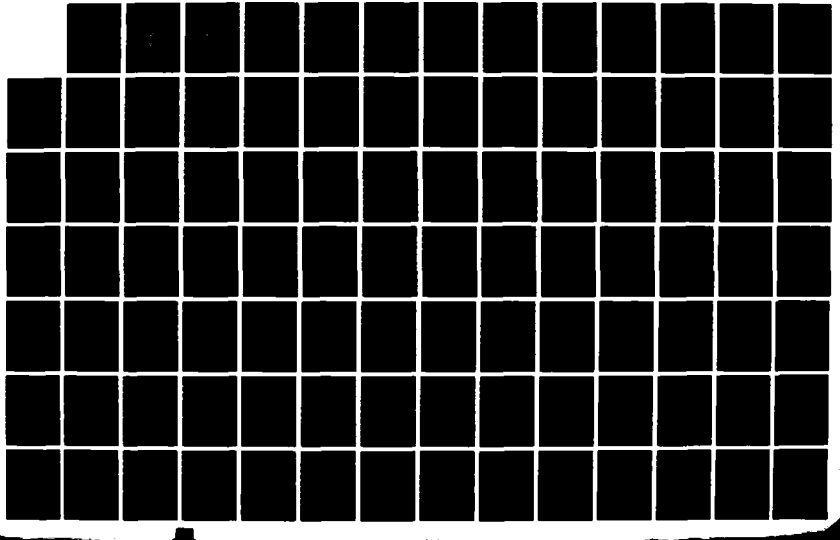
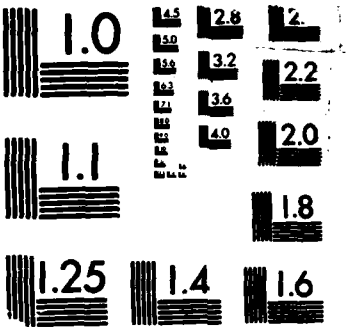
AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS 1/3
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

UNCLASSIFIED F/G 19/11 NN





MICROCOPY RESOLUTION TEST CHART

BUREAU OF STANDARDS-1963-A

(4)

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987

Science Applications International Corporation

(4)

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987

SAIC™

Science Applications International Corporation

Post Office Box 1303, 1710 Goodridge Drive, McLean, Virginia 22102, (703) 821-4300

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

June 1987

Submitted to:

Dr. Jay Boris
Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C. 20375

Prepared by:

Dr. Mark A. Fry

Prepared Under:

Contract No. N00014-86-C-2197

For the Period from March 9, 1986
to March 9, 1987

SCIENCE APPLICATIONS INTERNATIONAL CORPORATION

1710 GOODRIDGE DRIVE, MCLEAN, VIRGINIA 22102
(703) 734-5840

-A-

TABLE OF CONTENTS

NUMERICAL MODELING OF AIRBLAST

CONTRACT No. N00014-86-C-2197

1st Year Final Report

Introduction

The SAIC effort during the period March 10, 1986 to March 10, 1987 is described. Included in this report are appendices for the deliverables such as fortran code listings. The effort was scaled back due to budget considerations. The areas that are contained herein are Tasks 2.1, 2.2, 2.4, 2.5, 2.6, 2.7, 2.8, and 2.9 as described in the statement of work. SAIC over the last year has provided a highly qualified staff of research scientists to perform the tasks described under the contract statement of work. Dr. Mark A. Fry has acted as principal scientist for the contract work. Dr. Shmuel Eidelman has worked on site for 8 months and then 4 months offsite at SAIC. Mr. Ken Laskey assisted as a junior level research scientist for 3 months. Mr. Mike Zuniga joined the effort in November 1986 as a junior level research scientist.

TASK 2.1

The 2-D axisymetric FCT code was applied to nuclear and non-nuclear problems. A burn routine for HE has been added to the code but has not been thoroughly checked. The Minor-scale event was simulated and waveforms at various station locations were obtained. These were used in a comparison with a full 3-D simulation and with the experimental data it was determined that the simulated thermal layer produced multiple peaks in the overpressure waveforms. This was primarily due to a tear in the plastic bag used to contain the helium. The numerical simulations did, however, agree quite well. The results of this work were presented at the Minor Scale Symposium in Albuquerque.

The 2-D code was further optimized for running on the CRAY class computers by employing pointers in the data structure. Speed increases of 10 to 15% were obtained.

Work was also performed in addressing improvements for data management in 2-D codes. This work has been titled "Monotonic Logical Grid Algorithms". Mr. Mike Zuniga has been working full time on these improvements.

A simulation program has been written to test the algorithm. In addition, a graphics program has also been written to aid in this research.

TASK 2.2

A new first principles model for dust ingestion was developed. The momentum integral equation is solved in 1-D along the ground surface. This equation along with boundary layer physical assumptions provides an unsteady prediction of the mass flux emerging from the ground. This model was first proposed by Mirels and is now implemented into the 2-D FCT code. The option DUST=2 initiates the use of this model. A listing of the 2-D code and the subroutine, DUSTN, can be found in Appendix A.

TASK 2.4

The 3-D FCT code was used to study the late time cloud rise geometry from multiple nulear explosions. Calculations involving two bursts separated by distance and time were executed. Cases of 1 MT bursts at 20 and 30 second separation with distance separation of 250 m, 500 m, 1 km, and 3 km were completed. The results indicate a strong sensitivity in cloud formation with a small distance

separation. Previous system modeling codes assumed no sensitivity and have now been corrected.

Spike attacks involving many bursts simultaneously in a 20 x 5 km grid were investigated. Simulations performed with the 3-D code indicated a strong asymmetry out to several minutes in time and up to 40 to 50 km in altitude. A paper summarizing the results was presented at the DNA DUST Symposium in July , 1986. The abstract of the paper is presented in Appendix B. Additional presentations were made at DNA headquarters.

As an additional item, particles were added to the 3-D code for tracking material with time. The 3-D code was used to study the possiblity of using distributed chemical charges in array geometries. Numerous chemical energy release models where employed. The results are summarized in a report in Appendix C.

TASK 2.5

Work on this task was not performed due to budget limitations.

TASK 2.6

Work on this task was not performed due to budget limitations.

TASK 2.7

Work on this task was not performed due to budget limitations.

TASK 2.8

The 3-D graphics program has been updated and improved for hard copy display on Tektronix compatible terminals. Links were written to use DISSPLA as well as the NCAR graphics package. A choice of bold face lettering is now also available. The extensive changes and the improved documentation are available in the source listing, Appendix D .

TASK 2.9

Task 2.9 was not performed due to budget considerations.

