PSTS(5) = 0.0
       AO2STS(1) = 0.0
       AO2STS(2) = 0.0
       AO2STS(3) = 0.0
       AO2STS(4) = 0.0
       AO2STS(5) = 0.0
     ELSE
     CALL MOMENT(O2PEAK, O2KEPT, O2PSTS(1), O2PSTS(2), O2PSTS(3),
1          O2PSTS(4), O2PSTS(5), LESST2)

     CALL MOMENT(ARERO2, O2KEPT, AO2STS(1), AO2STS(2), AO2STS(3),
1          AO2STS(4), AO2STS(5), LESST2)
     ENDIF

     CALL MOMENT(XCO2, CO2KPT, XCO2ST(1), XCO2ST(2), XCO2ST(3),
1          XCO2ST(4), XCO2ST(5), LESST2)

     IF(LESST2) THEN
       XCO2ST(1) = CO2MAX
```

```
        XCO2ST(2) = 0.0
        XCO2ST(3) = 0.0
        XCO2ST(4) = CO2MAX
        XCO2ST(5) = CO2MAX
        CO2PST(1) = 0.0
        CO2PST(2) = 0.0
        CO2PST(3) = 0.0
        CO2PST(4) = 0.0
        CO2PST(5) = 0.0
        ACO2ST(1) = 0.0
        ACO2ST(2) = 0.0
        ACO2ST(3) = 0.0
        ACO2ST(4) = 0.0
        ACO2ST(5) = 0.0
      ELSE
      CALL MOMENT(CO2PEK, CO2KPT, CO2PST(1), CO2PST(2), CO2PST(3),
     1          CO2PST(4), CO2PST(5), LESST2)

      CALL MOMENT(ARRCO2, CO2KPT, ACO2ST(1), ACO2ST(2), ACO2ST(3),
     1          ACO2ST(4), ACO2ST(5), LESST2)
      ENDIF

      RETURN
      END


      SUBROUTINE GETDAT(DATFIL, LSKIP, N2KEPT)
      INTEGER MAXSAM, N2KEPT
      PARAMETER (MAXSAM = 1000)
      CHARACTER*(*) DATFIL, DUMSTR*1
      INTEGER DATNUM, I, SHONUM
      LOGICAL LSKIP
      DATA DATNUM/10/
      COMMON /THEDAT/ NUMSAM, CNRTOT(MAXSAM), TEMP(MAXSAM),
XCO(MAXSAM),
     1        XCO2(MAXSAM), XO2(MAXSAM), N2PEAK(MAXSAM),
     2        COPEAK(MAXSAM), O2PEAK(MAXSAM), CO2PEK(MAXSAM),
     3        ARERN2(MAXSAM), ARERCO(MAXSAM), ARERO2(MAXSAM),
     4        ARRCO2(MAXSAM)
      INTEGER NUMSAM
      REAL N2PEAK, COPEAK, O2PEAK, CO2PEK
      REAL CNRTOT, TEMP, XCO, XCO2, XO2, ARERN2, ARERCO, ARERO2, ARRCO2

      OPEN(UNIT=DATNUM, FILE=DATFIL, STATUS='OLD', ERR=40)
C  Read the file headers.
      DO 5 I=1,3
        READ(DATNUM,*) DUMSTR
5     CONTINUE

      I = 1
10    IF(I .LE. MAXSAM + 1) THEN
        READ(DATNUM,*, END=15, ERR = 41) SHONUM, CNRTOT(I), TEMP(I),
     1     XCO(I), XCO2(I), XO2(I), N2PEAK(I), COPEAK(I), O2PEAK(I),
     2     CO2PEK(I), ARERN2(I), ARERCO(I), ARERO2(I), ARRCO2(I)

C Sieve out bad shots: N2 information is off-limits
        IF(ARERN2(I).GT. 0.98 .AND. ARERN2(I).LT. 1.2 .AND.
     1    TEMP(I).GT.250 .AND. TEMP(I).LT.2050) I = I + 1
        GOTO 10
      ELSE
        PRINT*, 'FILE ', DATFIL, 'HAS MORE THAN ', MAXSAM,
     1        ' DATA LINES....'
        PRINT*, 'CHANGE PARAMETER "MAXSAM" TO ACCEPT MORE VALUES.'
        PRINT*, 'STOPPING PROGRAM EXECUTION'
        STOP
      ENDIF

15    NUMSAM = I - 1
C     PRINT*, 'Number of shots kept = ', NUMSAM
C Initialize number of points of each array to use in MOMENT subroutine
      IF(NUMSAM.GE.2) THEN
        N2KEPT = NUMSAM
        LSKIP = .FALSE.
      ELSE
        LSKIP = .TRUE.
        PRINT*, 'File ', DATFIL, 'has less than 2 samples. Skipping ...'
      ENDIF
      GOTO 50

40    PRINT*,'Error while opening ', DATFIL, '. Skipping file...'
      LSKIP = .TRUE.
      GOTO 50

41    PRINT*,'Error in line ', I-1,' of ', DATFIL,'. Skipping file...'
      LSKIP = .TRUE.

50    CLOSE(DATNUM)

      RETURN
      END
```

```
      SUBROUTINE INIOUT(CONTRL, Z, R)
C  *********  INItialize and write the OUTput for the MSCSTS program  *********
      INTEGER MAXSTR, NSTS, Z, R
      PARAMETER (MAXSTR=31, NSTS=5)
      CHARACTER*(MAXSTR) BNAM, FISUBS(NSTS), HEADER(NSTS)
      INTEGER CONTRL, I, IFIRSC, LENTH, OUTFIL(NSTS)
      COMMON /STATS/ CNTSTS(NSTS), TSTS(NSTS), XCOSTS(NSTS),
     1        XCO2ST(NSTS), XO2STS(NSTS), N2PSTS(NSTS), COPSTS(NSTS),
     2        O2PSTS(NSTS), CO2PST(NSTS), AN2STS(NSTS), ACOSTS(NSTS),
     3        AO2STS(NSTS), ACO2ST(NSTS)
      REAL CNTSTS, TSTS, XCOSTS, XCO2ST, XO2STS, AN2STS, ACOSTS, AO2STS,
     1   ACO2ST
      REAL N2PSTS, COPSTS, O2PSTS, CO2PST
      INTEGER N2P, O2P, CO2P, COP

      DATA FISUBS/'.ave','.adev', '.sdev', '.min', '.max'/
      DATA OUTFIL/26, 27, 28, 29, 30/
      DATA HEADER/'CARS DATA AVERAGES:',
     1         'CARS DATA ABSOLUTE DEVIATIONS:',
     2         'CARS DATA STANDARD DEVIATIONS:',
     3         'CARS DATA MINIMA:', 'CARS DATA MAXIMA:'/

      IF(CONTRL.EQ.1) THEN
C  Write the output to the statistics files.
        DO 10 I=1,5
          N2P = ANINT(N2PSTS(I))
          O2P = ANINT(O2PSTS(I))
          CO2P = ANINT(CO2PST(I))
          COP = ANINT(COPSTS(I))

          WRITE(OUTFIL(I),2000) Z, R, CNTSTS(I), TSTS(I), XCOSTS(I),
     1               XCO2ST(I), XO2STS(I), N2P,COP, O2P, CO2P,
     2               AN2STS(I), ACOSTS(I), AO2STS(I), ACO2ST(I)
10      CONTINUE
      ELSEIF(CONTRL.EQ.0) THEN
C  Open and initialize the output files.
        PRINT*, 'Enter the desired basename for the output files:'
        READ*, BNAM
        PRINT*, 'Initializing output files'
        DO 15 I=1,5
          OPEN(UNIT=OUTFIL(I), FILE=BNAM(IFIRSC(BNAM):LENTH(BNAM))
     1      //FISUBS(I)(IFIRSC(FISUBS(I)):LENTH(FISUBS(I))))
          WRITE(OUTFIL(I),*) '******** ', HEADER(I)
          WRITE(OUTFIL(I),1000) 'Z_(mm)', 'R_(mm)','CHINRT', 'T(K)',
     1               'XCO', 'XCO2', 'XO2', 'N2PEAK', 'COPEAK',
     2               'O2PEAK', 'CO2PEK', 'ARERN2', 'ARERCO',
     3               'ARERO2', 'ARRCO2'
15      CONTINUE
      ELSEIF(CONTRL.EQ.-1) THEN
C  Close the output files.
        DO 20 I=1,5
          CLOSE(OUTFIL(I))
20      CONTINUE
      ELSE
        PRINT*, 'Wrong control number passed to WRISTA.....'
        PRINT*, 'Stopping program execution.'
        STOP
      ENDIF

1000  FORMAT(1X, A, 1X, A, 3X, A, 8X, A, 8X, A, 7X, A, 8X, A, 6X, 5(A,
     1   3X), 2X, 3(A, 5X))
C Use this format for numerical output!
2000  FORMAT(2X, I4, 2X, I4, 3X, G12.6, 1X, G11.5, 3(1X, G10.4), 1X,
     1     4(I7, 2X), 4(1X, G10.4))

      RETURN
      END


      SUBROUTINE MOMENT(DATA, N, AVE, ADEV, SDEV, MIN, MAX, LESST2)
C  Code taken in its entirety from "Numerical Recipes in FORTRAN"
C
      INTEGER J, N
      LOGICAL LESST2
      REAL ADEV, AVE, MIN, MAX, SDEV, VAR, DATA(N)
      REAL P, S, EP

      IF(N.LE.1) THEN
C       PRINT*, 'N must be at least 2 in moment'
        LESST2 = .TRUE.
        RETURN
      ELSE
        LESST2 = .FALSE.
      ENDIF

      S = 0.
      DO 11 J=1, N
        S = S + DATA(J)
11    CONTINUE
      MIN = DATA(1)
      MAX = DATA(1)
```

```
      AVE = S/N
      ADEV = 0.
      VAR = 0.
      EP = 0.

      DO 12 J = 1, N
        IF(MIN.GT .DATA(J)) MIN = DATA(J)
        IF(MAX.LT.DATA(J)) MAX = DATA(J)
        S = DATA(J) - AVE
        EP = EP + S
        ADEV = ADEV + ABS(S)
        P = S*S
        VAR = VAR + P
12    CONTINUE

      ADEV = ADEV/N
      VAR = (VAR - EP**2/N)/(N-1)
      SDEV = SQRT(VAR)


      RETURN
      END

      SUBROUTINE SIEVE(N2KEPT, O2KEPT, CO2KPT)
C Sieve all statistical arrays using 4*sigma as the criterion.
      INTEGER MAXSAM, NSTS, N2KEPT, O2KEPT, CO2KPT
      PARAMETER (MAXSAM = 1000, NSTS = 5)
      COMMON /THEDAT/ NUMSAM, CNRTOT(MAXSAM), TEMP(MAXSAM),
XCO(MAXSAM),
     1          XCO2(MAXSAM), XO2(MAXSAM), N2PEAK(MAXSAM),
     2          COPEAK(MAXSAM), O2PEAK(MAXSAM), CO2PEK(MAXSAM),
     3          ARERN2(MAXSAM), ARERCO(MAXSAM), ARERO2(MAXSAM),
     4          ARRCO2(MAXSAM)
      INTEGER NUMSAM
      REAL N2PEAK, COPEAK, O2PEAK, CO2PEK
      REAL CNRTOT, TEMP, XCO, XCO2, XO2, ARERN2, ARERCO, ARERO2, ARRCO2
      COMMON /STATS/ CNTSTS(NSTS), TSTS(NSTS), XCOSTS(NSTS),
     1          XCO2ST(NSTS), XO2STS(NSTS), N2PSTS(NSTS), COPSTS(NSTS),
     2          O2PSTS(NSTS), CO2PST(NSTS), AN2STS(NSTS), ACOSTS(NSTS),
     3          AO2STS(NSTS), ACO2ST(NSTS)
      REAL CNTSTS, TSTS, XCOSTS, XCO2ST, XO2STS, AN2STS, ACOSTS, AO2STS,
     1    ACO2ST
      REAL N2PSTS, COPSTS, O2PSTS, CO2PST
      COMMON /CONIND/XO2IND(MAXSAM), XCO2ID(MAXSAM)
      REAL XO2IND, XCO2ID
      INTEGER I, J, K, TMPIND(MAXSAM)
      REAL LOLIM, UPLIM

C Sieve N2 arrays:
      I = 0
      UPLIM = TSTS(1) + 4*TSTS(3)
      LOLIM = TSTS(1) - 4*TSTS(3)
      DO 10 J = 1,NUMSAM
        IF(TEMP(J).GT.LOLIM .AND. TEMP(J).LT.UPLIM) THEN
          I = I + 1
C TMPIND contains the temperature index number prior to sieving,
C   it is used in sieving the other species information
          TMPIND(I) = J
          TEMP(I) = TEMP(J)
          CNRTOT(I) = CNRTOT(J)
          N2PEAK(I) = N2PEAK(J)
          ARERN2(I) = ARERN2(J)
        ENDIF
10    CONTINUE
      N2KEPT = I

C Sieve O2 arrays:
C   Only consider concentrations whose corresponding shots were kept in N2 sieve
      I = 0
      K = 1
      UPLIM = XO2STS(1) + 4*XO2STS(3)
      LOLIM = XO2STS(1) - 4*XO2STS(3)
      DO 20 J = 1,O2KEPT
15      IF(XO2IND(J).GT.TMPIND(K).AND.K.LT.N2KEPT) THEN
          K = K + 1
          GOTO 15
        ENDIF
        IF(XO2IND(J).EQ.TMPIND(K).AND.XO2(J).GT.LOLIM
     1     .AND.XO2(J).LT.UPLIM) THEN
          I = I + 1
          XO2(I) = XO2(J)
          O2PEAK(I) = O2PEAK(J)
          ARERO2(I) = ARERO2(J)
        ENDIF
20    CONTINUE
      O2KEPT = I

C Sieve CO2 arrays:
C   Only consider concentrations whose corresponding shots were kept in N2 sieve
      I = 0
      K = 1
```

```
      UPLIM = XCO2ST(1) + 4*XCO2ST(3)
      LOLIM = XCO2ST(1) - 4*XCO2ST(3)
      DO 30 J = 1,CO2KPT
25      IF(XCO2ID(J).GT.TMPIND(K).AND.K.LT.N2KEPT) THEN
          K = K + 1
          GOTO 25
        ENDIF
        IF(XCO2ID(J).EQ.TMPIND(K).AND.XCO2(J).GT.LOLIM
     1     .AND.XCO2(J).LT.UPLIM) THEN
          I = I + 1
          XCO2(I) = XCO2(J)
          CO2PEK(I) = CO2PEK(J)
          ARRCO2(I) = ARRCO2(J)
        ENDIF
30    CONTINUE
      CO2KPT = I