APPENDIX A

SOURCE LISTING OF THE 2-D AXISYMMETRIC AND CARTESIAN NUCLEAR AND NON-NUCLEAR MODELLING CODE

```
      program sp2r2d
c-----------------------------------------------------------------------
c     a 2d fct code for high explosive and nuclear blasts in air.
c     coded for cartesian (x-y) and cylindrical (r-z)
c     coordinate systems.
c-----------------------------------------------------------------------
      parameter (mnx=150, mny=200, mxy=202)
      parameter (mnxx=1, mnyy=1)
      parameter (nsta=1)
      parameter (nop=1)
      parameter (mxx=1)
c
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real xbnd(5), dxr(4), ybnd(3), dyr(2)
      common /grdcon/ ndx, nsmth, mdy, alt, naked, xfine, hx, hy,
     1    xbnd, ybnd, dxr, dyr, xleft
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      logical part
      real xp(nop), yp(nop)
      common /part/ part, xp, yp, nopp
c
      real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vls, gms, prs, tme, xs, ys, frac, eblast,
     1    ixx
c
      common /tstep/ dtmin, dtmax
      common /jwl/ r0cj, pcj, dcj, e0cj, gcj, acj, beos, ccj, r1cj,
     1    r2cj, wcj
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1    , jtherm
c
      logical lscan, rigt, left, ltop, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1    jbt, fadd, jadd, factor
c
```

```
      integer itape, otape, ifile, ofile
      logical rezone, lrest, lsave, lgrid, dust, ldump, lmgold
     x      , dustn
      character *8 prntf, restf, dumpf, readf
      character *25 path
      character *40 glab
      character *40 label
      common /inout/ itape, otape, ifile, ofile, lsave, npath, nlab
c
      parameter (ntdump=57)
      real dmptim(ntdump)
      logical ltdump
      data ldump /.true./
c
      data readf /'megdat'/, prntf /'megprt'/, dumpf /'sp2r'/
      data minstp, maxstp /1,11/
      data itape, otape, ifile, ofile /14,15,0,1/
c
      data ndx, nsmth, mdy /75,25,37/
      data hx, hy /1.0,1.0/
      data xfine, alt, xleft /0.0,0.0,0.0/
      data nahed /30/
      data ixx /0/
c
      data dtmin, dtmax, dtstrt /1.0e-6,2.0e-5,1.e-5/
      data jtherm /0/, nbod /0/
c
      namelist /contrl/ minstp,maxstp,idiag,idump,lmgrid,leos,lnuc
     1  ,itape,otape,ifile,ofile,lstat,part,therml,lgrav,lrest,lsave
     2  ,ltdump,dmptim
      namelist /mesh/ nx,ny,alpha,dx,dy,ndx,nsmth,mdy,xfine,alt
     1  ,nahed,hx,hy,iadd,jadd,factor,lscan,rigt,left,ltop,botm,lgrid
     2  ,xleft
      namelist /bcond/ bcr,bcl,bct,bcb
      namelist /phydat/ gamma,rhomin,dtmin,dtmax,nspec,jtherm,jdirt
     1  ,dtstrt,dust,nbod,dustn
      namelist /jwlist/ r0cj,pcj,dcj,e0cj,gcj,acj,beos,ccj,r1cj,r2cj
     1  ,wcj
      namelist /stats/ xs,ys
      namelist /newgrd/ ixr,ixl,jyt,jyb
c
c     initialize the code control parameters.
      idiag=10000
      idump=10000
      lmgrid=.false.
      lstat=.false.
      part=.false.
      leos=.false.
      lnuc=.false.
      lgrav=.false.
      therml=.false.
      rezone=.false.
      lscan=.false.
      rigt=.false.
      left=.false.
      ltop=.false.
      botm=.false.
      lsave=.false.
      lgrid=.false.
      dust=.false.
      dustn=.false.
      ltdump=.false.
c
c     initialize grid parameters.
```

```
129             nx=mnx
130             ny=mny
131             nxp=nx+1
132             nyp=ny+1
133             alpha=1
134             dx=1.0
135             dy=1.0
136             ixr=mnx
137             jxr=1
138             ixl=1
139             jxl=1
140             iyt=1
141             jyt=mny
142             iyb=1
143             jyb=1
144             ilr=mnx
145             jbt=mny
146             iadd=10
147             jadd=10
148 c
149 c       initialize boundary conditions.
150             bcr=0.0
151             bcl=0.0
152             bct=0.0
153             bcb=-1.0
154 c
155 c       physical data.
156             rhomin=1.29e-06
157             gamma=1.4
158             gmi=gamma-1.0
159             nspec=1
160 c
161             time=0.0
162             tme(1)=time
163             radl=0.0
164 c
165 c       open read and print files. read and write data.
166             call bufman
167             open (unit=5,file=readf,status='old')
168             read (5,180) label
169             read (5,200) prntf
170             read (5,200) dumpf
171             read (5,200) restf
172             read (5,180) path
173             nlab=0
174             do 10 i=1,40
175             nlab=nlab+1
176             if (label(i:i).eq.'$') go to 20
177     10      continue
178     20      npath=0
179             do 30 i=1,25
180             if (path(i:i).eq.' ') go to 40
181     30      npath=npath+1
182     40      continue
183 c
184             read (5,contrl)
185             if (.not.lrest) then
186             do 50 i=8,1,-1
187             prntf(i:i)=dumpf(i:i)
188             if (prntf(i:i).ne.' '.and.prntf(i+1:i+1).eq.' ') prntf(i+1:i+1
189           1 )='0'
190     50      continue
191             ifile=0
192             ofile=0
```

```
193           endif
194           open (unit=6,file=prntf,status='new')
195           write (6,190) label
196           write (6,contrl)
197           read (5,mesh)
198           write (6,mesh)
199           read (5,bcond)
200           write (6,bcond)
201           read (5,phydat)
202           write (6,phydat)
203           dt=dtstrt
204           if (nspec.gt.1) read (5,jwlist)
205           if (nspec.gt.1) write (6,jwlist)
206           if (.not.therml) jtherm=0
207           if (.not.lstat) go to 70
208           read (5,stats)
209           write (6,stats)
210           do 60 ksta=1,nsta
211    60     ys(ksta)=ys(ksta)+alt
212           if (idump.le.mxx) go to 70
213           write (6,220) idump,mxx
214           go to 160
215    70     continue
216           do 80 k=1,mxy
217    80     unit(k)=1.0
218 c
219 c     determine bluff body profiles.
220           if (nbod.ne.0) call bluf (nbod)
221 c
222 c     initialize variables to ambient conditions.
223           if (.not.lrest) then
224           call inital (bcl,bcr,bcb,bct)
225           if (dust) call dust0
226           if(dustn) call dust01
227           if (therml) call tlayer (rho,rvx,rvy,erg,rhomin,ensum)
228 c          if (lnuc) call dposit (time)
229           if (lnuc) then
230           call burst (xcor,ycor,erg)
231 c          call dposit (time)
232           else
233           call jump (gamma,leos,er00(2))
234           call dpshok (rho,rvx,rvy,erg)
235           endif
236           call dumper (0,time,dumpf,label,path,glab)
237           call diagno (istep,time)
238           go to 160
239           endif
240 c
241 c     restart if lrest=.true.
242           lmgold=lmgrid
243           call restrt (minstp,time,restf,glab)
244           lmgrid=lmgold
245           if (.not.lnuc) call jump (gamma,leos,er00(2))
246 c
247 c     determine dump time number.
248           itdump=1
249           if (ltdump) then
250           do 90 ktdump=ntdump,1,-1
251           if (time.ge.dmptim(ktdump)) go to 100
252    90     continue
253           go to 110
254   100     itdump=ktdump+1
255           itdump=itdump.le.ntdump
256   110     continue
```

```
257           endif
258 c
259 c     read in boundaries of active grid.
260           if (lgrid) then
261           read (5,newgrd)
262           write (6,newgrd)
263           ilr=ixr-ixl+1
264           jbt=jyt-jyb+1
265           endif
266 c
267 c     initialize various routines.
268         ratio=rho(mnx,7)/rho(mnx,1)
269         write(6,987) ratio
270  987    format(1x,'ratio=',1pe12.5,' rad loss fbloss')
271           if (lnuc) call fbloss (time,dt,radl,eblast,ratio)
272           if (lscan) call xyg
273           iactiv=min0(iadd,jadd)
274 c
275 c     begin the loop over time steps.
276           do 150 istep=minstp,maxstp
277           kstep=istep-minstp
278 c
279 c     determine active grid boundaries.
280           if (lscan) then
281           if (mod(kstep,iactiv).eq.0) call xygrid (erg,scrh)
282           endif
283 c
284 c     the time and the timestep are updated.
285           call dtset (istep,time,radl,ensum)
286           if (nbod.ne.0) go to 130
287 c
288 c     update grid.
289           if (.not.lmgrid) go to 120
290           call xgrid (jtherm)
291           call ygrid
292   120     continue
293 c
294 c     integrate along rows and columns.
295           call bndrx (bcr,bcl)
296           call integx
297           call bndry (bct,bcb)
298           call integy
299           go to 140
300 c
301 c     integration for shock on bluff bodies.
302   130     call blufxy (bcl,bcr,bcb,bct)
303   140     continue
304 c
305 c     perform radiation loss calculation.
306           if (lnuc) call radlos (time,xcor,ycor,radl,fracn)
307 c
308 c     calculate station data.
309           if (lstat) call staton (ixx,istep,time)
310 c
311 c     push particles
312           if (part) call partic (xcor,ycor)
313 c
314 c     inject dust.
315           if (dust) call duster(istep,time)
316           if(dustn) call dustm(istep,time)
317 c
318 c     dump at the appropriate step.
319           if (ltdump) then
320           ldump=(istep.eq.maxstp).or.(time.ge.dmptim(ltdump))
```

```
321           else
322           ldump=(istep.eq.maxstp).or.(idump.ne.0.and.mod(istep,idump).eq
323          1     .0)
324           endif
325           ldump=cvmgt(.false.,ldump,istep.eq.minstp)
326           if (ldump) then
327           call flush
328           call dumper (istep,time,dumpf,label,path,glab)
329           call flush
330           ixx=0
331           if (itdump) then
332           if (time.ge.dmptim(ntdump)) go to 160
333           itdump=itdump+1
334           itdump=itdump.le.ntdump
335           endif
336           endif
337 c
338 c     perform all required diagnostics
339           if (idiag.ne.0.and.mod(istep,idiag).eq.0) call diagno (istep
340          1     ,time)
341           if(time.gt.25.) go to 170
342   150     continue
343 c
344 c     save print file.
345   160     write (6,210) ptrem
346           if (.not.lsave) go to 170
347           write (glab,230) path(1:npath),prntf
348           ierr=1000
349           call mass ('save','\wait=on\',0,glab,ierr)
350           write (6,240) prntf,ierr
351 c
352 c     close files.
353   170     close (unit=5,status='keep')
354           close (unit=6,status='keep')
355           stop
356 c
357   180 format (a)
358   190 format ('1fast2d hydrodynamics  ',5x,a)
359   200 format (15x,a)
360   210 format ('0ptrem = ',e13.5)
361   220 format (1x,'idump,mxx=',2i5/1x,'idump should be .le. mxx')
362   230 format ('&',a,a,'&')
363   240 format (1x,'ierr for ',a,' =',i4/)
364       end
365           subroutine xygrid (erg,scr1)
366 c-----------------------------------------------------------------
367 c     determines extent of active grid.
368 c-----------------------------------------------------------------
369           parameter (mnx=150, mny=200, mxy=202)
370           real erg(mnx,mny), scr1(mxy), scr2(mxy)
371 c
372           real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
373           parameter (jthm=1)
374           real fsky(mxy), rhtl(mnx,jthm)
375           common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
376          1     , jtherm
377 c
378           logical lscan, rigt, ltop, left, botm
379           common /scan/ lscan, rigt, left, ltop, botm
380           common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
381          1     jbt, iadd, jadd, factor
382 c
383 c     save old grid boundaries.
384           ixro=ixr
```

```
385             ixlo=ixl
386             jyto=jyt
387             jybo=jyb
388 c
389 c       right.
390             if (.not.rigt) go to 30
391             ixr=1
392             do 20 j=jmin,jmax
393             do 10 i=imin,imax
394      10     scr1(i)=cvmgt(float(i),0.0,erg(i,j).ge.scr2(j))
395             klim=ismax((imax-imin+1),scr1(imin),1)+imin-1
396             ixr=max0(klim,ixr)
397             jxr=cvmgt(j,jxr,ixr.eq.klim)
398      20     continue
399             ixr=min0((ixr+iadd),mnx)
400             imax=ixr
401             rigt=cvmgt(.false.,.true.,ixr.eq.mnx)
402 c
403 c       top.
404      30     if (.not.itop) go to 60
405             jyt=1
406             do 50 i=imin,imax
407             do 40 j=jmin,jmax
408      40     scr1(j)=cvmgt(float(j),0.0,erg(i,j).ge.scr2(j))
409             klim=ismax((jmax-jmin+1),scr1(jmin),1)+jmin-1
410             jyt=max0(klim,jyt)
411             iyt=cvmgt(i,iyt,jyt.eq.klim)
412      50     continue
413             jyt=min0((jyt+jadd),mny)
414             jmax=jyt
415             itop=cvmgt(.false.,.true.,jyt.eq.mny)
416 c
417 c       left.
418      60     if (.not.left) go to 90
419             ixl=mnx
420             do 80 j=jmin,jmax
421             do 70 i=imin,imax
422      70     scr1(j)=cvmgt(float(i),float(mnx),erg(i,j).ge.scr2(j))
423             klim=ismin((imax-imin+1),scr1(imin),1)+imin-1
424             ixl=cvmgt(min0(klim,ixl),ixl,klim.ne.imin)
425             jxl=cvmgt(j,jxl,ixl.eq.klim)
426      80     continue
427             ixl=max0((ixl-iadd),1)
428             imin=ixl
429             left=cvmgt(.false.,.true.,ixl.eq.1)
430 c
431 c       bottom.
432      90     if (.not.botm) go to 120
433             jyb=mny
434             do 110 i=imin,imax
435             do 100 j=jmin,jmax
436     100     scr1(j)=cvmgt(float(j),float(mny),erg(i,j).ge.scr2(j))
437             klim=ismin((jmax-jmin+1),scr1(jmin),1)+jmin-1
438             jyb=cvmgt(min0(klim,jyb),jyb,klim.ne.jmin)
439             iyb=cvmgt(i,iyb,jyb.eq.klim)
440     110     continue
441             jyb=max0((jyb-jadd),1)
442             jmin=max0(jyb,jtherm+1)
443             botm=cvmgt(.false.,.true.,jyb.eq.1)
444     120     ixr=cvmgt(max0(ixr,ixro),ixr,ixro.ne.mnx)
445             ixl=cvmgt(min0(ixl,ixlo),ixl,ixlo.ne.1)
446             jyt=cvmgt(max0(jyt,jyto),jyt,jyto.ne.mny)
447             jyb=cvmgt(min0(jyb,jybo),jyb,jybo.ne.1)
448             ilr=ixr-ixl+1
```

```
449           jbt=jyt-jyb+1
450           lscan=cvmgt(.false.,.true.,ixl.eq.1.and.ixr.eq.mnx.and.jyb.eq.
451          1    1.and.jyt.eq.mny)
452           return
453 c
454           entry xyg
455           imin=ixl
456           imax=ixr
457           jmin=max0(jyb,jtherm+1)
458           jmax=jyt
459           do 130 j=1,mny
460       130 scr2(j)=factor*er00(j+1)
461           return
462         end
463           subroutine integx
464 c-------------------------------------------------------------------
465 c     integrates along rows.
466 c-------------------------------------------------------------------
467           parameter (mnx=150, mny=200, mxy=202)
468           parameter (mnxx=1, mnyy=1)
469           integer alpha
470           logical lmgrid, lstat, lnuc, therml, lgrav, leos
471           real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
472           real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
473           real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
474           common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
475           common lstat, lnuc, therml, lgrav, leos
476           common alpha, gamma, gm1, dt, dx, dy, rhomin