      RETURN
      END

      SUBROUTINE WRTARR(N2KEPT, O2KEPT, CO2KPT, FITFIL, PSPOS)
C***********     WRiTe sieved ARRays to a file     ***********
      INTEGER MAXSAM, N2KEPT, O2KEPT, CO2KPT, PSPOS
      CHARACTER*(*) FITFIL
      PARAMETER (MAXSAM = 1000)
      COMMON /THEDAT/ NUMSAM, CNRTOT(MAXSAM), TEMP(MAXSAM),
XCO(MAXSAM),
     1          XCO2(MAXSAM), XO2(MAXSAM), N2PEAK(MAXSAM),
     2          COPEAK(MAXSAM), O2PEAK(MAXSAM), CO2PEK(MAXSAM),
     3          ARERN2(MAXSAM), ARERCO(MAXSAM), ARERO2(MAXSAM),
     4          ARRCO2(MAXSAM)
      INTEGER NUMSAM
      REAL N2PEAK, COPEAK, O2PEAK, CO2PEK
      REAL CNRTOT, TEMP, XCO, XCO2, XO2, ARERN2, ARERCO, ARERO2, ARRCO2

      CHARACTER*80 OUTFIL
      INTEGER UNINUM, I, PROXIE
      DATA UNINUM/10/

C Write N2-related sieved arrays to file
      OUTFIL = FITFIL(1:PSPOS-1) // '_N2.siev'
      OPEN(UNIT=UNINUM, FILE= OUTFIL)
      WRITE(UNINUM, '(A)') '***** Sieved N2 arrays *****'
      WRITE(UNINUM, 1001) 'CHINRT', 'T(K)', 'N2PEAK', 'ARERN2'
      DO 10 I=1,N2KEPT
        PROXIE = N2PEAK(I)
        WRITE(UNINUM, 2001) CNRTOT(I), TEMP(I), PROXIE, ARERN2(I)
10    CONTINUE
      CLOSE(UNINUM)

C Write CO2-related sieved arrays to file
      OUTFIL = FITFIL(1:PSPOS-1) // '_CO2.siev'
      OPEN(UNIT=UNINUM, FILE= OUTFIL)
      WRITE(UNINUM, '(A)') '***** Sieved CO2 arrays *****'
      WRITE(UNINUM, 1002) 'XCO2', 'CO2PEK', 'ARRCO2'
      DO 20 I=1,CO2KPT
        PROXIE = CO2PEK(I)
        WRITE(UNINUM, 2002)XCO2(I), PROXIE, ARRCO2(I)
20    CONTINUE
      CLOSE(UNINUM)

C Write O2-related sieved arrays to file
      OUTFIL = FITFIL(1:PSPOS-1) // '_O2.siev'
      OPEN(UNIT=UNINUM, FILE= OUTFIL)
      WRITE(UNINUM, '(A)') '***** Sieved O2 arrays *****'
      WRITE(UNINUM, 1003) 'XO2', 'O2PEAK', 'ARERO2'
      DO 30 I=1,O2KEPT
        PROXIE = O2PEAK(I)
        WRITE(UNINUM, 2002)XO2(I), PROXIE, ARERO2(I)
30    CONTINUE
      CLOSE(UNINUM)

1001  FORMAT(5X, A, 7X, A, 7X, A, 2X, A, 3X)
1002  FORMAT(3X, A, 6X, A, 3X, A)
1003  FORMAT(3X, A, 7X, A, 3X, A)
2001  FORMAT(3X, G12.6, 1X, G11.5, 1X, I7, 2X, G10.4)
2002  FORMAT(1X, G10.4, 1X, I7, 2X, 1X, G10.4)