477           common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
478 c
479           real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
480           common /grids/ xcor, xcoro, ycor, ycoro, unit
481 c
482           real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
483           real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
484           real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
485           real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
486           real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
487           real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
488           real rmin(mxy), rmax(mxy), rvr(mxy)
489           common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
490          1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
491          2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
492 c
493           real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
494           equivalence (rscr,srvx), (pscr,srvy), (gscr,serg)
495           equivalence (scr1,scrh), (scr2,rmin)
496           real gams(mxy), rhah(mxy), rhbh(mxy), rhch(mxy), rgrv(mxy)
497           equivalence (gams,sgam), (rgrv,dely)
498           equivalence (rhah,rmax), (rhbh,rvr), (rhch,delx)
499 c
500           real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
501           parameter (jthm=1)
502           real fsky(mxy), rhtl(mnx,jthm)
503           common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
504          1    , jtherm
505 c
506           common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
507          1    jbt, iadd, jadd, factor
508 c
509           common /lbc/ rholbc, rvxlbc, rvylbc, erglbc
510           common /rbc/ rhorbc, rvxrbc, rvyrbc, ergrbc
511           common /bbc/ rhobbc, rvxbbc, rvybbc, ergbbc
512           common /tbc/ rhotbc, rvxtbc, rvytbc, ergtbc
```

```
c
          call ngridd (xcor0(ixl),ilr+1,alpha)
          call ogridd (ilr+1)
          call ngridd (xcor(ixl),ilr+1,alpha)
c
c     loop over rows to be integrated.
          do 250 j=jyb,jyt
          rhor=cvmgt(rho0(j+1),rhor,abcr.eq.1.0)
          rhol=cvmgt(rho0(j+1),rhol,abcl.eq.1.0)
          ergr=cvmgt(er00(j+1),ergr,abcr.eq.1.0)
          ergl=cvmgt(er00(j+1),ergl,abcl.eq.1.0)
          rhcr=rhor
          rhcl=rhol
          do 10 i=ixl,ixr
          rho0(i)=rho(i,j)
          rhc0(i)=rho(i,j)
          rha0(i)=0.0
          rhb0(i)=0.0
          rvx0(i)=rvx(i,j)
          rvy0(i)=rvy(i,j)
          erg0(i)=erg(i,j)
          rhoh(i)=rho0(i)
          rvxh(i)=rvx0(i)
          rvyh(i)=rvy0(i)
   10     ergh(i)=erg0(i)
c
          go to (60,40,20), nspec
   20     do 30 i=ixl,ixr
          rhb0(i)=rhb(i,j)
          rhbh(i)=rhb0(i)
   30     rhc0(i)=rhc0(i)-rhb0(i)
          rhcr=rhcr-rhbr
          rhcl=rhcl-rhbl
   40     do 50 i=ixl,ixr
          rha0(i)=rha(i,j)
          rhah(i)=rha0(i)
          rhc0(i)=amax1((rhc0(i)-rha0(i)),0.0)
   50     rhch(i)=rhc0(i)
          rhcr=rhcr-rhar
          rhcl=rhcl-rhal
   60     continue
c
c     define boundary velocities by averaging values in interior and
c     exterior boundary cells.
          rvxlef=frvxl*rvx(ixl,j)+rvxl
          rholef=frhol*rho(ixl,j)+rhol
          uh(ixl)=rvx(ixl,j)/rho(ixl,j)
          uh(ixl)=0.5*(uh(ixl)+rvxlef/rholef)
c
          rvxrig=frvxr*rvx(ixr,j)+rvxr
          rhorig=frhor*rho(ixr,j)+rhor
          uh(ixr+1)=rvx(ixr,j)/rho(ixr,j)
          uh(ixr+1)=0.5*(uh(ixr+1)+rvxrig/rhorig)
c
c     advance total density, energy and x and y momentum for
c     half and then whole step.
          do 190 ipass=1,2
          dtsub=dt/float(3-ipass)
c
c     calculate transport velocities.
          do 70 i=ixl,ixr
          rhom(i)=1.0/rhoh(i)
          vel(i)=rvxh(i)*rhom(i)
   70     ergk(i)=amax1(0.0,ergh(i)-0.5*rhom(i)*(rvxh(i)*rvxh(i)+rvyh(i)
```

```
577       1     *rvyh(i))-rhoh(i)*gp00(j+1))
578             do 80 i=ixl+1,ixr
579          80 uh(i)=0.5*(vel(i)+vel(i-1))
580             call velocd (uh(ixl),(ilr+1),dtsub,ilr)
581 c
582 c        calculate pressure.
583             if (leos) then
584 c            call eos (ilr,nspec,rhoh(ixl),ergk(ixl),gams(ixl),pres(ixl)
585 c     1       ,rscr(ixl),pscr(ixl),gscr(ixl),scr1(ixl),scr2(ixl),rhah(ixl)
586 c     2       ,rhbh(ixl))
587             call eosp1 (rhoh(ixl),ergk(ixl),ilr,gams(ixl),pres(ixl))
588             else
589             do 90 i=ixl,ixr
590          90 pres(i)=gm1*ergk(i)
591             endif
592 c
593 c        calculate source terms.
594             do 100 i=ixl,ixr
595             srvx(i)=-pres(i)
596         100 serg(i)=srvx(i)*vel(i)
597             srvxl=-pres(ixl)
598             srvxr=-pres(ixr)
599             sergl=-pres(ixl)*uh(ixl)
600             sergr=-pres(ixr)*uh(ixr+1)
601 c
602 c        advance total density, energy and x and y momentum.
603             if (abs(abc).eq.1.0) call setbd (1,1)
604             call fryfct (rho0(ixl),rhoh(ixl),ilr,frhor,rhor,frhol,rhol)
605             do 110 i=ixl,ixr
606         110 rhoh(i)=amax1(rhoh(i),rhomin)
607             call sourcd (ilr,dtsub,2,unit(ixl),srvx(ixl),srvxr,srvxl)
608             call fryfct (rvx0(ixl),rvxh(ixl),ilr,frvxr,rvxr,frvxl,rvxl)
609             call fryfct (rvy0(ixl),rvyh(ixl),ilr,frvyr,rvyr,frvyl,rvyl)
610             call sourcd (ilr,dtsub,1,unit(ixl),serg(ixl),sergr,sergl)
611             call fryfct (erg0(ixl),ergh(ixl),ilr,fergr,ergr,fergl,ergl)
612 c
613 c        advance species.
614             do 120 i=ixl,ixr
615             scr1(i)=1.e-8*rhoh(i)
616         120 scr2(i)=0.0
617             go to (190,150,130), nspec
618         130 call fryfct (rhb0(ixl),rhbh(ixl),ilr,frhbr,rhbr,frhbl,rhbl)
619             do 140 i=ixl,ixr
620             rhbh(i)=cvmgt(0.0,rhbh(i),rhbh(i).le.scr1(i))
621         140 scr2(i)=scr2(i)+rhbh(i)
622         150 call fryfct (rha0(ixl),rhah(ixl),ilr,frhar,rhar,frhal,rhal)
623             do 160 i=ixl,ixr
624             rhah(i)=cvmgt(0.0,rhah(i),rhah(i).le.scr1(i))
625         160 scr2(i)=scr2(i)+rhah(i)
626             call fryfct (rhc0(ixl),rhch(ixl),ilr,frhcr,rhcr,frhcl,rhcl)
627             do 170 i=ixl,ixr
628             rhch(i)=cvmgt(0.0,rhch(i),rhch(i).le.scr1(i))
629         170 scr2(i)=scr2(i)+rhch(i)
630             do 180 i=ixl,ixr
631         180 scrh(i)=rhoh(i)/scr2(i)
632         190 continue
633 c
634             do 200 i=ixl,ixr
635             rho(i,j)=rhoh(i)
636             rvx(i,j)=rvxh(i)
637             rvy(i,j)=rvyh(i)
638         200 erg(i,j)=ergh(i)
639             go to (250,230,210), nspec
640         210 do 220 i=ixl,ixr
```

```
641   220     rhb(i,j)=rhbh(i)*scrh(i)
642   230     do 240 i=ix1,ixr
643   240     rha(i,j)=rhah(i)*scrh(i)
644 c
645 c     loop back for the next row to be integrated.
646   250     continue
647           return
648 c
649           entry integy
650 c ---------------------------------------------------------- --------
651 c     integrates along columns.
652 c---- ------------------------------------------------------ ----------
653           call ngridd (ycoro(jyb),jbt+1,1)
654           call ogridd (jbt+1)
655           call ngridd (ycor(jyb),jbt+1,1)
656 c
657 c     loop over the cols to be integrated.
658           do 520 i=ix1,ixr
659           do 260 j=jyb,jyt
660           rho0(j)=rho(i,j)
661           rhc0(j)=rho(i,j)
662           rha0(j)=0.0
663           rhb0(j)=0.0
664           rvx0(j)=rvx(i,j)
665           rvy0(j)=rvy(i,j)
666           erg0(j)=erg(i,j)
667           rhoh(j)=rho0(j)
668           rvxh(j)=rvx0(j)
669           rvyh(j)=rvy0(j)
670   260     ergh(j)=erg0(j)
671 c
672           go to (310,290,270), nspec
673   270     do 280 j=jyb,jyt
674           rhb0(j)=rhb(i,j)
675           rhbh(j)=rhb0(j)
676   280     rhc0(j)=rhc0(j)-rhb0(j)
677           rhct=rhct-rhbt
678           rhcb=rhcb-rhbb
679   290     do 300 j=jyb,jyt
680           rha0(j)=rha(i,j)
681           rhah(j)=rha0(j)
682           rhc0(j)=amax1((rhc0(j)-rha0(j)),0.0)
683   300     rhch(j)=rhc0(j)
684           rhct=rhct-rhat
685           rhcb=rhcb-rhab
686   310     continue
687 c
688 c     set thermal layer densities.
689           do 312 j=1,mxy
690   312     rgrv(j)=rh00(j)
691           if (therml) then
692           do 314 j=1,jthm
693   314     rgrv(j+1)=rhtl(i,j)
694           endif
695 c
696 c     define boundary velocities by averaging values in interior and
697 c     exterior boundary cells.
698           rvybot=frvyb*rvy(i,jyb)+rvyb
699           rhobot=frhob*rho(i,jyb)+rhob
700           vh(jyb)=rvy(i,jyb)/rho(i,jyb)
701           vh(jyb)=0.5*(vh(j,b)+rvybot/rhobot)
702 c
703           rvytop=frvyt*rvy(i,jyt)+rvyt
704           rhotop=frhot*rho(i,jyt)+rhot
```

```
705           vh(jyt+1)=rvy(1,jyt)/rho(1,jyt)
706           vh(jyt+1)=0.5*(vh(jyt+1)+rvytop/rhotop)
707 c
708 c     advance total density, energy and x and y momentum
709 c     for half and then whole step.
710           do 460 ipass=1,2
711           dtsub=dt/float(3-ipass)
712 c
713 c     calculate transport velocities.
714           do 320 j=jyb,jyt
715           rhom(j)=1.0/rhoh(j)
716           vel(j)=rvyh(j)*rhom(j)
717    320    ergk(j)=amax1(0.0,ergh(j)-0.5*rhom(j)*(rvxh(j)*rvxh(j)+rvyh(j)
718       1   *rvyh(j))-rhoh(j)*gp00(j+1))
719           do 330 j=jyb+1,jyt
720    330    vh(j)=0.5*(vel(j)+vel(j-1))
721           call velocd (vh(jyb),(jbt+1),dtsub,jbt)
722 c
723 c     calculate pressure.
724           if (leos) then
725 c          call eos (jbt,nspec,rhoh(jyb),ergk(jyb),gams(jyb),pres(jyb)
726 c     1    ,rscr(jyb),pscr(jyb),gscr(jyb),scr1(jyb),scr2(jyb),rhah(jyb)
727 c     2    ,rhbh(jyb))
728           call eospi (rhoh(jyb),ergk(jyb),jbt,gams(jyb),pres(jyb))
729           else
730           do 340 j=jyb,jyt
731    340    pres(j)=gm1*ergk(j)
732           endif
733 c
734 c     calculate source terms.
735           if (lgrav) then
736           do 350 j=jyb,jyt
737 c          srvy(j)=pr00(j+1)-pres(j)
738 c  350     sgrv(j)=-(rgrv(j+1)-rhoh(j))*gravy(j+1)
739 c          srvyb=pr00(jyb)-pres(jyb)
740 c          srvyt=pr00(jyt+1)-pres(jyt)
741 c          sgrv(jyt+1)=-(rgrv(jyt+2)-rhoh(jyt))*gravy(jyt+2)
742           srvy(j)=pr00(j+1)-pres(j)-fsky(j)
743    350    sgrv(j)=rhoh(j)*gravy(j+1)
744           srvyb=pr00(jyb)-pres(jyb)-fsky(jyb)
745           srvyt=pr00(jyt+1)-pres(jyt)-fsky(jyt+1)
746           sgrv(jyt+1)=rhoh(jyt)*gravy(jyt+2)
747           else
748           do 355 j=jyb,jyt
749    355    srvy(j)=-pres(j)
750           srvyb=-pres(jyb)
751           srvyt=-pres(jyt)
752           endif
753           do 370 j=jyb,jyt
754 c  370     serg(j)=-pres(j)*vel(j)
755 c          sergt=-pres(jyt)*vh(jyt+1)
756 c          sergb=-pres(jyb)*vh(jyb)
757    370    serg(j)=-(pres(j)+fsky(j+1)-pr00(j+1))*vel(j)
758           sergt=-(pres(jyt)+fsky(jyt+2)-pr00(jyt+2))*vel(jyt+1)
759           sergb=-(pres(jyb)+fsky(jyb)-pr00(jyb))*vh(jyb)
760 c
761 c     advance density, energy and x and y momentum.
762           if (abs(abcb).eq.1.0) call setbd (1,1)
763           call fryfct (rho0(jyb),rhoh(jyb),jbt,frhot,rhot,frhob,rhob)
764           do 380 j=jyb,jyt
765    380    rhoh(j)=amax1(rhoh(j),rhomin)
766           call sourcd (jbt,dtsub,2,unit(jyb),srvy(jyb),srvyt,srvyb)
767           if (lgrav) then
768           call sourcd (jbt,dtsub,3,unit(jyb),sgrv(jyb),sgrv(jyt+1),0.0)
```

```
769             endif
770             call fryfct (rvy0(jyb),rvyh(jyb),jbt,frvyt,rvyt,frvyb,rvyb)
771             call fryfct (rvx0(jyb),rvxh(jyb),jbt,frvxt,rvxt,frvxb,rvxb)
772             call sourcd (jbt,dtsub,1,unit(jyb),sergt(jyb),sergt,sergb)
773             call fryfct (erg0(jyb),ergh(jyb),jbt,fergt,ergt,fergb,ergb)
774 c
775 c       advance species.
776             do 390 j=jyb,jyt
777             scr1(j)=1.e-8*rhoh(j)
778     390     scr2(j)=0.0
779             go to (460,420,400), nspec
780     400     call fryfct (rhb0(jyb),rhbh(jyb),jbt,frhbt,rhbt,frhbb,rhbb)
781             do 410 j=jyb,jyt
782             rhbh(j)=cvmgt(0.0,rhbh(j),rhbh(j).le.scr1(j))
783     410     scr2(j)=scr2(j)+rhbh(j)
784     420     call fryfct (rha0(jyb),rhah(jyb),jbt,frhat,rhat,frhab,rhab)
785             do 430 j=jyb,jyt
786             rhah(j)=cvmgt(0.0,rhah(j),rhah(j).le.scr1(j))
787     430     scr2(j)=scr2(j)+rhah(j)
788             call fryfct (rhc0(jyb),rhch(jyb),jbt,frhct,rhct,frhcb,rhcb)
789             do 440 j=jyb,jyt
790             rhch(j)=cvmgt(0.0,rhch(j),rhch(j).le.scr1(j))
791     440     scr2(j)=scr2(j)+rhch(j)
792             do 450 j=jyb,jyt
793     450     scrh(j)=rhoh(j)/scr2(j)
794     460     continue
795 c
796 c       replace variables in 2D arrays.
797             do 470 j=jyb,jyt
798             rho(1,j)=rhoh(j)
799             rvx(1,j)=rvxh(j)
800             rvy(1,j)=rvyh(j)
801     470     erg(1,j)=ergh(j)
802             go to (520,500,480), nspec
803     480     do 490 j=jyb,jyt
804     490     rhb(1,j)=rhbh(j)*scrh(j)
805     500     do 510 j=jyb,jyt
806     510     rha(1,j)=rhah(j)*scrh(j)
807 c
808 c       loop back for the next col to be integrated.
809     520     continue
810             return
811 c
812             entry bndrx(bcr,bcl)
813 c-----------------------------------------------------------------------
814 c       sets up x boundary conditions for next call to integx.
815 c         notation :
816 c             bc = 0   outflow,
817 c                = 1   subsonic (ambient),
818 c                =-1   reflective,
819 c                = 2   prescribed.
820 c-----------------------------------------------------------------------
821 c       check if active grid includes boundaries.
822             abcl=cvmgt(bcl,0.0,ix1.eq.1)
823             abcr=cvmgt(bcr,0.0,ixr.eq.mnx)
824 c
825 c       density
826             frhor=cvmgt(0.0,1.0,abcr.ge.1.0)
827             rhor=cvmgt(rhorbc,0.0,abcr.eq.2.0)
828             frhol=cvmgt(0.0,1.0,abcl.ge.1.0)
829             rhol=cvmgt(rholbc,0.0,abcl.eq.2.0)
830 c       species density.
831             frhar=1.0
832             rhar=0.0
```

```
            frhal=1.0
            rhal=0.0
            frhbr=1.0
            rhbr=0.0
            frhbl=1.0
            rhbl=0.0
            frhcr=frhor
            rhcr=rhor
            frhcl=frhol
            rhcl=rhol
c
c     x - component of momentum.
            frvxr=cvmgt(abcr,amax1(0.0,1.0-abcr),abcr.eq.-1.0)
            rvxr=cvmgt(rvxrbc,0.0,abcr.eq.2.0)
            frvxl=cvmgt(abcl,amax1(0.0,1.0-abcl),abcl.eq.-1.0)
            rvxl=cvmgt(rvxlbc,0.0,abcl.eq.2.0)
c
c     y - component of momentum.
            frvyr=cvmgt(1.0,amax1(0.0,1.0-abcr),abcr.eq.-1.0)
            rvyr=cvmgt(rvyrbc,0.0,abcr.eq.2.0)
            frvyl=cvmgt(1.0,amax1(0.0,1.0-abcl),abcl.eq.-1.0)
            rvyl=cvmgt(rvylbc,0.0,abcl.eq.2.0)
c
c     energy.
            fergr=cvmgt(0.0,1.0,abcr.ge.1.0)
            ergr=cvmgt(ergrbc,0.0,abcr.eq.2.0)
            fergl=cvmgt(0.0,1.0,abcl.ge.1.0)
            ergl=cvmgt(erglbc,0.0,abcl.eq.2.0)
c
            entry bndry(bct,bcb)
c-----------------------------------------------------------------------
c     sets up y boundary conditions for next call to integy.
c-----------------------------------------------------------------------