      RETURN
      END
```

APPENDIX C

This appendix presents the experimental CARS temperature data in various forms.

A table is presented of the computed statistics on the experimental CARS temperatures for all combustion cases. Values of averaged gas temperature (<T>) and normalized standard deviations (σ/<T> %) are shown at each sampling location.

Axial and radial profiles of temperature (<T>) are presented starting in pages C-7 and C-9, respectively.

Gas temperature PDFs are presented as a function of radial position beginning in page C-13.

CARS Temperature Statistics for All Combustion Cases

Position		MS65			MS80			HS65			HS80		
Z (mm)	**R (mm)**	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %
5	-3	406	68	17	427	81	19	566	115	20	619	141	23
5	-6	362	62	17	378	59	16	432	106	25	460	117	26
5	0	428	68	16	460	90	20	620	97	16	696	134	19
5	3	395	70	18	402	74	18	544	117	22	605	157	26
5	6	344	63	18	353	65	18	402	101	25	431	115	27
5	9	377	97	26	411	135	33	394	137	35	383	94	25
5	12	1033	307	30	1141	318	28	705	314	45	748	312	42
5	15	1273	144	11	1390	132	9	1111	275	25	1062	231	22
5	18	1310	106	8	1417	126	9	1268	149	12	1355	161	12
5	21	1308	102	8	1408	112	8	1297	124	10	1375	116	8
5	24	1304	105	8	1412	110	8	1297	112	9	1371	130	9
5	27	1310	112	9	1426	121	8	1294	115	9	1372	128	9
5	30	1310	111	8	1418	115	8	1309	111	8	1445	138	10
5	32	1318	109	8	1424	110	8	1317	108	8	1466	142	10
10	-3	459	79	17	528	122	23	629	91	14	681	143	21
10	-6	421	75	18	450	104	23	563	106	19	569	131	23
10	0	493	86	17	567	132	23	650	90	14	719	160	22
10	3	465	83	18	514	128	25	612	98	16	658	131	20
10	6	415	76	18	420	97	23	533	112	21	548	133	24
10	9	418	103	25	414	115	28	491	138	28	490	159	32
10	12	667	276	41	661	288	44	595	247	42	568	268	47
10	15	960	292	30	1014	338	33	754	266	35	726	287	40
10	18	1169	248	21	1266	264	21	1041	289	28	899	318	35
10	21	1274	161	13	1367	167	12	1206	208	17	1250	243	19
10	24	1316	120	9	1410	115	8	1290	128	10	1326	163	12
10	27	1331	107	8	1409	113	8	1320	114	9	1276	172	13
10	30	1332	107	8	1431	104	7	1321	105	8	1464	124	8
10	32	1337	101	8	1425	113	8	1321	111	8	1472	131	9
20	-3	586	96	16	691	141	20	717	103	14	814	198	24
20	-6	549	107	20	648	138	21	678	97	14	715	134	19
20	0	608	96	16	711	148	21	720	89	12	835	195	23
20	3	592	98	17	680	145	21	707	95	13	797	195	24
20	6	566	108	19	627	143	23	668	90	14	723	154	21
20	9	534	132	25	597	162	27	631	107	17	675	144	21
20	12	555	176	32	664	232	35	620	132	21	621	141	23
20	15	660	230	35	780	263	34	676	164	24	690	174	25
20	18	792	240	30	931	281	30	786	201	26	792	223	28
20	21	946	259	27	1097	281	26	917	230	25	958	237	25
20	24	1088	253	23	1284	248	19	1107	233	21	1108	244	22
20	27	1246	194	16	1381	181	13	1225	194	16	1287	217	17
20	30	1303	157	12	1451	142	10	1304	143	11	1379	177	13
20	32	1342	117	9	1472	132	9	1327	125	9	1403	149	11
30	-3	693	107	15	910	256	28				971	230	24
30	-6	669	110	16	846	221	26				899	222	25
30	0	696	95	14	933	253	27				994	247	25
30	3	689	103	15	906	251	28				982	233	24
30	6	670	106	16	813	201	25				911	217	24
30	9	642	108	17	755	170	23				799	169	21
30	12	644	131	20	731	176	24				759	144	19
30	15	693	179	26	761	195	26				750	141	19

CARS Temperature Statistics for All Combustion Cases

Position		MS65			MS80			HS65			HS80		
Z (mm)	R (mm)	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %
30	18	747	190	25	857	212	25				787	168	21
30	21	844	215	25	962	247	26	867	197	23	875	201	23
30	24	957	235	25	1094	261	24	998	229	23	959	218	23
30	27	1079	238	22	1227	251	20	1108	231	21	1102	244	22
30	30	1198	223	19	1350	222	16	1219	205	17	1232	239	19
30	32	1250	200	16	1387	208	15	1264	189	15	1328	220	17
40	-3	777	124	16	1087	284	26	927	155	17	1102	234	21
40	-6	746	116	16	1026	281	27	897	144	16	1026	244	24
40	0	785	119	15	1148	310	27	927	155	17	1214	258	21
40	3	768	108	14	1070	290	27	905	142	16	1177	273	23
40	6	755	109	14	975	254	26	914	162	18	1107	269	24
40	9	739	111	15	936	244	26	869	133	15	1042	274	26
40	12	733	124	17	873	208	24	829	124	15	936	221	24
40	15	751	147	20	873	203	23	823	129	16	891	199	22
40	18	789	169	21	884	202	23	827	133	16	855	167	20
40	21	859	203	24	932	220	24	876	158	18	863	180	21
40	24	930	208	22	1022	240	24	952	187	20	931	213	23
40	27	1049	237	23	1123	266	24	1029	204	20	994	210	21
40	30	1138	236	21	1244	258	21	1110	203	18	1094	229	21
40	32	1208	213	18	1330	258	19	1179	206	18	1152	234	20
50	-3	851	153	18	1326	294	22	1056	193	18	1265	246	19
50	-6	831	143	17	1254	299	24	1016	185	18	1212	249	21
50	0	849	149	17	1330	301	23	1068	194	18	1332	249	19
50	3	858	156	18	1296	300	23	1049	186	18	1351	242	18
50	6	839	149	18	1229	299	24	1022	180	18	1337	245	18
50	9	815	134	16	1158	297	26	966	169	17	1273	256	20
50	12	798	130	16	1089	286	26	937	161	17	1192	262	22
50	15	789	131	17	1002	252	25	902	143	16	1138	263	23