c     check if active grid includes boundaries.
            abcb=cvmgt(bcb,0.0,jyb.eq.1)
            abct=cvmgt(bct,0.0,jyt.eq.mny)
c
c     density.
            frhot=cvmgt(0.0,1.0,abct.ge.1.0)
            rhot=cvmgt(rh00(jyt+2),0.0,abct.eq.1.0)
            rhot=cvmgt(rhotbc,rhot,abct.eq.2.0)
            frhob=cvmgt(0.0,1.0,abcb.ge.1.0)
            rhob=cvmgt(rh00(jyb),0.0,abcb.eq.1.0)
            rhob=cvmgt(rhobbc,rhob,abcb.eq.2.0)
c
c     species density.
            frhat=1.0
            rhat=0.0
            frhab=1.0
            rhab=0.0
            frhbt=1.0
            rhbt=0.0
            frhbb=1.0
            rhbb=0.0
            frhct=frhot
            rhct=rhot
            frhcb=frhob
            rhcb=rhob
c
c     x - component of momentum.
            frvxt=cvmgt(1.0,amax1(0.0,1.0-abct),abct.eq.-1.0)
            rvxt=cvmgt(rvxtbc,0.0,abct.eq.2.0)
            frvxb=cvmgt(1.0,amax1(0.0,1.0-abcb),abcb.eq.-1.0)
            rvxb=cvmgt(rvxbbc,0.0,abcb.eq.2.0)
```

```
c
c     y - component of momentum.
          frvyt=cvmgt(abct,amax1(0.0,1.0-abct),abct.eq.-1.0)
          rvyt=cvmgt(rvytbc,0.0,abct.eq.2.0)
          frvyb=cvmgt(abcb,amax1(0.0,1.0-abcb),abcb.eq.-1.0)
          rvyb=cvmgt(rvybbc,0.0,abcb.eq.2.0)
c
c     energy.
          fergt=cvmgt(0.0,1.0,abct.ge.1.0)
          ergt=cvmgt(er00(jyt+2),0.0,abct.eq.1.0)
          ergt=cvmgt(ergtbc,ergt,abct.eq.2.0)
          fergb=cvmgt(0.0,1.0,abcb.ge.1.0)
          ergb=cvmgt(er00(jyb),0.0,abcb.eq.1.0)
          ergb=cvmgt(ergbbc,ergb,abcb.eq.2.0)
          return
        end
          subroutine diagno (nstep,time)
c-------------------------------------------------------------
c     performs general diagnostics.
c-------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (mnxx=1, mnyy=1)
          parameter (mxx=1)
          parameter (nop=1, nsta=1)
          integer otape
          integer alpha
          logical lmgrid, lstat, lnuc, therml, lgrav, leos
          real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
          real tem(mnxx,mnyy), sci(mnxx,mnyy), rho(mnx,mny)
          real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
          common nx, ny, nxp, nyp, nspec, fdump, fdiag, lmgrid
          common lstat, lnuc, therml, lgrav, leos
          common alpha, gamma, gm1, dt, dx, dy, rhomin
          common tem, sci, rho, rvx, rvy, erg, rha, rhb, rhc
c
          real pre(mnx,mny)
          equivalence (pre,rvx)
          real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
          common /grids/ xcor, xcoro, ycor, ycoro, unit
c
          real xbnd(5), dxr(4), ybnd(3), dyr(2)
          common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1    xbnd, ybnd, dxr, dyr, xleft
c
          real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
          real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
          real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
          real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
          real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
          real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
          real rmin(mxy), rmax(mxy), rvr(mxy)
          common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
          real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
          equivalence (rscr,vel), (pscr,rvr), (gscr,uh), (scr1,vh)
          equivalence (scr2,scrh)
c
          logical part
          real xp(nop), yp(nop)
          common /part/ part, xp, yp, nopp
c
          real rhs(mxx,nsta), vls(mxx,nsta), qms(mxx,nsta), tme(mxx)
```

```
      real prs(mxx,nsta), xs(nsta), ys(nsta)
      common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
     1    ixx
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1    , jtherm
c
      logical lscan, rigt, left, ltop, botm
      common /scan/ lscan, rigt, left, ltop, botm
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1    jbt, iadd, jadd, factor
c
c     compute the various conservation sums for the system.
      call ngridd (xcor,nxp,alpha)
      call ogridd (nxp)
      call ngridd (xcor,nxp,alpha)
      do 30 j=1,ny
      call consrd (rho(1,j),nx,rho0(j))
      call consrd (rvx(1,j),nx,rvx0(j))
      call consrd (rvy(1,j),nx,rvy0(j))
      call consrd (erg(1,j),nx,erg0(j))
      go to (30,20,10), nspec
   10 call consrd (rhb(1,j),nx,rhb0(j))
   20 call consrd (rha(1,j),nx,rha0(j))
   30 continue
      call ngridd (ycor,nyp,1)
      call ogridd (nyp)
      call ngridd (ycor,nyp,1)
      call consrd (rho0,ny,rhosum)
      call consrd (rvx0,ny,rvxsum)
      call consrd (rvy0,ny,rvysum)
      call consrd (erg0,ny,ergsum)
      go to (60,50,40), nspec
   40 call consrd (rhb0,ny,rhbsum)
   50 call consrd (rha0,ny,rhasum)
   60 continue
      write (6,210) nstep,time,rhasum,rhbsum,rhosum,rvxsum,rvysum
     1    ,ergsum
      return
c
      entry pressg
c-------------------------------------------------------------
c     calculates pressure in the entire grid and stores it in rvx,
c     which is stored temporarily on disk.
c-------------------------------------------------------------
      call scrdmp (rvx,1)
      do 90 j=1,mny
      do 70 i=1,mnx
   70 ergk(i)=amax1(0.0,erg(i,j)-0.5*(rvx(i,j)*rvx(i,j)+rvy(i,j)*rvy
     1    (i,j))/rho(i,j)-rho(i,j)*gp00(j+1))
      if (leos) then
c      call eos (mnx,nspec,rho(1,j),ergk,sgam,pre(1,j),rscr,pscr,gscr
c     1    ,scr1,scr2,rha(1,j),rhb(1,j))
      call eosp1 (rho(1,j),ergk,mnx,sgam,pre(1,j))
      else
      do 80 i=1,mnx
   80 pre(i,j)=gm1*ergk(i)
      endif
   90 continue
      return
c
```

```
1025            entry datain(itape)
1026            read (itape) lx,ly,nspec,lmgrid,lstat,lnuc,lgrav,leos,part
1027         1  ,therml
1028            if (lx.eq.nx.and.ly.eq.ny) go to 100
1029            write (6,220) itape,lx,ly,nx,ny
1030            stop
1031     100    continue
1032            if (.not.lstat) go to 110
1033            read (itape) lxx
1034            read (itape) ((rhs(i,kk),i=1,mxx),kk=1,nsta)
1035            read (itape) ((vls(i,kk),i=1,mxx),kk=1,nsta)
1036            read (itape) ((gms(i,kk),i=1,mxx),kk=1,nsta)
1037            read (itape) ((prs(i,kk),i=1,mxx),kk=1,nsta)
1038            read (itape) (xs(ll),ll=1,nsta)
1039            read (itape) (ys(ll),ll=1,nsta)
1040            read (itape) (tme(i),i=1,mxx)
1041     110    read (itape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
1042            read (itape) lscan,rlgt,left,ltop,botm,lxr,jxr,lxl,jxl,lyt,jyt
1043         1  ,lyb,jyb
1044            go to (140,130,120), nspec
1045     120    read (itape) ((rhb(i,j),i=1,nx),j=1,ny)
1046     130    read (itape) ((rha(i,j),i=1,nx),j=1,ny)
1047     140    continue
1048            read (itape) ((rho(i,j),i=1,nx),j=1,ny)
1049            read (itape) ((pre(i,j),i=1,nx),j=1,ny)
1050            read (itape) ((rvx(i,j),i=1,nx),j=1,ny)
1051            read (itape) ((rvy(i,j),i=1,nx),j=1,ny)
1052            read (itape) ((erg(i,j),i=1,nx),j=1,ny)
1053            if (lnuc) read (itape) radl,fracn,eblast
1054            read (itape) (pr00(i),i=1,mxy)
1055            read (itape) (rh00(i),i=1,mxy)
1056            read (itape) (er00(i),i=1,mxy)
1057            read (itape) (gp00(i),i=1,mxy)
1058            read (itape) (gravy(i),i=1,mxy)
1059            read (itape) (fsky(i),i=1,mxy)
1060            if (therml) read (itape) jtherm,((rhtl(i,j),i=1,mnx),j=1,jthm)
1061            read (itape) dt
1062            if (.not.part) go to 150
1063            read (itape) nopp
1064            read (itape) (xp(i),i=1,nopp)
1065            read (itape) (yp(i),i=1,nopp)
1066     150    continue
1067            read (itape) (xcoro(i),i=1,nxp),(ycoro(j),j=1,nyp)
1068            read (itape) (xbnd(i),i=1,5),(ybnd(j),j=1,3)
1069            read (itape) (dxr(i),i=1,4),(dyr(j),j=1,2)
1070            return
1071 c
1072            entry datout(otape)
1073            write (otape) nx,ny,nspec,lmgrid,lstat,lnuc,lgrav,leos,part
1074         1  ,therml
1075            if (.not.lstat) go to 160
1076            write (otape) lxx
1077            write (otape) ((rhs(i,kk),i=1,mxx),kk=1,nsta)
1078            write (otape) ((vls(i,kk),i=1,mxx),kk=1,nsta)
1079            write (otape) ((gms(i,kk),i=1,mxx),kk=1,nsta)
1080            write (otape) ((prs(i,kk),i=1,mxx),kk=1,nsta)
1081            write (otape) (xs(ll),ll=1,nsta)
1082            write (otape) (ys(ll),ll=1,nsta)
1083            write (otape) (tme(i),i=1,mxx)
1084     160    write (otape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
1085            write (otape) lscan,rlgt,left,ltop,botm,lxr,jxr,lxl,jxl,lyt
1086         1  ,jyt,lyb,jyb
1087            go to (190,180,170), nspec
1088     170    write (otape) ((rhb(i,j),i=1,nx),j=1,ny)
```

```
1089    180      write (otape) ((rha(i,j),i=1,nx),j=1,ny)
1090    190      continue
1091             write (otape) ((rho(i,j),i=1,nx),j=1,ny)
1092             write (otape) ((pre(i,j),i=1,nx),j=1,ny)
1093             call scrdmp (rvx,2)
1094             write (otape) ((rvx(i,j),i=1,nx),j=1,ny)
1095             write (otape) ((rvy(i,j),i=1,nx),j=1,ny)
1096             write (otape) ((erg(i,j),i=1,nx),j=1,ny)
1097             if (inuc) write (otape) radl,fracn,eblast
1098             write (otape) (pr00(i),i=1,mxy)
1099             write (otape) (rh00(i),i=1,mxy)
1100             write (otape) (er00(i),i=1,mxy)
1101             write (otape) (gp00(i),i=1,mxy)
1102             write (otape) (gravy(i),i=1,mxy)
1103             write (otape) (fsky(i),i=1,mxy)
1104             if (therml) write (otape) jtherm,((rhtl(i,j),i=1,mnx),j=1,jthm
1105        1    )
1106             write (otape) dt
1107             if (.not.part) go to 200
1108             write (otape) nopp
1109             write (otape) (xp(i),i=1,nopp)
1110             write (otape) (yp(i),i=1,nopp)
1111    200      continue
1112             write (otape) (xcoro(i),i=1,nxp),(ycoro(j),j=1,nyp)
1113             write (otape) (xbnd(i),i=1,5),(ybnd(j),j=1,3)
1114             write (otape) (dxr(i),i=1,4),(dyr(j),j=1,2)
1115             return
1116 c
1117    210 format ('0at step ',i4,'  and time ',1pe12.4,'  the conservation s
1118       1ums are ...',/,'  sum of rha    sum of rhb    sum of rho  ','  sum
1119       2 of rvx    sum of rvy    sum of erg  '/6e14.6)
1120    220 format ('0datain input error on itape =',i3,'   the grid ','size o
1121       1ff tape is ',i3,' by ',i3,'   but fast2d ','expects ',i3,' by ',i3
1122       2 )
1123        end
1124             subroutine scrdmp (arr,io12)
1125 c-------------------------------------------------------
1126 c     temporarily stores data on disk.
1127 c-------------------------------------------------------
1128             parameter (mnx=150, mny=200, mxy=202)
1129             real arr(mnx,mny)
1130             go to (10,20), io12
1131     10      open (unit=10,file='store',status='new')
1132             write (10) ((arr(i,j),i=1,mnx),j=1,mny)
1133             return
1134     20      rewind (10)
1135             read (10) ((arr(i,j),i=1,mnx),j=1,mny)
1136             close (unit=10,status='delete')
1137             return
1138        end
1139             subroutine fryfct (rhoo,rhon,na,rbc,rhorbc,lbc,rholbc)
1140 c
1141             real rhoo(na), rhon(na)
1142 cd
1143 cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1144 cd
1145 cd    fryfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc)
1146 cd    originator - fry and book       code 4040, nrl          june 1982
1147 cd    does multicoefficient fct with a switch to choose either
1148 cd    of the form       drho/dt = -div (rho*v) + sources       in either
1149 cd    cartesian, cylindrical, or spherical geometry. the finite-differ-
1150 cd    ence grid can be eulerian, sliding rezone, or lagrangian and can
1151 cd    be arbitrarily spaced. the algorithm used is a low-phase error fct
1152 cd    algorithm, vectorized and optimized for speed. a complete descrip-
```

```
1153 cd    tion appears in nrl memo report 3237, "flux-corrected transport
1154 cd    modules for solving generalized continuity equations".
1155 cd
1156 cd    arguments:  in this routine the right boundary radh(np) is half a
1157 cd    cell beyond the last grid point n at rnc(n) and the left boundary
1158 cd    radh(1) is half a cell before the first grid point at rnc(1).
1159 cd    rhoo    real array(na)    grid point densities at start of step  i
1160 cd    rhon    real array(na)    grid point densities at end of step    o
1161 cd    na      integer           number of cells in the system          i
1162 cd    rbc     real              right boundary condition factor        i
1163 cd    rhorbc  real              right guard cell value of rho          i
1164 cd    lbc     real              left boundary condition factor         i
1165 cd    rholbc  real              left guard cell value of rho           i
1166 cd
1167 cd    language and limitations:  the subroutine jpbfct is a multiple-
1168 cd    entry fortran routine in single precision (32 bits asc).  the asc
1169 cd    parameter statement is used to set symbolically the internal array
1170 cd    dimensions.  underflows are possible when the function being trans-
1171 cd    ported has many zeroes.  the calculations generally misconserve by
1172 cd    one or two bits per cycle.  the relative phase and amplitude errors
1173 cd    (for smooth functions) are typically several percent for charac-
1174 cd    teristic lengths of 1 - 2 cells (wavelengths of order 10 cells).
1175 cd    shocks are generally accurate to better than 1 percent.  this sub-
1176 cd    routine must be compiled with the y option to force storage and
1177 cd    retention of internal variables.  alternatively a common block can
1178 cd    be added to accomplish the same end.
1179 cd
1180 cd    entry points:  jhgfct, dlbfct, ogridd, ngridd, velocd, sourcd, and
1181 cd    consrd.  the detailed documentation (or the listing below) gives
1182 cd    the explanation and use of the arguments to these other entries.
1183 cd
1184 cd    no auxiliary or library routines are called by jpbfct.
1185 cd
1186 cd    . . . . . .
1187 c
1188           parameter (npt=202)
1189 c
1190           logical lsourc, lmom, logic(2)
1191           real rbc, lbc, mask1, mask2
1192           real mask3
1193           real scrh(npt), rhot(npt), diff(npt), source(npt)
1194           real flxh(npt), rulh(npt), mulh(npt), adugth(npt)
1195           real epsh(npt), fsgn(npt), fabs(npt), lnrhot(npt)
1196           real terp(npt), term(npt), psgn(npt), lorhot(npt)
1197           real int(npt), flxm(npt), flxp(npt), deltap(npt)
1198           real rlo(npt), ih(npt), rih(npt), adudth(npt)
1199           real vdtodr(npt), gamma(npt)
1200 c
1201 c     common block /fctcom/ contains grid, geometry and cell information
1202 c     to be communicated between subroutines, hydrodynamic utilities,
1203 c     and the main program.
1204 cd    roh     real array(na+1)  old interface locations set by ogridd
1205 cd    rnh     real array(na+1)  new interface locations set by ngridd
1206           real roh(npt), rnh(npt), roc(npt), rnc(npt)
1207           real ah(npt), lo(npt), in(npt), rin(npt)
1208           common /fctcom/ roh, rnh, roc, rnc, ah, lo, in, rin