50	18	785	131	17	991	234	24	880	129	15	1070	240	22
50	21	819	162	20	995	222	22	888	141	16	1019	216	21
50	24	850	177	21	1020	231	23	907	153	17	981	196	20
50	27	919	207	23	1083	248	23	965	179	19	984	190	19
50	30	995	217	22	1188	265	22	1029	197	19	1038	206	20
50	32	1039	229	22	1243	264	21	1090	205	19	1065	198	19
60	-3	942	186	20	1229	271	22	1181	194	16	1495	197	13
60	-6	915	177	19	1196	259	22	1148	200	17	1459	219	15
60	0	952	183	19	1254	244	19	1179	191	16	1533	165	11
60	3	949	186	20	1264	254	20	1187	195	16	1514	164	11
60	6	942	192	20	1249	252	20	1139	190	17	1468	199	14
60	9	928	184	20	1215	267	22	1099	192	17	1461	202	14
60	12	887	165	19	1140	256	22	1047	192	18	1429	234	16
60	15	873	157	18	1092	251	23	998	174	17	1383	246	18
60	18	870	151	17	1082	244	23	973	162	17	1307	267	20
60	21	852	135	16	1046	231	22	958	159	17	1239	275	22
60	24	882	161	18	1036	215	21	952	147	15	1158	234	20
60	27	916	190	21	1046	215	21	969	154	16	1130	230	20
60	30	956	198	21	1071	215	20	991	174	18	1098	213	19
60	32	1007	218	22	1104	234	21	1031	176	17	1098	210	19
70	-3	1027	205	20	1422	217	15	1282	185	14	1563	138	9
70	-6	996	188	19	1416	234	17	1254	185	15	1540	165	11

CARS Temperature Statistics for All Combustion Cases

Position		MS65			MS80			HS65			HS80		
Z (mm)	R (mm)	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %
70	0	1064	210	20	1449	208	14	1306	180	14	1524	135	9
70	3	1059	205	19	1437	195	14	1283	184	14	1559	127	8
70	6	1062	215	20	1408	221	16	1269	191	15	1574	128	8
70	9	1052	210	20	1403	222	16	1239	201	16	1571	135	9
70	12	1021	204	20	1339	244	18	1203	200	17	1574	149	9
70	15	972	181	19	1296	247	19	1156	195	17	1568	170	11
70	18	946	180	19	1260	256	20	1115	194	17	1534	189	12
70	21	934	171	18	1212	255	21	1090	191	18	1504	211	14
70	24	919	157	17	1182	245	21	1066	177	17	1508	219	15
70	27	929	166	18	1172	244	21	1046	163	16	1313	257	20
70	30	969	188	19	1136	235	21	1033	171	17	1294	255	20
70	32	972	187	19	1130	227	20	1028	159	15	1259	233	19
80	-3	1143	215	19	1574	191	12	1385	154	11	1591	119	7
80	-6	1099	216	20	1529	218	14	1371	170	12	1596	129	8
80	0	1178	202	17	1542	186	12	1383	164	12	1588	118	7
80	3	1169	215	18	1549	189	12	1394	161	12	1601	116	7
80	6	1159	200	17	1544	175	11	1362	165	12	1589	113	7
80	9	1133	216	19	1553	187	12	1369	163	12	1585	114	7
80	12	1130	210	19	1500	210	14	1318	192	15	1579	125	8
80	15	1104	212	19	1480	229	16	1304	186	14	1574	142	9
80	18	1057	203	19	1481	238	16	1255	200	16			
80	21	1028	203	20	1407	261	19	1222	188	15	1594	140	9
80	24	1013	197	19	1395	274	20	1185	199	17	1585	157	10
80	27	1014	180	18	1386	272	20	1138	185	16	1509	199	13
80	30	1000	177	18	1357	272	20	1129	184	16	1504	216	14
80	32	998	177	18	1337	276	21	1125	180	16	1471	216	15
90	-3	1239	193	16	1570	102	6	1421	136	10	1590	111	7
90	-6	1202	204	17	1557	114	7	1415	117	8	1595	121	8
90	0	1257	191	15	1570	103	7	1424	134	9	1588	112	7
90	3	1256	192	15	1572	95	6	1422	129	9	1597	117	7
90	6	1258	186	15	1578	98	6	1422	137	10	1592	115	7
90	9	1247	188	15	1586	114	7	1422	149	11	1595	114	7
90	12	1239	194	16	1597	115	7	1393	157	11	1590	119	7
90	15	1237	197	16	1581	129	8	1381	153	11	1615	121	7
90	18	1194	202	17	1568	141	9	1362	173	13	1616	131	8
90	21	1157	211	18	1560	150	10	1347	183	14	1618	134	8
90	24	1144	204	18	1564	173	11	1319	187	14	1633	136	8
90	27	1117	200	18	1581	177	11	1293	187	14	1602	152	9
90	30	1097	205	19	1542	206	13	1280	197	15	1569	174	11
90	32	1075	191	18	1506	218	15	1254	188	15	1549	159	10
100	-3	1343	164	12	1554	91	6	1451	111	8	1599	130	8
100	-6	1345	171	13	1549	99	6	1445	114	8	1606	122	8
100	0	1346	157	12	1556	99	6	1451	100	7	1598	129	8
100	3	1337	158	12	1557	98	6	1448	104	7	1604	134	8
100	6	1324	161	12	1563	96	6	1452	127	9	1602	127	8
100	9	1306	174	13	1542	104	7	1436	123	9	1602	125	8
100	12	1282	180	14	1554	109	7	1427	131	9	1603	130	8
100	15	1257	181	14	1560	102	7	1430	135	9	1613	128	8
100	18	1235	179	14	1529	119	8	1420	146	10	1618	126	8
100	21	1221	199	16	1529	140	9	1411	147	10	1609	134	8

CARS Temperature Statistics for All Combustion Cases

Position		MS65			MS80			HS65			HS80		
Z (mm)	R (mm)	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %
100	24	1182	191	16	1534	141	9	1396	154	11	1608	123	8
100	27	1170	194	17	1536	149	10	1378	162	12	1622	154	9
100	30	1152	191	17	1563	140	9	1373	164	12	1615	156	10
100	32	1137	186	16	1555	135	9	1362	163	12	1596	150	9
112	-3	1337	130	10	1480	132	9	1448	95	7	1596	134	8
112	-6	1324	132	10	1499	124	8	1458	91	6	1606	134	8
112	0	1344	126	9	1491	121	8	1450	92	6	1581	129	8
112	3	1341	123	9	1497	119	8	1457	90	6	1565	116	7
112	6	1333	129	10	1484	131	9	1449	97	7	1541	121	8
112	9	1326	136	10	1490	133	9	1448	101	7	1552	103	7
112	12	1349	134	10	1474	147	10	1450	100	7	1562	111	7
112	15	1340	142	11	1493	141	9	1456	107	7	1573	113	7
112	18	1331	148	11	1513	137	9	1463	124	8	1565	100	6
112	21	1325	155	12	1489	145	10	1449	139	10	1559	112	7
112	24	1283	162	13	1492	146	10	1436	113	8	1566	111	7
112	27	1265	172	14	1506	133	9	1431	124	9	1631	155	10
112	30	1269	172	14	1517	140	9	1433	136	10	1636	161	10
112	32	1244	179	14	1520	131	9	1418	123	9	1537	152	10
125	-3	1415	87	6	1577	132	8	1454	84	6	1529	91	6
125	-6	1421	86	6	1575	133	8	1455	83	6	1532	99	6
125	0	1421	88	6	1568	129	8	1465	87	6	1539	105	7
125	3	1412	95	7	1555	118	8	1484	103	7	1543	101	7
125	6	1405	96	7	1554	118	8	1464	83	6	1541	98	6
125	9	1412	97	7	1580	128	8	1461	85	6	1541	100	6
125	12	1392	105	8	1573	114	7	1457	88	6	1549	101	7
125	15	1398	107	8	1572	114	7	1461	88	6	1562	104	7
125	18	1392	123	9	1573	110	7	1477	109	7	1559	103	7
125	21	1393	112	8	1575	109	7	1480	109	7	1550	102	7
125	24	1383	117	8	1571	106	7	1465	91	6	1550	100	6
125	27	1383	133	10	1576	107	7	1443	88	6	1613	151	9
125	30	1365	139	10	1572	109	7	1446	92	6	1581	140	9
125	32	1358	146	11	1579	99	6	1441	94	7	1583	151	10
138	-3	1419	83	6	1536	78	5	1452	83	6	1528	102	7
138	-6	1422	81	6	1538	78	5	1460	79	5	1527	102	7
138	0	1424	78	6	1541	75	5	1450	80	6	1519	107	7
138	3	1421	78	6	1537	92	6	1453	82	6	1522	103	7
138	6	1423	78	5	1540	94	6	1460	85	6	1519	109	7
138	9	1424	79	6	1544	91	6	1457	84	6	1527	105	7
138	12	1424	79	6	1544	90	6	1467	82	6	1529	111	7
138	15	1425	80	6	1536	98	6	1446	79	5	1537	117	8
138	18	1423	89	6	1555	93	6	1466	86	6	1526	118	8
138	21	1419	93	7	1552	97	6	1468	89	6	1507	125	8
138	24	1430	90	6	1557	89	6	1476	106	7	1506	135	9
138	27	1421	92	6	1562	90	6	1482	108	7	1559	133	9
138	30	1405	106	8	1565	94	6	1475	106	7	1570	139	9
138	32	1382	120	9	1549	98	6				1571	138	9
150	-3	1430	81	6	1538	78	5	1494	116	8	1576	137	9
150	-6	1435	78	5	1530	76	5	1468	81	5	1569	140	9
150	0	1428	81	6	1536	77	5	1458	84	6	1574	142	9
150	3	1428	78	5	1543	90	6	1458	85	6	1534	141	9

CARS Temperature Statistics for All Combustion Cases													
Position		MS65			MS80			HS65			HS80		
Z (mm)	R (mm)	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %	<T> (K)	s	s/<T> %
150	6	1423	79	6	1539	79	5	1487	109	7	1510	128	8
150	9	1425	80	6	1552	93	6	1467	83	6	1466	122	8
150	12	1420	77	5	1544	74	5	1469	84	6	1466	118	8
150	15	1424	78	5	1558	93	6	1483	106	7	1519	117	8
150	18	1424	81	6	1560	90	6	1489	123	8	1509	121	8
150	21	1422	80	6	1562	96	6	1497	136	9	1504	128	8
150	24	1424	80	6	1556	94	6	1504	135	9	1493	127	8
150	27	1428	81	6	1571	105	7	1498	125	8	1545	135	9
150	30	1430	80	6	1567	99	6	1487	112	8	1540	128	8
150	32										1538	128	8
162	-3	1423	72	5				1470	82	6	1496	128	9
162	-6	1419	71	5	1540	94	6	1474	79	5	1538	149	10
162	0	1426	73	5	1540	99	6	1458	77	5	1505	106	7
162	3	1419	74	5				1467	77	5	1507	105	7
162	6	1421	75	5	1553	99	6	1474	80	5	1520	117	8
162	9	1425	76	5				1474	82	6	1521	113	7
162	12	1429	71	5	1559	97	6	1472	82	6	1520	114	7
162	15	1419	75	5				1475	81	5	1505	110	7
162	18	1420	72	5	1549	94	6	1477	80	5	1520	116	8
162	21	1429	75	5				1478	81	5	1520	119	8
162	24	1425	73	5	1569	94	6	1471	81	6	1512	111	7
162	27	1425	76	5				1482	74	5	1566	128	8
162	30	1419	81	6	1577	98	6	1476	79	5	1551	121	8
175	-3	1418	73	5				1437	82	6	1534	147	10
175	-6	1420	72	5	1546	96	6	1448	82	6	1535	139	9
175	0	1422	69	5	1534	93	6	1450	83	6	1550	139	9
175	3	1412	74	5				1448	79	5	1543	147	10
175	6	1411	77	5	1546	91	6	1451	79	5	1525	149	10
175	9	1412	77	5				1456	79	5	1515	171	11
175	12	1419	78	6	1550	96	6	1462	82	6	1521	157	10
175	15	1417	77	5				1461	77	5	1481	173	12
175	18	1421	75	5	1550	96	6	1471	80	5	1500	179	12
175	21	1420	77	5				1469	81	6	1532	172	11
175	24	1421	76	5	1554	94	6	1472	81	5	1524	164	11
175	27	1422	75	5				1473	82	6	1558	124	8
175	30	1420	78	6	1558	98	6	1470	79	5	1543	122	8
188	-6	1409	73	5				1439	84	6	1475	172	12
188	0	1412	77	5				1442	82	6	1473	168	11
188	6	1414	73	5				1442	80	6	1477	178	12
188	12	1418	75	5				1446	82	6	1501	179	12
188	18	1417	71	5				1450	79	5	1494	162	11
188	24	1414	80	6				1445	86	6	1505	160	11
188	30	1419	78	5				1438	90	6	1528	115	8
200	-6	1401	74	5	1519	94	6	1383	105	8	1551	128	8
200	0	1403	75	5	1518	93	6	1395	103	7	1578	147	9
200	6	1414	72	5	1528	96	6	1432	84	6	1554	136	9
200	12	1421	72	5	1540	91	6	1425	86	6	1562	134	9
200	18	1417	73	5	1539	95	6	1437	78	5	1558	144	9
200	24	1417	73	5	1536	91	6	1430	88	6	1540	131	8
200	30	1413	75	5	1544	90	6	1439	86	6	1523	127	8

CARS Temperature Statistics for All Combustion Cases													
Position		**MS65**			**MS80**			**HS65**			**HS80**		
Z (mm)	**R (mm)**	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %	**<T> (K)**	s	s/<T> %
212	-6	1404	76	5				1435	74	5	1529	155	10
212	0	1397	75	5				1388	104	7	1495	143	10
212	6	1403	74	5				1421	73	5	1520	141	9
212	12	1396	76	5				1421	76	5	1516	159	10
212	18	1399	76	5				1427	77	5	1517	144	9
212	24	1407	74	5				1430	79	6	1491	128	9
212	30	1403	78	6				1430	76	5	1447	149	10
225	-6	1398	73	5	1510	82	5	1426	77	5	1523	145	10
225	0	1401	73	5	1511	82	5	1412	75	5	1528	143	9
225	6	1400	72	5	1512	81	5	1396	79	6	1511	147	10
225	12	1407	76	5	1511	81	5	1416	76	5	1507	124	8
225	18	1415	74	5	1511	86	6	1412	80	6	1505	130	9
225	24	1411	73	5	1519	84	6	1419	82	6	1475	131	9
225	30	1406	78	6	1524	84	6	1415	77	5	1464	133	9
238	-6	1402	72	5				1426	78	5	1497	133	9
238	0	1408	71	5				1423	76	5	1495	138	9
238	6	1408	73	5				1418	75	5	1472	146	10
238	12	1407	73	5				1428	73	5	1486	133	9
238	18	1407	74	5				1418	77	5	1482	129	9
238	24	1401	75	5				1417	78	5	1470	136	9
238	30	1395	78	6				1407	82	6	1428	137	10
250	-6	1392	71	5	1498	88	6	1416	75	5	1473	139	9
250	0	1394	75	5	1499	89	6	1412	73	5	1571	150	10
250	6	1389	75	5	1499	87	6	1413	71	5	1564	150	10
250	12	1394	74	5	1500	85	6	1415	77	5	1554	151	10
250	18	1391	72	5	1499	80	5	1415	80	6	1540	141	9
250	24	1391	75	5	1507	83	5	1414	78	5	1519	133	9
250	30	1369	79	6	1503	81	5	1399	80	6	1497	138	9

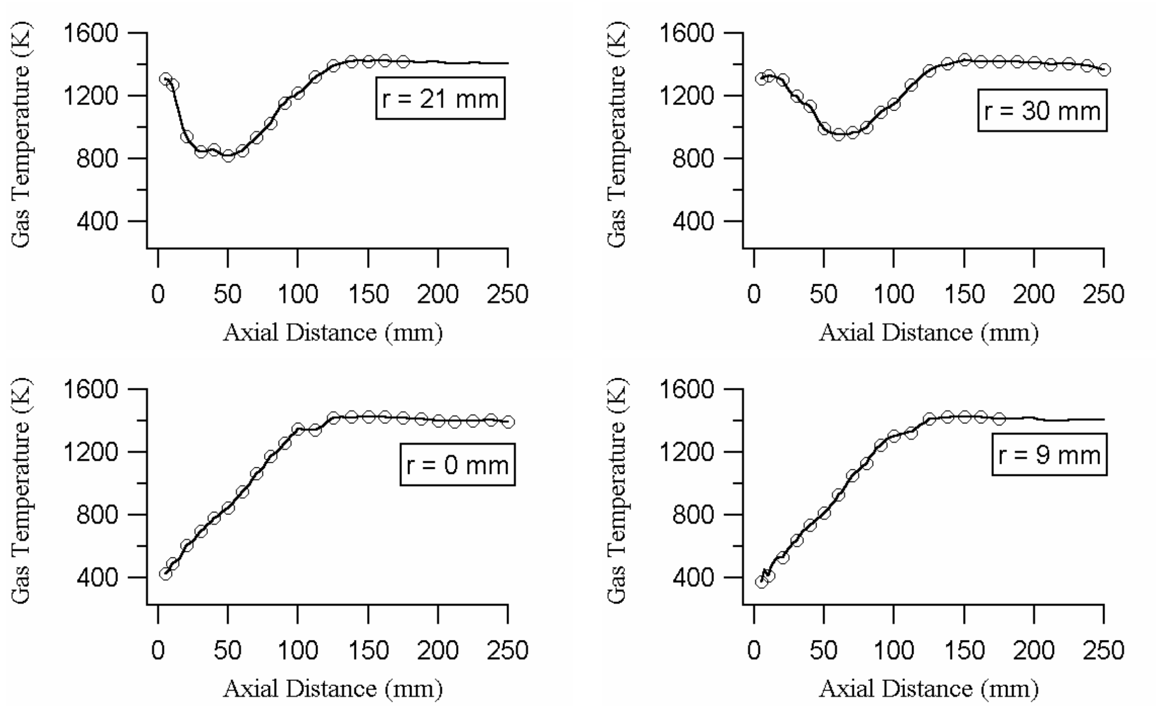

Figure C.1. Axial temperature profiles at selected radial positions for the MS65 case.

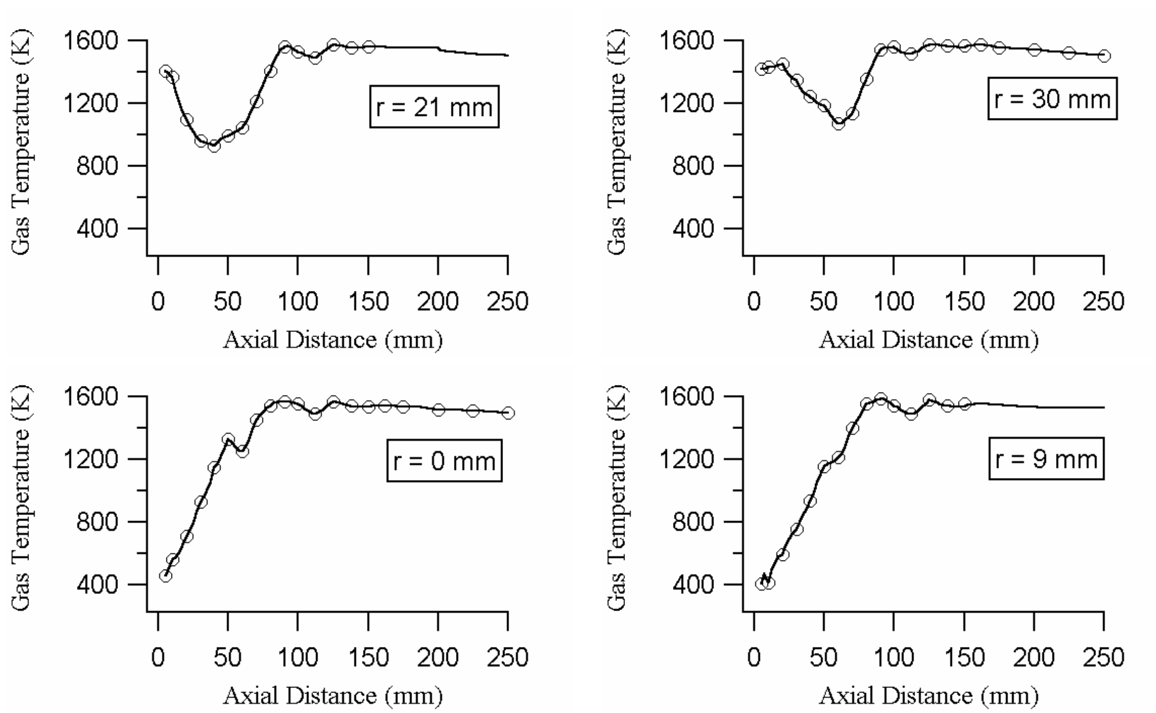

Figure C.2. Axial temperature profiles at selected radial positions for the MS80 case.

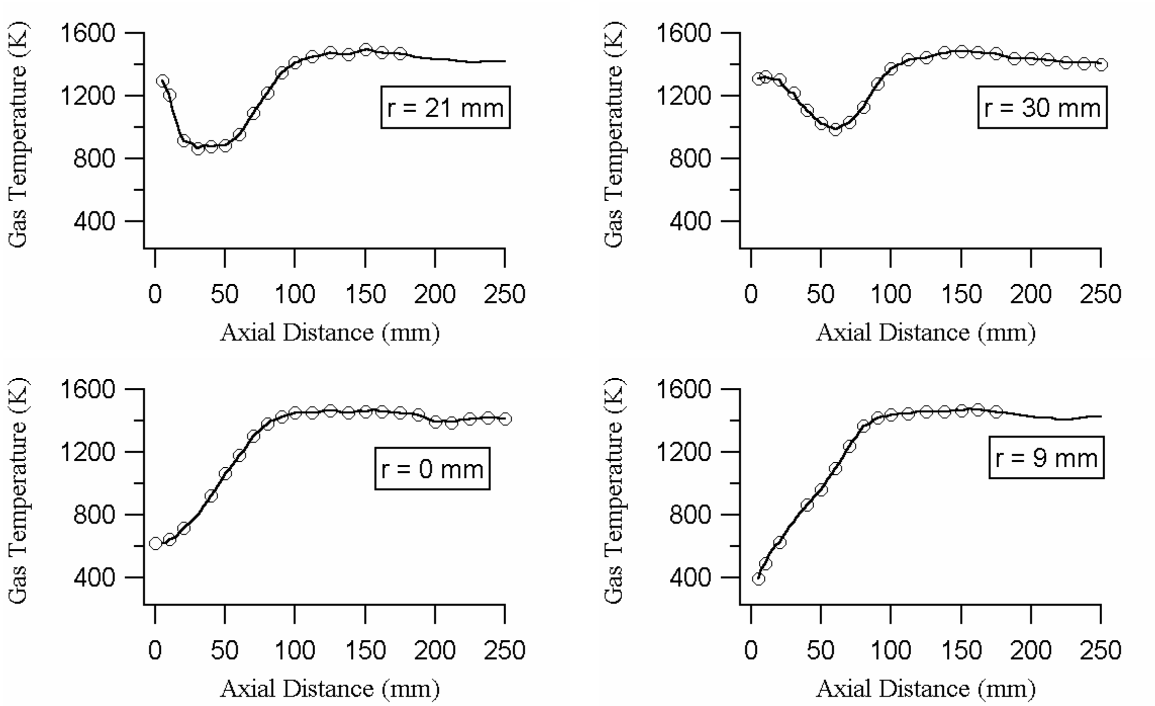

Figure C.3. Axial temperature profiles at selected radial positions for the HS65 case.

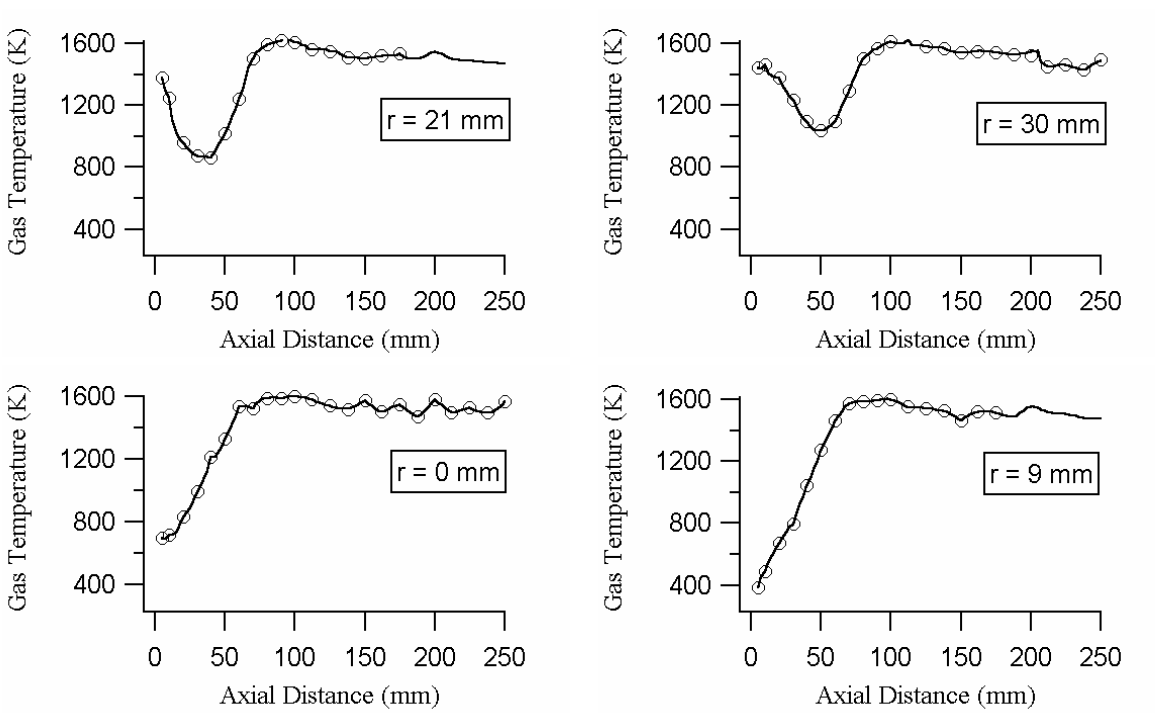

Figure C.4. Axial temperature profiles at selected radial positions for the HS80 case.

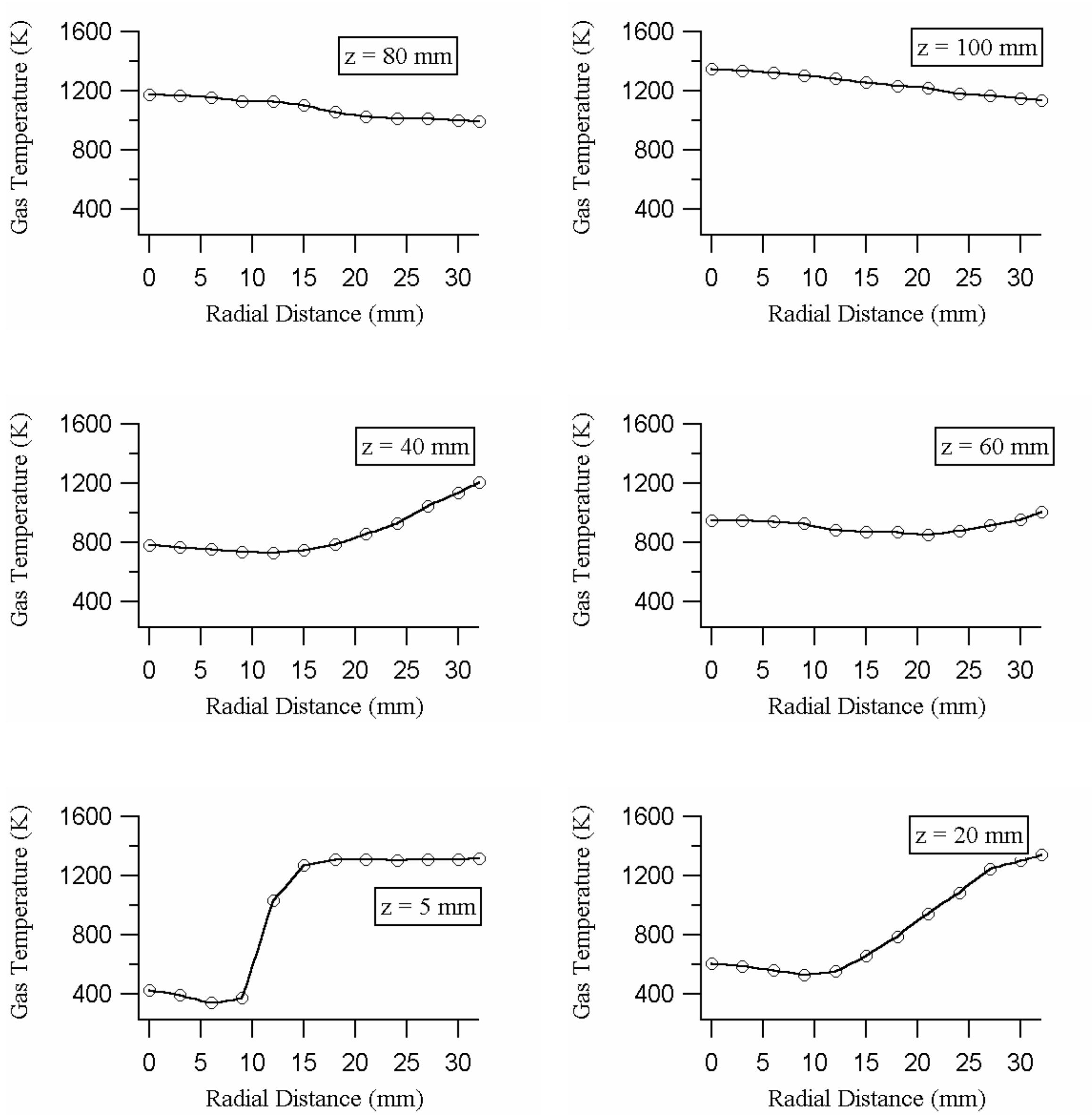

Figure C.5. Radial temperature profiles at selected axial positions for the MS65 case.

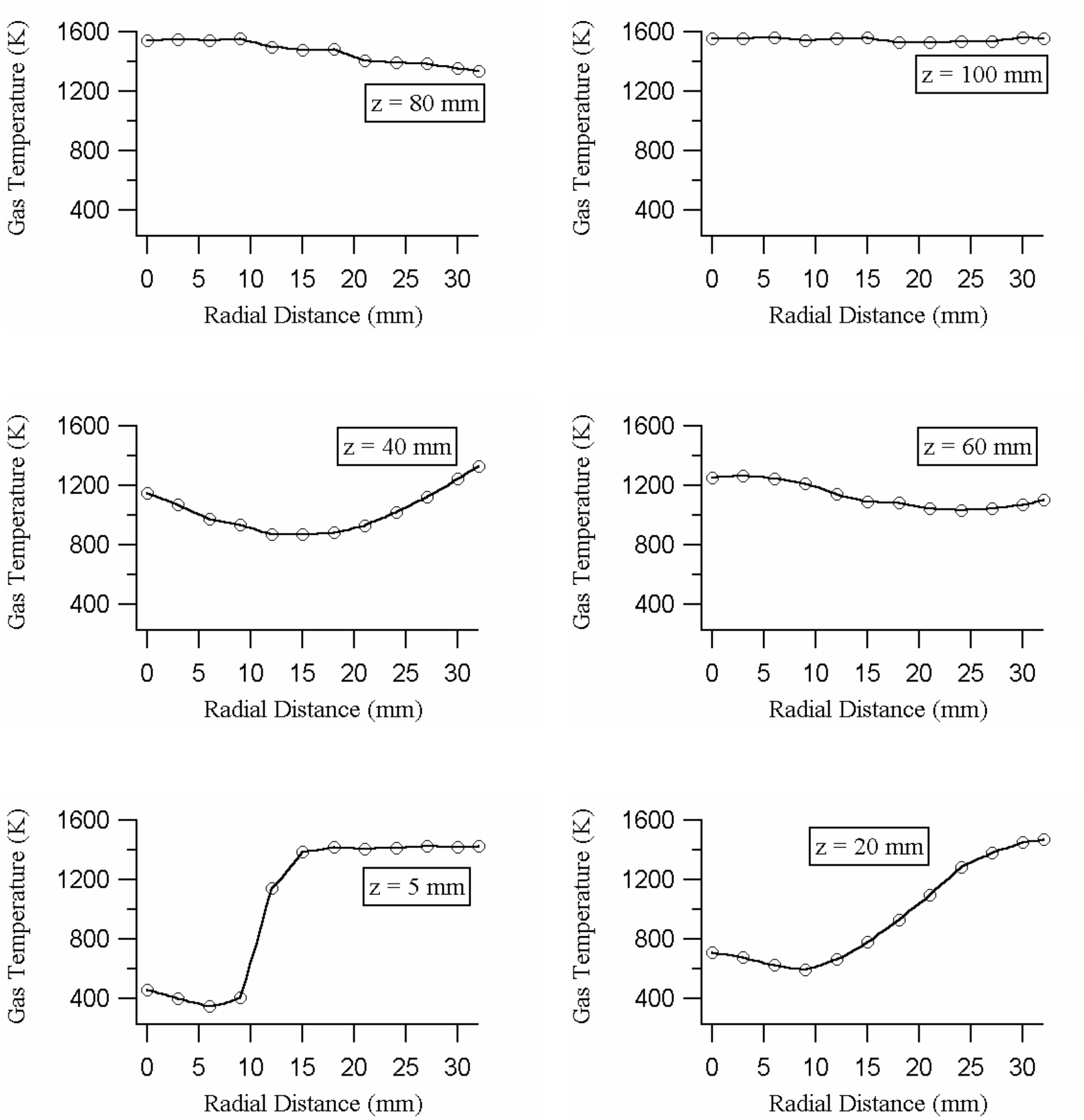

Figure C.6. Radial temperature profiles at selected axial positions for the MS80 case.

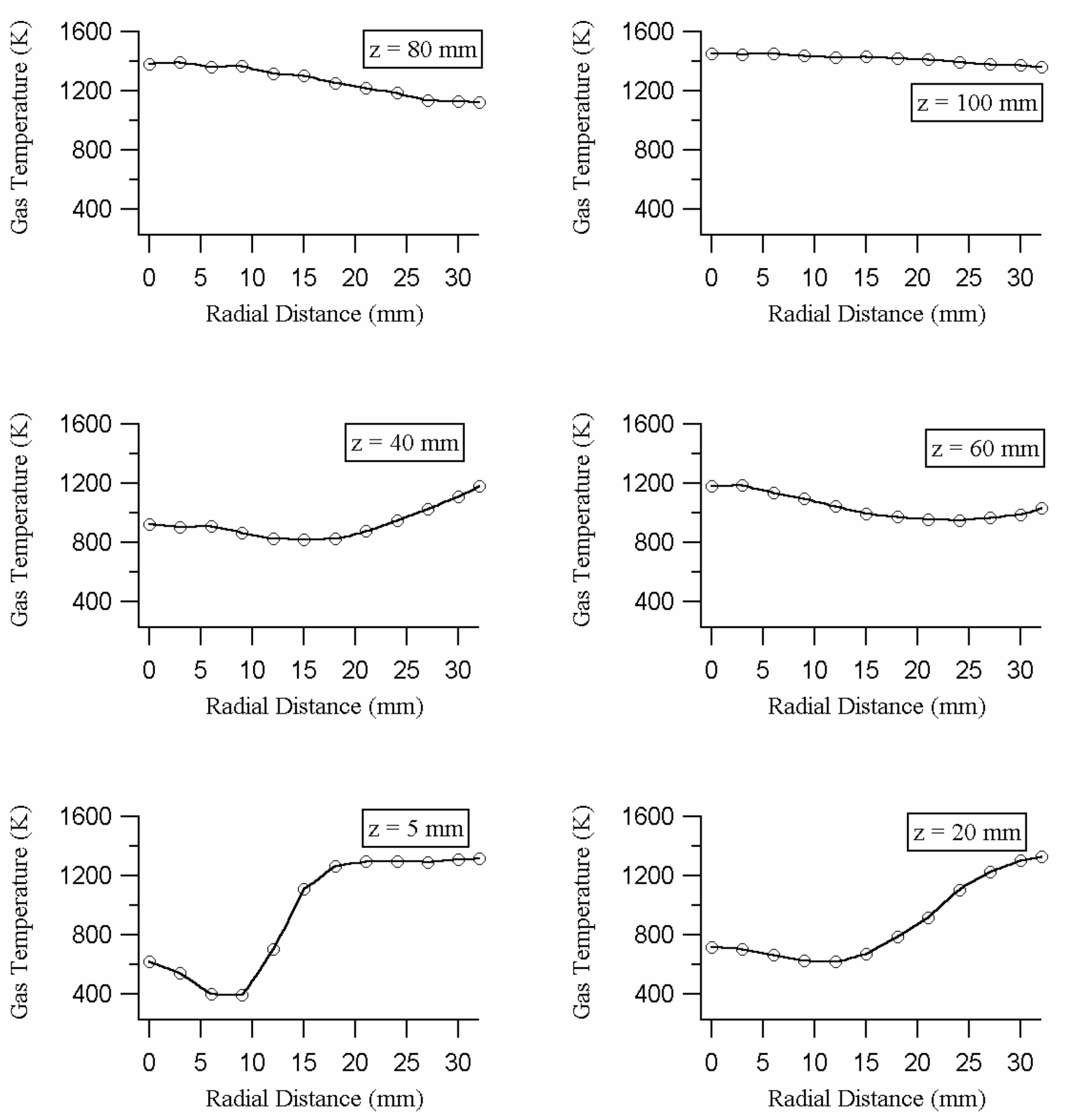

Figure C.7. Radial temperature profiles at selected axial positions for the HS65 case.

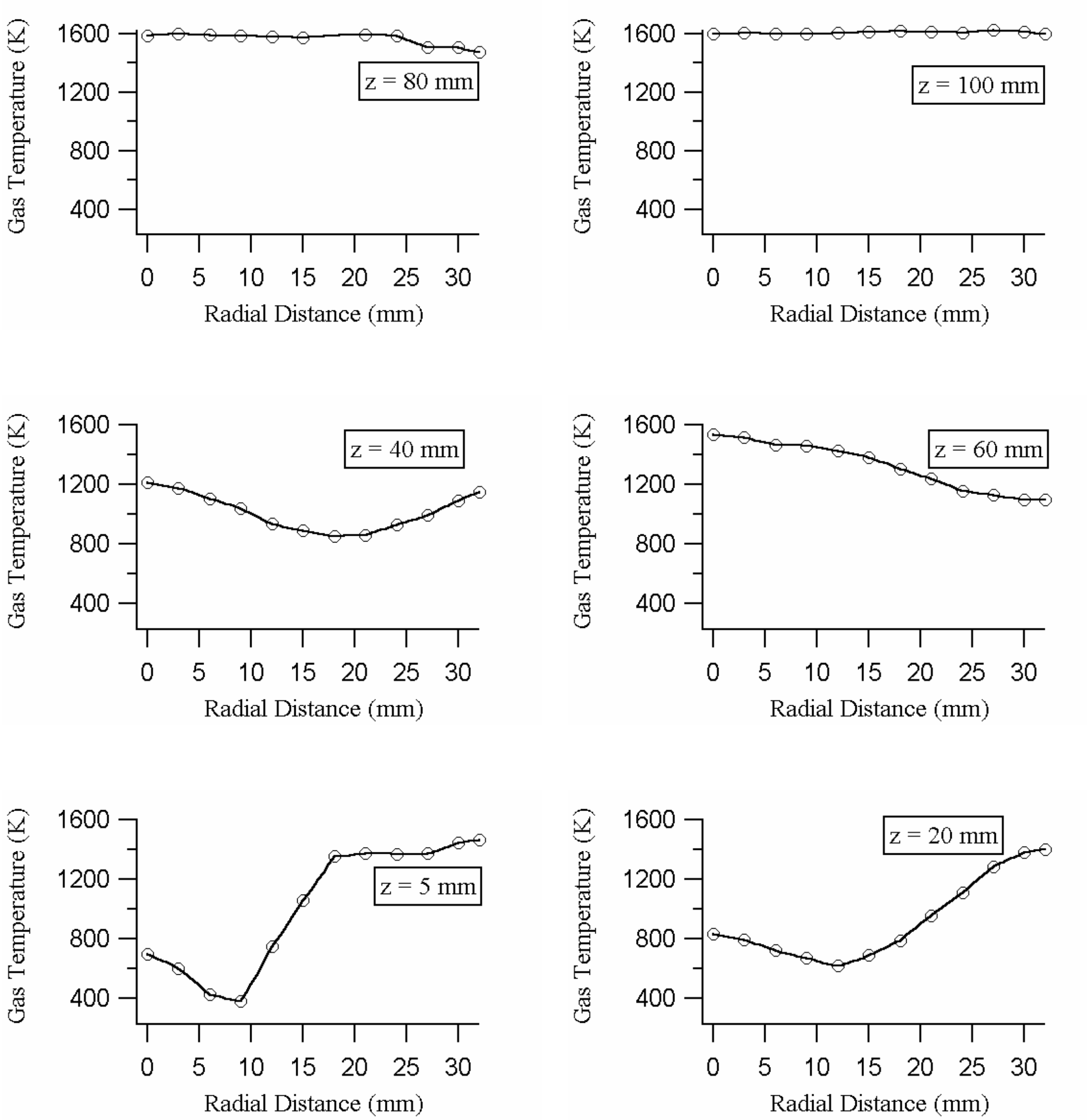

Figure C.8. Radial temperature profiles at selected axial positions for the HS80 case.

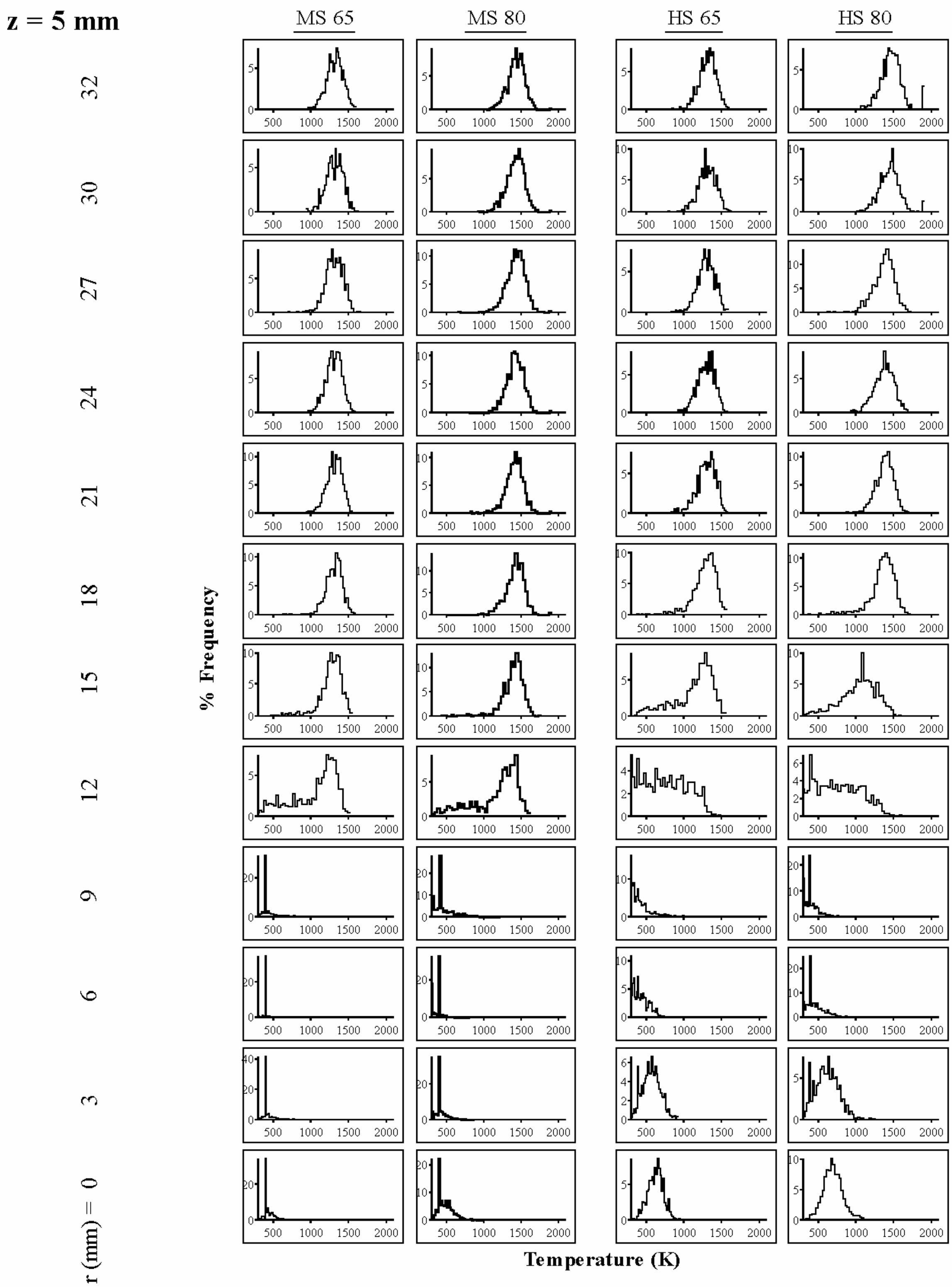

Figure C.9. Gas temperature PDFs as a function of radial position for z = 5 mm.

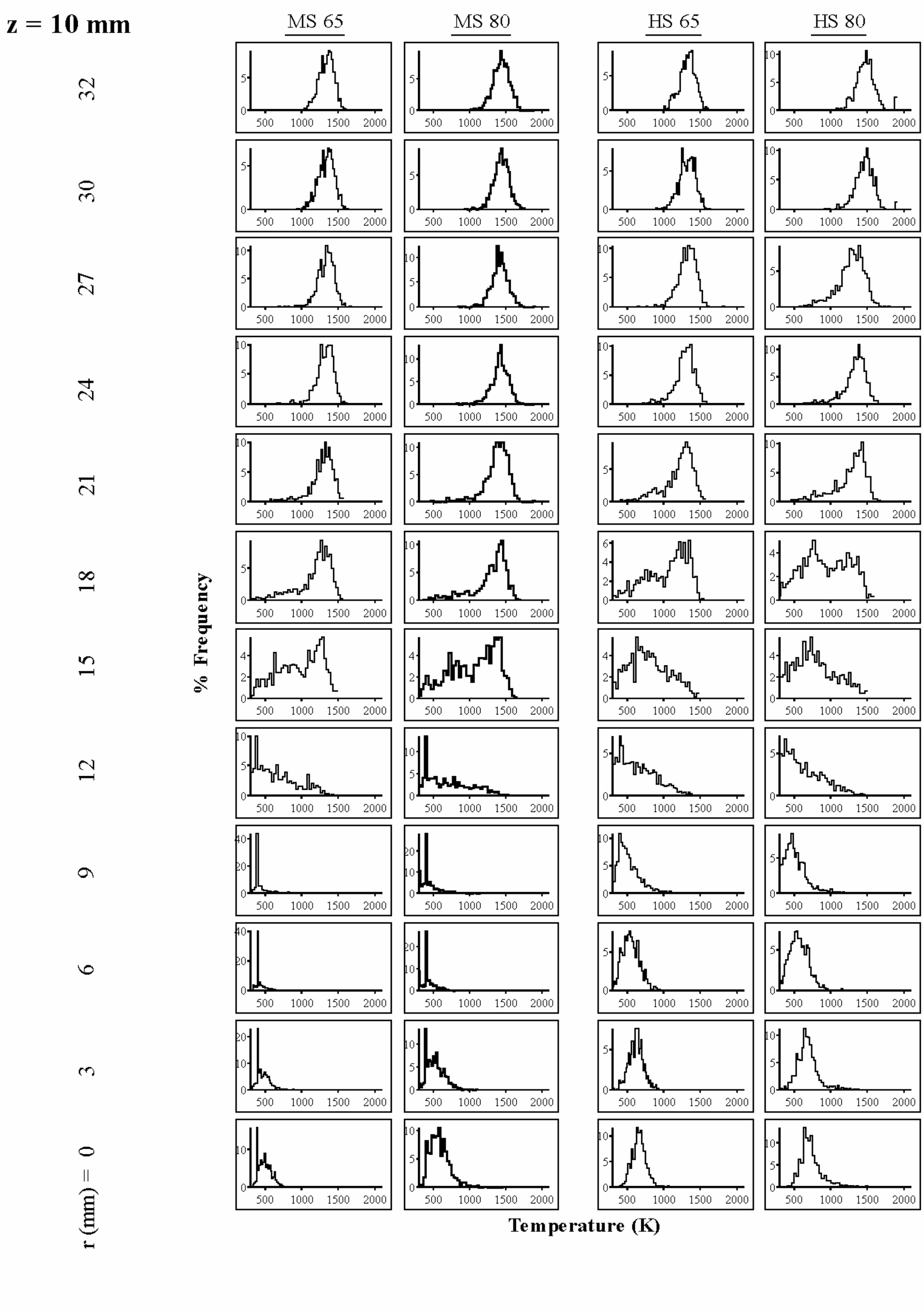

Figure C.10. Gas temperature PDFs as a function of radial position for z = 10 mm.

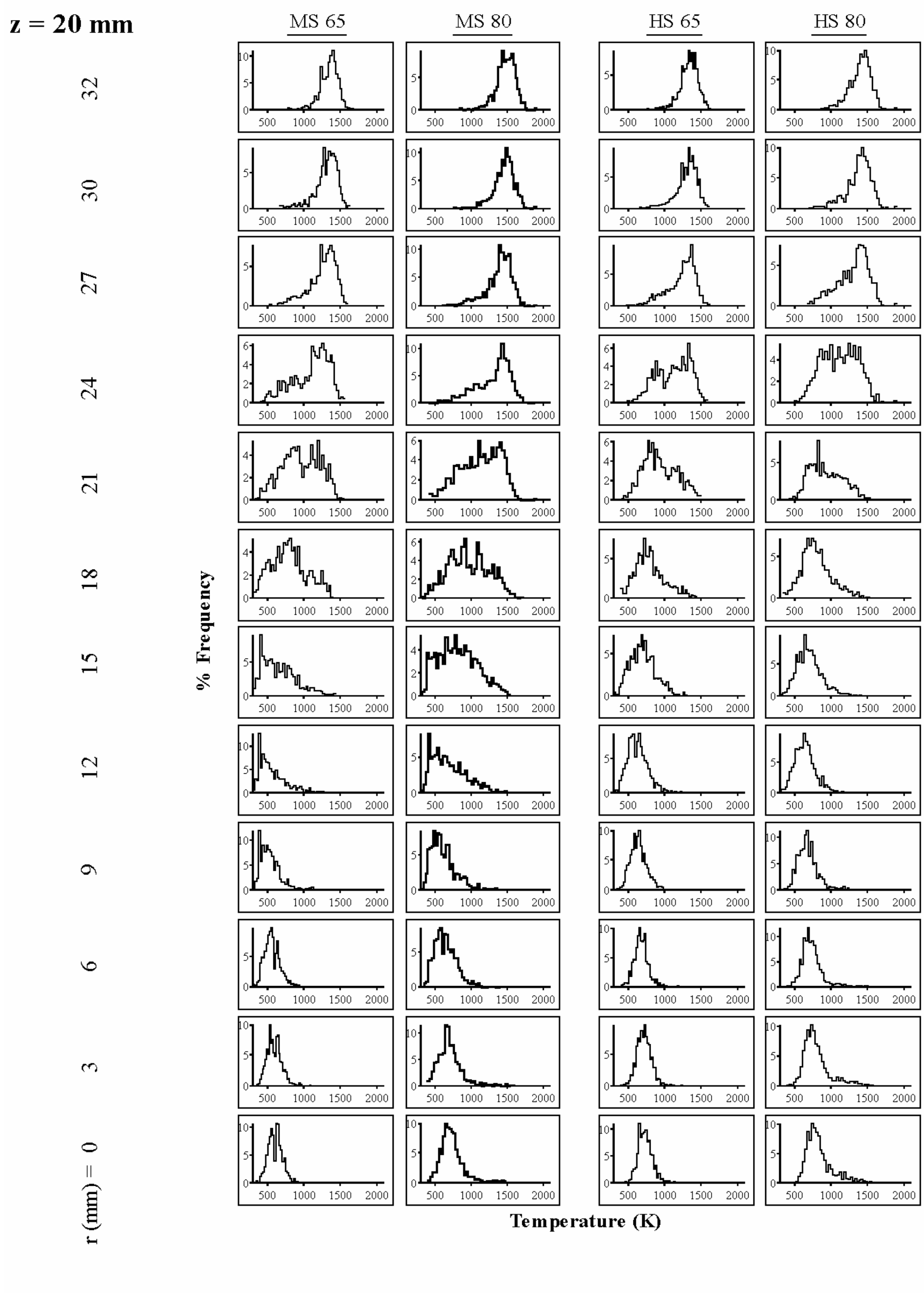

Figure C.11. Gas temperature PDFs as a function of radial position for z = 20 mm.

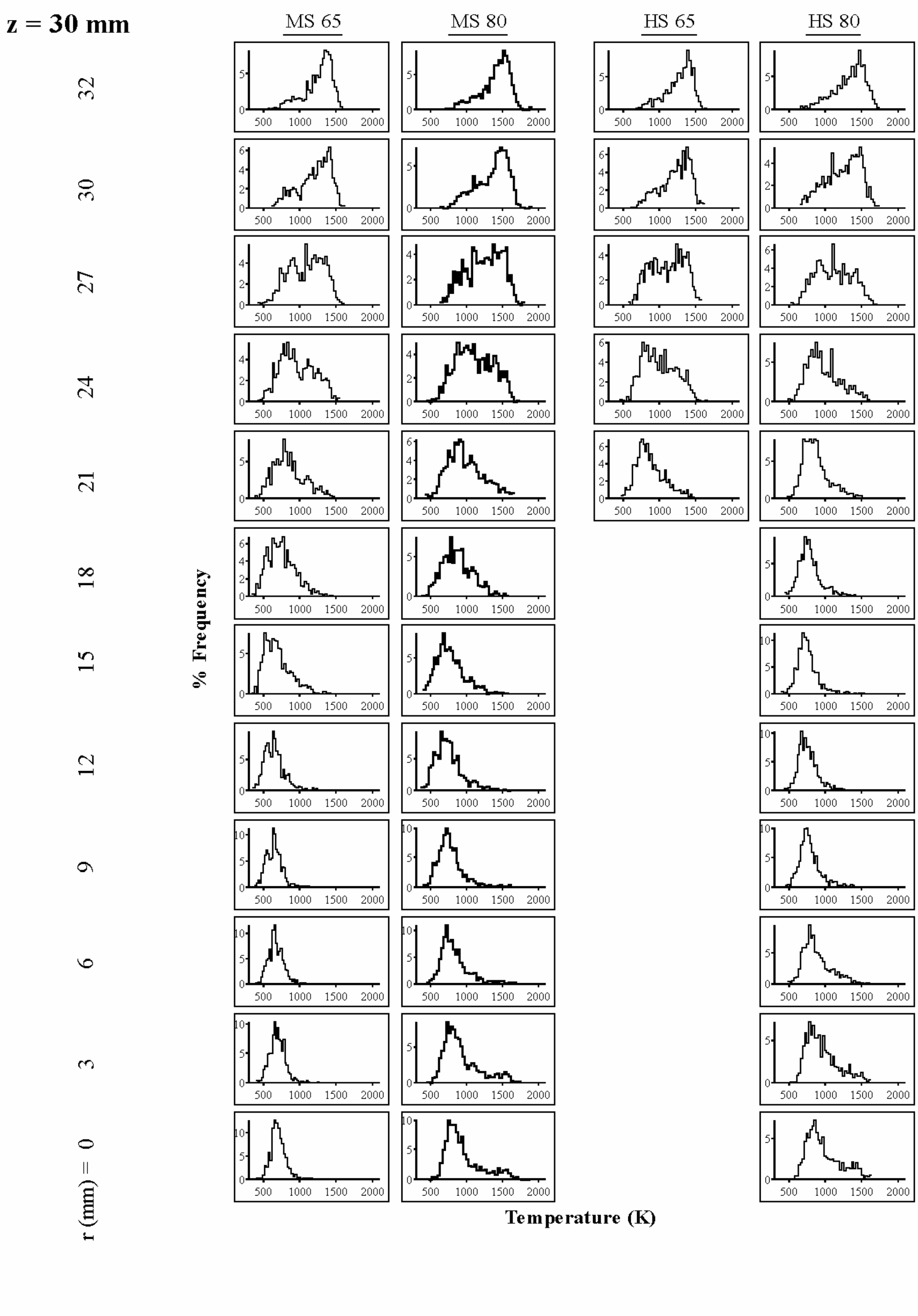

Figure C.12. Gas temperature PDFs as a function of radial position for z = 30 mm.

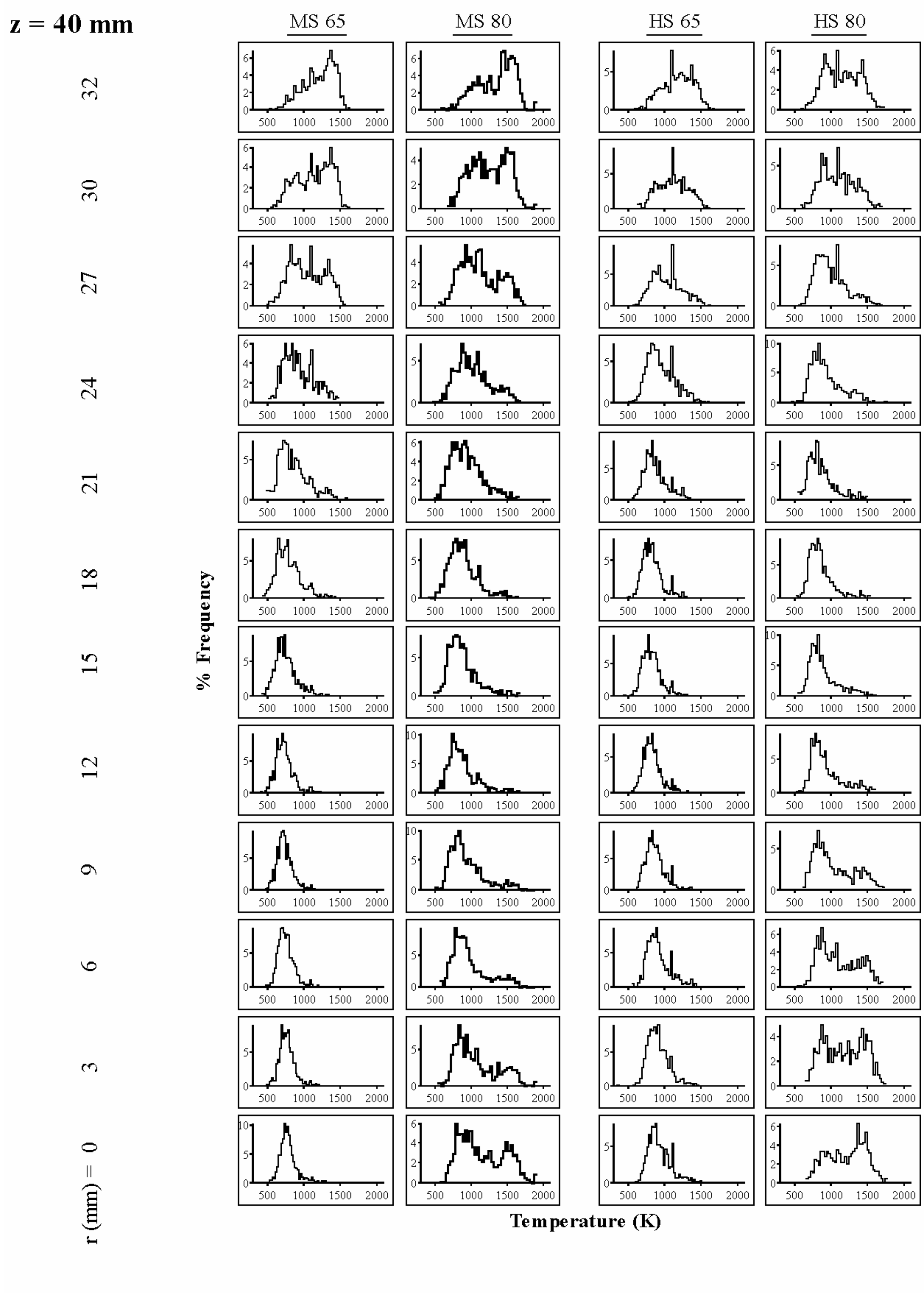

Figure C.13. Gas temperature PDFs as a function of radial position for z = 40 mm.

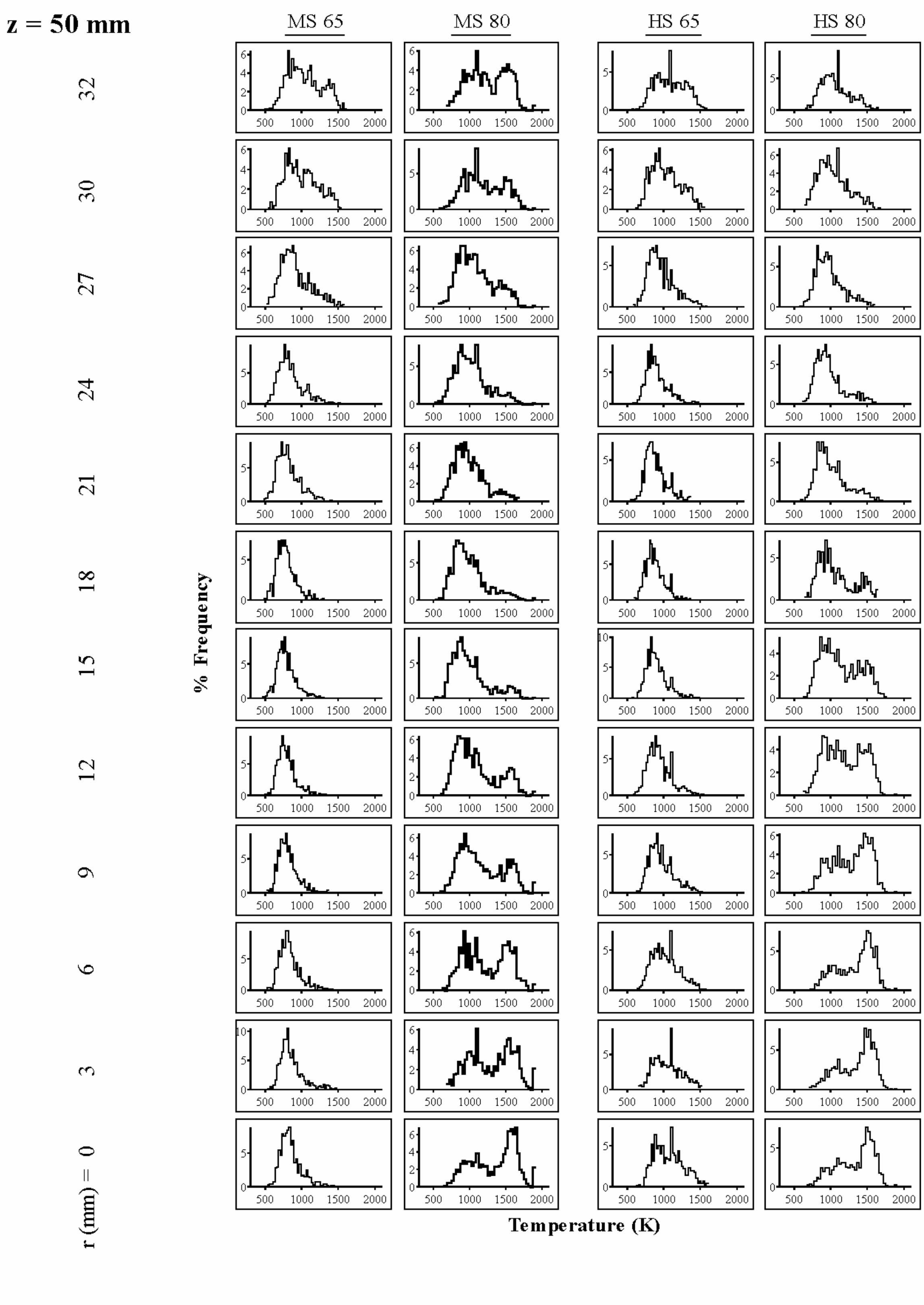

Figure C.14. Gas temperature PDFs as a function of radial position for z = 50 mm.

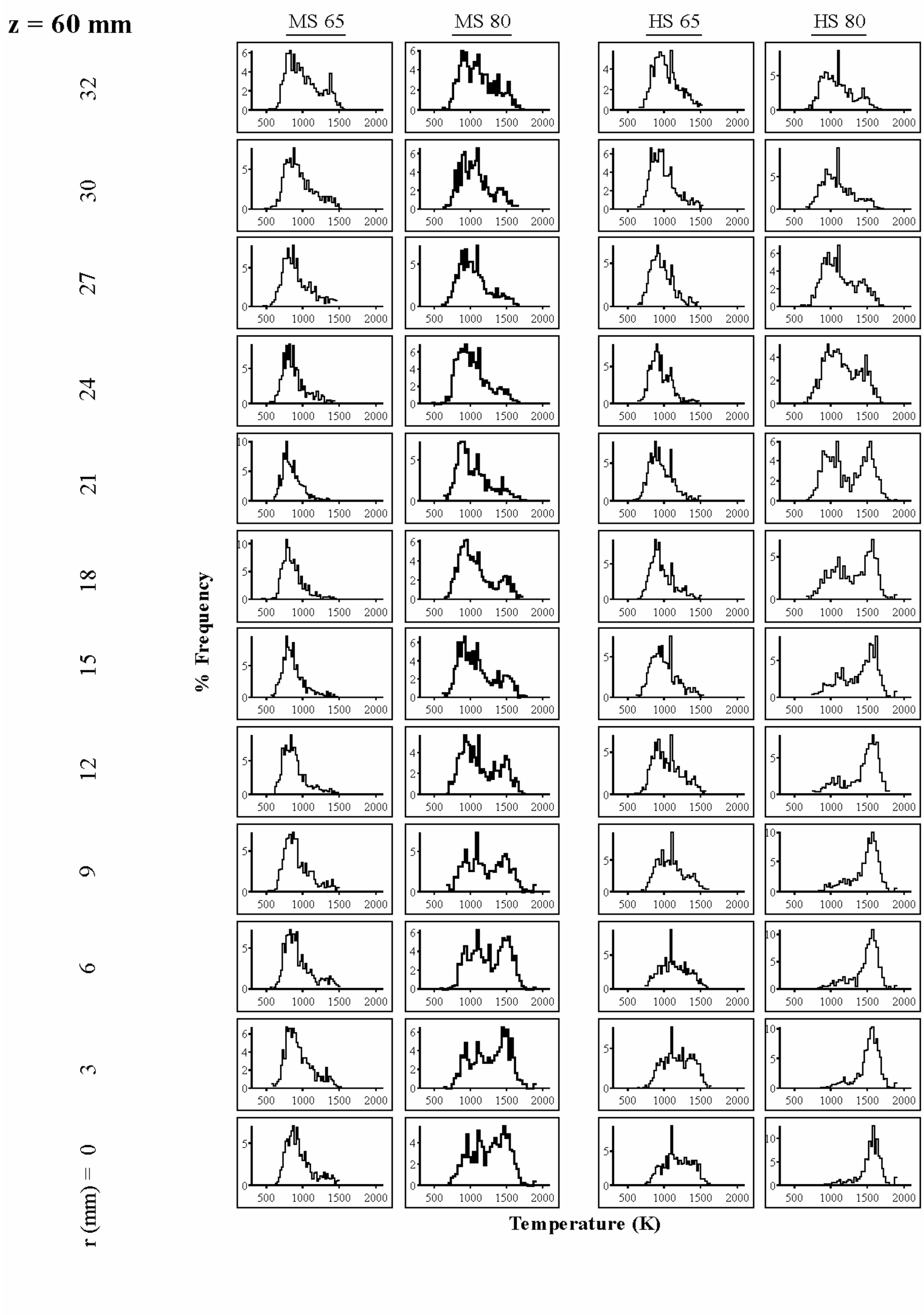

Figure C.15. Gas temperature PDFs as a function of radial position for z = 60 mm.

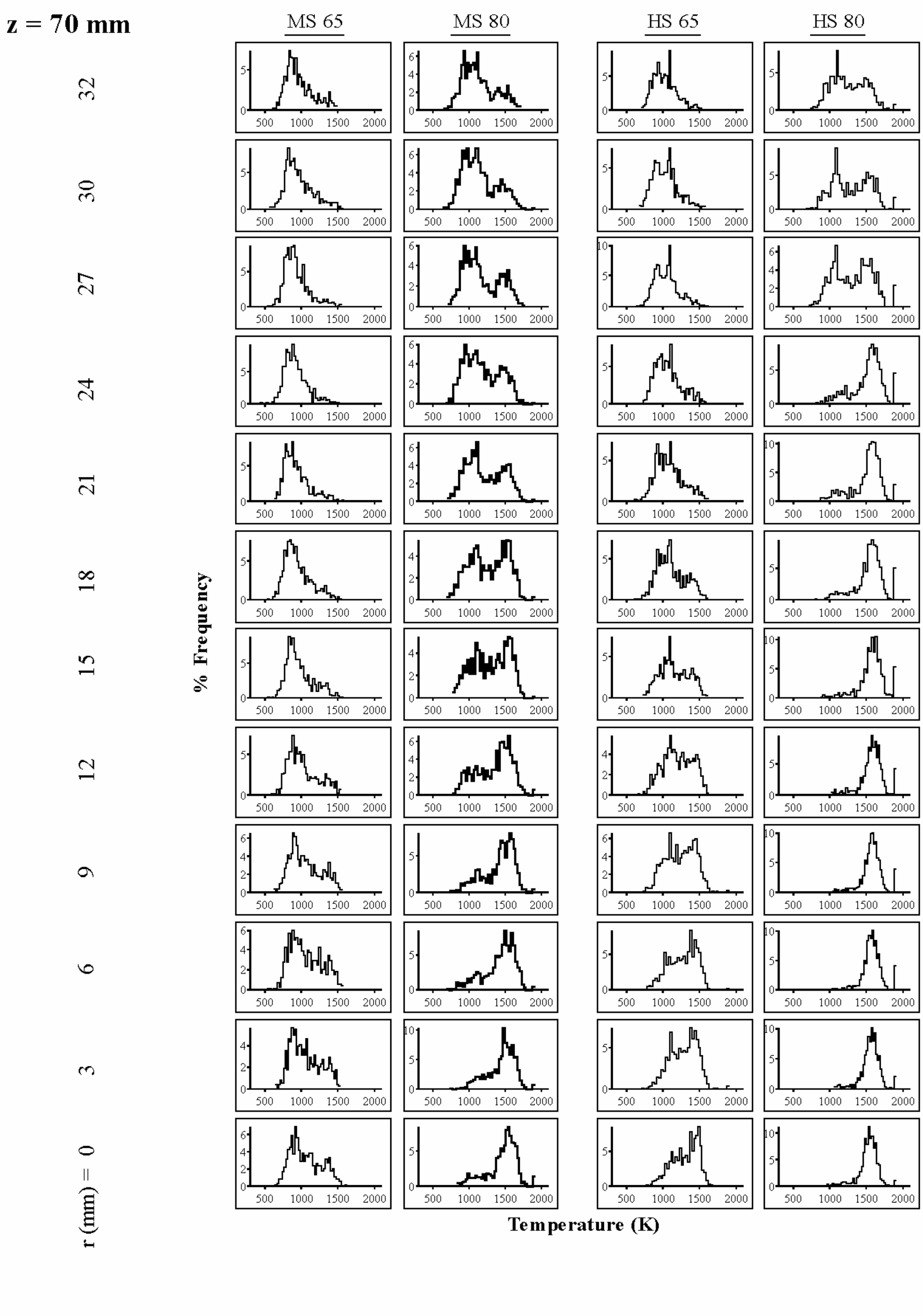

Figure C.16. Gas temperature PDFs as a function of radial position for z = 70 mm.

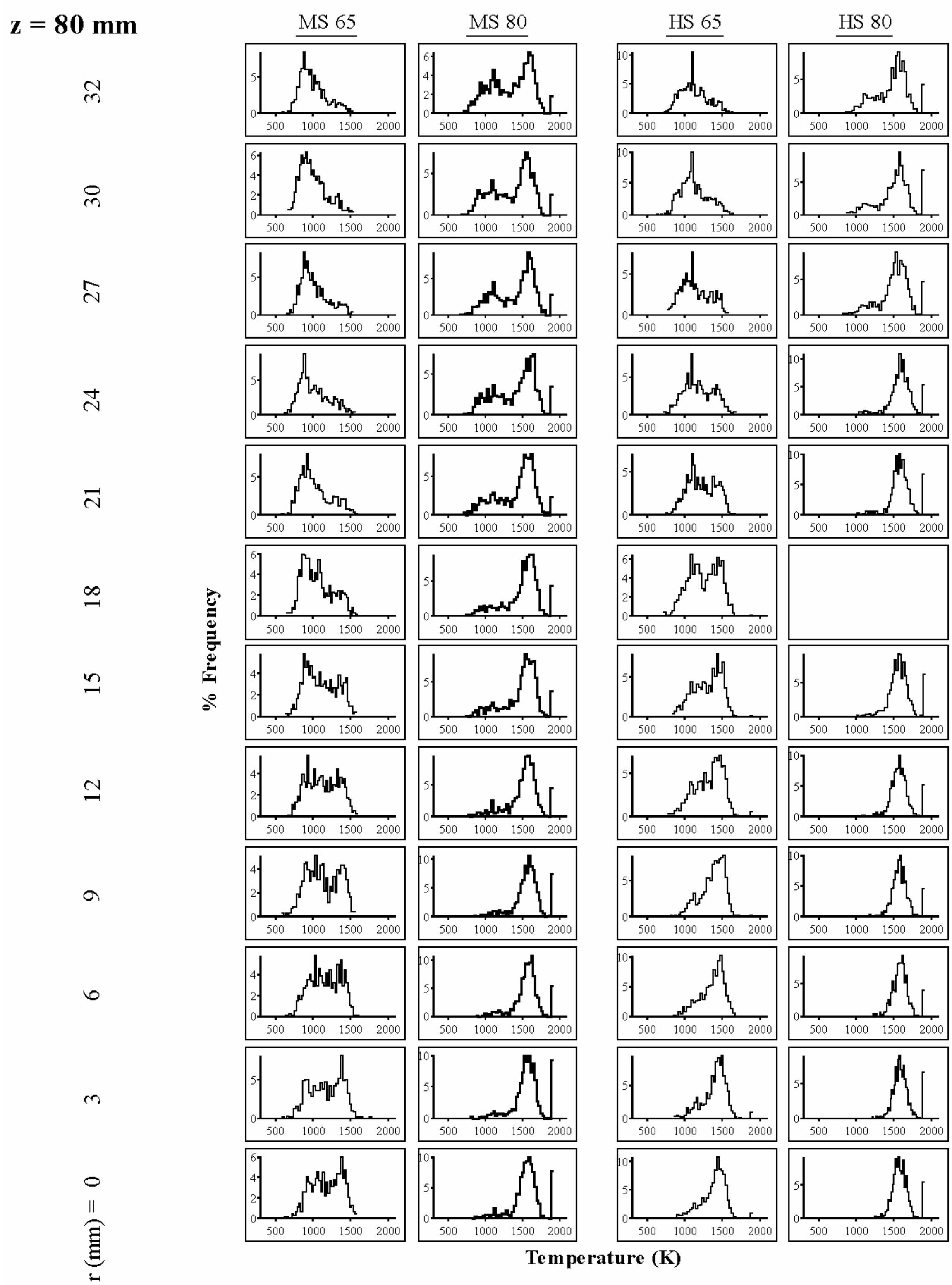

Figure C.17. Gas temperature PDFs as a function of radial position for z = 80 mm.

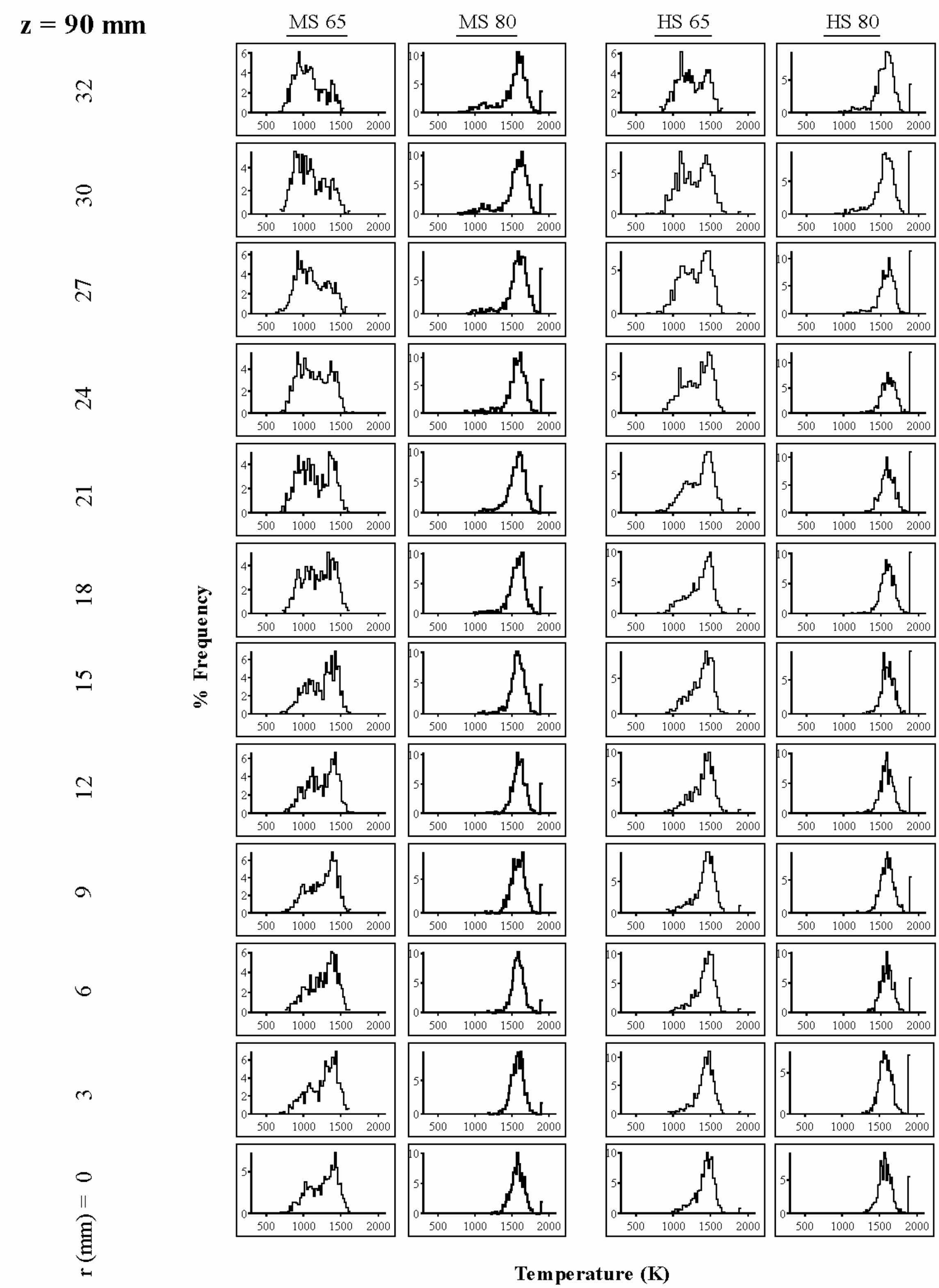

Figure C.18. Gas temperature PDFs as a function of radial position for z = 90 mm.

APPENDIX D

This appendix details the data management system used to handle the large amount of CARS measurements collected in this work. In addition, a description of input files for CARSFT and FMCARS is given. Some sample files are also included.

Table D.1. Data Management System: Experimental Files

	Sample file name	Description
Experimental Data files	C_dyeProfile.dat	Contain the averaged dye laser spectral profile, $I_{dyeProfile}$, for all the samples in a directory.
	C_log.txt	Contains the values of ambient temperature and pressure, which are used to process ambient CARS samples for determining the instrument functions using CARSFT.
	C_xen.dat	Contains the averaged, background-free Xe spectrum over the IPDA.
	IpdaRS	Contains the IPDA Raman Shift, $\Delta w_{R,i}$, computed from Xe spectrum (C_xen.dat).
	hsPhi0.80_z150_r09.dat	Contain single-shot spectra, I_{raw}, recorded by the OMA in ASCII format. The file name indicates the combustion case (hsPhi0.80), the axial position ("_z" followed by three numeric characters), the radial position ("_r" followed by two numeric characters; first numeric value have a "-" sign instead).
	hsPhi0.80_z150_r09.bg	Contain the averaged background noise, I_{bg}, for the corresponding ".dat" file
	Calibration samples to determine instrument function	Average spectra of sample gases used to compute the instrument function. The averaged spectra were obtained by averaging single-shot data and then subtracting the corresponding background noise. After pre-processing (see Chapter 4), the calibration sample CARS spectra were tabulated together with the IPDA Raman Shift into a text file, formatted in the way required by CARSFT (Palmer, 1989), to determine the instrument function .

Table D.2. Data Management System: Computational Files

	Sample file name	Description
Instrument Function	cars.parLibN2, cars.parLibO2, cars.parLibCO2	CARSFT input files to obtain fit libraries for FMCARS (see page D-3). Each have the instrument function Voigt parameters calculated from calibration samples: WLL is the left side value of Δw_l, WVL is the left side value Δw_v, WLR is the right side value of Δw_l, WVR is the right side value Δw_v Fit libraries for different temperatures can be obtained by varying the value of the temperature in these files These files do not contain all the input parameters that CARSFT can take. The default values provided by CARSFT were used for those parameters not included in the '.parLib' files.
	cars.mol, co2.mol	Files containing molecular parameters for calculating CARS spectra for N_2, CO, O_2, and CO_2. These files came with the source code for CARSFT (Palmer, 1989).
	t0850.libN2, x.libN2 t0850.libN2, x.libO2 t0850.libN2, x.libCO2	Fit libraries at 850 K for FMCARS to fit N_2, O_2, and CO_2. The x.libN2 file is the same for all other temperature N_2 libraries. Similarly for O_2, and CO_2. Libraries were produced from 250 to 2050 K in 50 K increments.
FMCARS Input/Output	fmcars.par	Fitting control parameters for FMCARS
	Text file listing experimental "hsPhi0.80_z***_r**.dat" files to be reduced.	FMCARS takes as input a list of .dat files to be reduced. The files must exist in the directory where FMCARS is being executed. This directory contains all data taken during a period where $I_{dyeProfile}$ can be considered constant. FMCARS expects one file name per row in the data file list, including their .dat post-fix. FMCARS ends processing the data file list upon encountering either a blank row or the end-of-file.
	hsPhi0.80_z150_r09.fit	FMCARS output file (see page D-7) containing the temperatures and concentrations for each sample in the single-shot file hsPhi0.80_z150_r09.dat.

File: cars.parLibN2

Probe instrument function file = voigt
Voigt parameters (WLL WVL WLR WVR) = 1.650838 2.758585 1.335586 2.211851
Task number = 1
Beginning wavenumber = 2250.0
Ending wavenumber = 2335.0
Wavenumber range extension = 3.0
Convolution method =double
Adaptive gridding =disable
Probe line width (FWHM) = 1.00000
Pump line width (FWHM) = 0.510000
Temperature = 295.000
N2 mole fraction = 1.000000
Pressure = 1.000000

File: cars.parLibO2

Probe instrument function file = voigt
Voigt parameters (WLL WVL WLR WVR) = 1.545556 2.151904 1.270400 1.550474
Task number = 1
Beginning wavenumber = 1470.0
Ending wavenumber = 1560.0
Wavenumber range extension = 3.0
Convolution method =double
Adaptive gridding =disable
Probe line width (FWHM) = 1.00000
Pump line width (FWHM) = 0.510000
Temperature = 295.000
O2 mole fraction = 1.000000
Pressure = 1.00

File: cars.parLibCO2

Probe instrument function file = voigt
Voigt parameters (WLL WVL WLR WVR) = 1.168307 1.961016 1.477545 6.455628
Task number = 1
Beginning wavenumber = 1385.0
Ending wavenumber = 1455.0
Wavenumber range extension = 3.0
Convolution method =double
Adaptive gridding =disable
Probe line width (FWHM) = 1.00000
Pump line width (FWHM) = 0.510000
Temperature = 295.000
CO2 mole fraction = 1.000000
Pressure = 1.0

File: cars.mol

Molecular parameters used by CARSFT to calculate CARS spectra.

	N2	CO	O2	CO2
WE	0.23585180E+04	0.21698135E+04	0.15801932E+04	0.
WX	0.14293500E+02	0.13288310E+02	0.11980804E+02	0.
WY	-0.59294901E-02	0.10511000E-01	0.47474736E-01	0.
WZ	-0.24000000E-03	0.57400001E-04	0.12127481E-04	0.
BE	0.19982600E+01	0.19312809E+01	0.14456220E+01	0.
ALPHAE	0.17303500E-01	0.17504411E-01	0.15932680E-01	0.
DE	0.57740000E-05	0.61206300E-05	0.	0.
BETAE	0.15500000E-07	-0.11500000E-08	0.	0.
GAME	-0.31536099E-04	0.54870000E-06	0.	0.
DELTE	0.	0.18000000E-09	0.	0.
H0	0.30000000E-11	0.54765001E-11	0.	0.
HE	0.18000000E-11	-0.17300000E-12	0.	0.
RE	0.20743101E+01	0.21322169E+01	0.22890000E+01	0.
GAM	0.47640000E+01	0.47640000E+01	0.47640000E+01	0.
DGAMDR	0.74000000E+01	0.93000000E+01	0.74000000E+01	0.
GNE	0.60000000E+01	0.10000000E+01	0.00000000E+01	0.
GNO	0.30000000E+01	0.10000000E+01	0.10000000E+01	0.
DFSPHI	0.05	0.50000001E-01	0.50000000E-01	0.025
AG	0.11660000E+01	0.15500000E+01	0.17350000E+01	0.
CHINR	0.79000000E+01	0.12300000E+02	0.79800000E+01	0.11800000E+02
AC1	0.24800000E-11	0.24670000E-11	0.25300000E-11	0.
CMASS	0.28013000E+02	0.28010000E+02	0.32000000E+02	0.44000000E+02
ANU4				
ANU5				

File: co2.mol

Additional molecular parameters used by CARSFT to calculate CO_2 CARS spectra.

C STATE 50 (24401) HAS BEEN SYNTHESIZED SINCE IT IS NOT AVAILABLE.
C SAME FOR 38 (15501), 61 (33301), AND 56 (41101)
C SAME FOR 36 (23302), 46 (32201), 48 (32202), AND 59 (41103)
C

LERG	ERG	BROT	BROT2	DROT	DROT2
1	0.	0.39021817	0.	1.33204	0.
2	667.3801	0.39063825	0.39125388	1.35133	1.35900
3	1388.1847	0.39018823	0.	1.14801	0.
4	1285.4087	0.39048155	0.	1.56973	0.
5	1335.1317	0.39166614	0.	1.37350	1.37919
6	2076.8557	0.39040906	0.39133321	1.25790	1.20968
7	1932.4702	0.39074410	0.39168920	1.49289	1.55930
8	2003.2463	0.39237835	0.	1.40490	0.
9	2797.1360	0.39060512	0.	0.97406	0.
10	2671.1433	0.38956092	0.	1.34540	0.
11	2548.3668	0.39110899	0.	1.81380	0.

File: co2.mol (Continued)

12	2760.7249	0.39154669	0.	1.41366	1.27962
13	2585.0223	0.39194216	0.	1.39520	1.52930
14	2671.7170	0.39307640	0.	1.39100	0.
15	3500.6724	0.39038633	0.39171452	1.17358	1.08289
16	3339.3560	0.39003453	0.39117333	1.37126	1.37710
17	3181.4637	0.39102218	0.39235120	1.63220	1.75900
18	3442.2155	0.39221442	0.	1.36330	0.
19	3240.6230	0.39265729	0.	1.49980	0.
20	3340.7190	0.39378400	0.	1.45000	0.
21	4225.0962	0.39098420	0.	0.69490	0.
22	3792.7020	0.39176000	0.	2.02000	0.
23	3942.4920	0.38958400	0.	1.66000	0.
24	4064.2749	0.38959299	0.	0.99260	0.
25	4197.3614	0.39159219	0.	1.58180	1.18060
26	4007.9130	0.39143750	0.	1.46200	1.41500
27	3821.9840	0.39235900	0.	1.46000	1.68000
28	4122.2686	0.39287377	0.	1.39680	0.
29	3898.3270	0.39343400	0.	1.54000	0.
30	4010.0690	0.39447100	0.	1.47000	0.
31	4938.3850	0.39037000	0.39113000	1.07200	0.80800
32	4416.1500	0.39135700	0.39307800	1.63000	1.76000
33	4753.4500	0.38970800	0.39109800	1.20000	1.33000
34	4591.1180	0.38992500	0.39134800	1.53000	1.48000
35	4890.0960	0.39218200	0.	1.61000	0.
36	5677.1710	0.3926	0.	1.3	0.
38	4801.2305	0.39352883	0.39353513	1.40356	1.42527
43	5475.0746	0.38815515	0.38943480	1.61000	1.74750
44	5197.2490	0.39009200	0.	1.89000	0.
46	5644.7072	0.39155805	0.39170548	1.58349	1.08649
48	5436.9679	0.39147525	0.39127505	2.53119	1.45141
49	5245.4690	0.39149800	0.	1.33500	1.53500
50	5579.8835	0.39289131	0.39276783	1.89314	1.47818
56	6388.0754	0.390408	0.	1.3	0.
58	6179.0100	0.38964200	0.	1.33000	0.
59	6002.0337	0.3899	0.	1.3	0.
61	6346.3489	0.39254095	0.39215534	2.83539	1.52982
92	2349.1433	0.38714069	0.	1.32873	0.
93	3004.0122	0.38759172	0.38818943	1.34546	1.35522
94	3714.7828	0.38706251	0.	1.14177	0.
95	3612.8417	0.38750237	0.	1.57314	0.
96	3659.2728	0.38863559	0.	1.36951	1.37351
97	4390.6288	0.38736453	0.38823467	1.24237	1.20084
98	4247.7053	0.38778239	0.38870403	1.49340	1.55942
99	4314.9144	0.38937629	0.	1.39236	0.
100	4853.6234	0.38819689	0.	1.81540	0.
101	5099.6605	0.38749879	0.	0.96329	0.
102	4977.8350	0.38653326	0.	1.36693	0.
103	5061.7792	0.38852321	0.	1.36523	1.26008
104	4887.9858	0.38894166	0.	1.31300	1.43400
105	4970.9312	0.39010706	0.	1.36790	0.

FMCARS Input Parameter File: fmcars.par
Contains Fitting Parameter Values Used
to Fit All Combustion Cases.

# of Vars.	# of Fixed Vars.
6	3

Var	Inp Val	Maximum	Minimum	Max. Delta	Min. Delta
1	0.00000	0.50000	-0.50000	0.020000	0.100000E-01
2	1.00000	1.1000	0.900000	0.010000	0.100000E-03
3	0.00000	1000.00	0.0000000	10.0000	0.100000
4	1.00000	1.10000	0.900000	0.100000E-01	0.100000E-03
5	400.000	2050.00	250.000	100.0000	1.00000
6	13.0000	10000.0	8.00000	1.00000	0.100000E-02

Fixed Variables
----
2 3 4

Variable # legend:
1 -> HORIZONTAL SHIFT
2 -> HORIZONTAL EXPANSION
3 -> VERTICAL SHIFT
4 -> INTENSITY EXPANSION (OPERATES ON DATA)
5 -> TEMPERATURE (K)
6 -> (TOTAL CHINR)/Xi, where Xi is ith species' mole fraction

Table D.3. Sample FMCARS output.

FMCARS output table for the hsPhi0.80_z150_r09.dat file

N2 mole fraction = .7300
Pressure (atm) = .840

# Shot	CHINRT	T(K)	XCO2	XO2	N2PEAK	O2PEAK	CO2PEK	ARERN2	ARERO2	ARRCO2
1	14.2606	1417.0	.2330	.3431	74	10	6	1.018	1.047	1.192
2	22.9667	1508.9	.1936	.2139	159	12	16	.9933	1.024	.9871
3	8.44529	1446.4	.4472E-01	.6159E-01	93	26	26	.9842	1.032	.9777
4	8.80523	1400.1	.3720E-01	.6432E-01	126	21	17	.9843	.9986	.9777
5	7.68879	1636.8	.4700E-01	.6378E-01	128	25	23	1.018	1.008	.9930
6	8.40198	1459.1	.4528E-01	.6441E-01	92	23	20	1.007	1.024	.9794
7	16.2274	1437.2	.2935	.5913	90	10	6	.9942	1.273	1.184
.	.	.	.	.	.	.	.	.	.	.
998	14.6000	1344.8	.1815	.3404	91	6	7	.9837	1.317	1.224
999	5.84000	1628.4	.6612E-01	.7575E-01	128	14	14	.9823	1.004	.9504
1000	5.84000	1552.7	.4218E-01	.6834E-01	48	19	20	1.024	1.058	.9717

*The CO parameters XCO, COPEAK and ARERCO were removed from the actual output in order to fit the relevant information in one page. See accompanying CD for ".fit" files.