1209 c
1210           real e(na), r(na)
1211           real uh(npa)
1212           integer ind(nsh)
1213           integer alpha
1214           real radh(npa)
1215           real c(na), d(na)
1216           real rho(na)
```

```
1217 c
1218           data fctone, roh, source, lsourc, lmom /1.0,npt*1.,npt*0.,
1219       1    .false.,.false./
1220           data pi /3.14159265358979323846264338327950288419716939937511/
1221           data ftpi /4.18879020478638608437904622405767440795898437 5/
1222           data c1, c2, c3, c4 /0.16666666666666666666666666666666666666666666666
1223       1    666666666666666667,0.33333333333333333333333333333333333333333333
1224       2    333333333333333333,0.13333333333333333333333333333333333333333333
1225       3    333333333333333333,0.03333333333333333333333333333333333333333333
1226       4    333333333333333333/
1227           equivalence (logic(1),lsourc), (logic(2),lmom)
1228           equivalence (lnrhot(1),lorhot(1))
1229           equivalence (flxh(1),scrh(1)), (term(1),source(1))
1230           equivalence (fabs(1),scrh(1)), (terp(1),source(1))
1231           equivalence (psgn(1),int(1))
1232 c
1233           lmom=.false.
1234           go to 10
1235 c
1236 c     ----------------------------------------------------------------
1237 c
1238       entry jhgfct(rhoo,rhon,na,rbc,rhorbc,lbc,rholbc,e,r)
1239 c
1240 cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1241 cd
1242 cd    jhgfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc, e, r)
1243 cd    description:  this entry allows the flux corrector in the momentum
1244 cd    equation to be limited in such a way as to prevent new maxima
1245 cd    and minima in the pressure.
1246 cd
1247 cd    aruments:  this entry uses the same arguments as jpbfct with the
1248 cd    addition of arguements e and r
1249 cd    e       real array(na)   grid point total energy density        |
1250 cd    r       real array(na)   grid point mass density                |
1251 cd
1252 cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1253 c
1254 c     calculate the diffusive and convective fluxes.
1255           lmom=.true.
1256    10     n=na
1257           np=n+1
1258           do 20 i=2,n
1259           flxh(i)=0.5*adudth(i)*(rhoo(i)+rhoo(i-1))
1260    20     diff(i)=(rhoo(i)-rhoo(i-1))
1261           rhol=rhoo(1)*lbc+rholbc
1262           rhor=rhoo(n)*rbc+rhorbc
1263           diff(1)=(rhoo(1)-rhol)
1264           diff(np)=(rhor-rhoo(n))
1265           flxh(1)=0.5*adudth(1)*(rhoo(1)+rhol)
1266           flxh(np)=0.5*adudth(np)*(rhor+rhoo(n))
1267 c
1268 c     calculate lambdao*rhot, the transported mass elements.
1269           do 30 i=1,n
1270    30     lorhot(i)=lo(i)*rhoo(i)-(flxh(i+1)-flxh(i))
1271 c
1272 c     add in the source terms as appropriate.
1273           if (.not.lsourc) go to 50
1274           do 40 i=1,n
1275    40     lorhot(i)=lorhot(i)+source(i)
1276    50     continue
1277           go to 90
1278 c
1279 c     ----------------------------------------------------------------
1280 c
```

```
      entry dlbfct(rhoo,rhon,na,rbc,rhorbc,lbc,rholbc)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    dlbfct (rhoo, rhon, na, rbc, rhorbc, lbc, rholbc)
cd    description: this entry performs nonconservative convection. the
cd    equation      d(rho)/dt = - v . grad(rho) + source    is solved.
cd
cd    arguments:  this routine uses the same arguments as jpbfct.
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c     calculate the diffusive and convective fluxes.
          lmom=.false.
          n=na
          np=n+1
          do 60 i=2,n
          diff(i)=rhoo(i)-rhoo(i-1)
          flxh(i)=vdtodr(i)*diff(i)
   60     diff(i)=mulh(i)*diff(i)
          rhol=rhoo(1)*lbc+rholbc
          rhor=rhoo(n)*rbc+rhorbc
          diff(1)=mulh(1)*(rhoo(1)-rhol)
          diff(np)=mulh(np)*(rhor-rhoo(n))
          flxh(1)=vdtodr(1)*(rhoo(1)-rhol)
          flxh(np)=vdtodr(np)*(rhor-rhoo(n))
c
c     calculate lambdao*rhot, the transported mass elements.
          do 70 i=1,n
   70     lorhot(i)=ln(i)*(rhoo(i)-0.5*(flxh(i+1)+flxh(i)))
c
c     add in the source terms as appropriate.
          if (.not.lsourc) go to 90
          do 80 i=1,n
   80     lorhot(i)=lorhot(i)+source(i)
c
c     calculate the phoenical antidiffusive fluxes here.
   90     continue
          do 100 i=1,n
  100     lnrhot(i)=lorhot(i)+gamma(i+1)*diff(i+1)-gamma(i)*diff(i)
          do 110 i=1,n
  110     rhot(i)=lorhot(i)*rln(i)
          rhotr=rbc*rhot(n)+rhorbc
          rhotl=lbc*rhot(1)+rholbc
c
          do 120 i=2,n
  120     flxh(i)=mulh(i)*(rhot(i)-rhot(i-1))
          flxh(1)=mulh(1)*(rhot(1)-rhotl)
          flxh(np)=mulh(np)*(rhotr-rhot(n))
c
c     diffuse the solution rhot using old fluxes.
          do 130 i=1,n
  130     lnrhot(i)=lorhot(i)+mulh(i+1)*diff(i+1)-mulh(i)*diff(i)
c
c     calculate the transported/diffused density and grid differences.
          do 140 i=1,n
  140     rhot(i)=lnrhot(i)*rln(i)
          rhotr=rbc*rhot(n)+rhorbc
          rhotl=lbc*rhot(1)+rholbc
          do 150 i=2,n
  150     diff(i)=rhot(i)-rhot(i-1)
          diff(1)=rhot(1)-rhotl
          diff(np)=rhotr-rhot(n)
c
```

```
c     calculate the sign and magnitude of the antidiffusive flux.
          do 160 i=1,np
160       fsgn(i)=sign(fctone,diff(i))
          do 170 i=1,np
170       fabs(i)=abs(flxh(i))
c
c     calculate the flux-limiting based on the pressure for the momemtum
          if (.not.lmom) go to 280
          do 180 i=1,n
180       int(i)=e(i)-0.5*rhot(i)*rhot(i)/r(i)
          do 190 i=2,n
190       deltap(i)=int(i)-int(i-1)
          deltap(1)=int(1)-lbc*int(1)
          deltap(np)=rbc*int(n)-int(n)
          do 200 i=1,n
          flxp(i+1)=-rhot(i)*(1.-sqrt(amax1(0.,1.-2.*r(i)*deltap(i+1)/
     1    (rhot(i)*rhot(i)))))
          flxm(i)=+rhot(i)*(1.-sqrt(amax1(0.,1.+2.*r(i)*deltap(i)/(rhot
     1    (i)*rhot(i)))))
200       continue
          do 210 i=1,np
210       deltap(i)=nulh(i)*deltap(i)
          do 220 i=1,n
220       int(i)=(int(i)*ln(i)+deltap(i+1)-deltap(i))*rln(i)
          do 230 i=2,n
230       deltap(i)=rhot(i)*(int(i-1)-int(i))
          deltap(1)=rhot(1)*(lbc*int(1)-int(1))
          deltap(np)=rhot(n)*(int(n)-rbc*int(n))
          do 240 i=1,np
240       psgn(i)=sign(fctone,deltap(i))
          do 250 i=2,n
250       terp(i)=fsgn(i)*psgn(i)*ln(i)*amin1(psgn(i)*diff(i+1),psgn(i)
     1    *flxm(i-1),psgn(i)*flxp(i+1))
          terp(1)=fsgn(1)*psgn(1)*ln(1)*amin1(psgn(1)*diff(2),psgn(1)
     1    *flxp(2))
          terp(np)=1.0e+75
          do 260 i=1,np
260       fabs(i)=amin1(terp(i),fabs(i))
          do 270 i=2,n
270       term(i)=psgn(i)*fsgn(i)*ln(i-1)*amin1(psgn(i)*diff(i-1),psgn(i
     1    )*flxp(i+1),psgn(i)*flxm(i-1))
          term(np)=psgn(np)*fsgn(np)*ln(n)*amin1(psgn(np)*diff(n),psgn
     1    (np)*flxm(n))
          term(1)=1.0e+75
          go to 320
c
c     calculate the flux-limiting changes on the right and the left.
280       do 290 i=1,n
290       terp(i)=fsgn(i)*ln(i)*diff(i+1)
          terp(np)=1.0e+75
          rhotrr=rbc*rhot(n-1)+rhorbc
          terp(np)=fsgn(np)*ln(n)*(rhotrr-rhotr)
          do 300 i=1,np
300       fabs(i)=amin1(terp(i),fabs(i))
          do 310 i=2,np
310       term(i)=fsgn(i)*ln(i-1)*diff(i-1)
          term(1)=1.0e+75
c
c     terp and term may also be estimated off the ends of the system.
          rhotll=lbc*rhot(2)+rholbc
          term(1)=fsgn(1)*ln(1)*(rhotl-rhotll)
c
c     correct the fluxes completely now.
320       do 330 i=1,np
```

```
      330     flxh(i)=fsgn(i)*amax1(0.0,amin1(fabs(i),term(i)))
c
c     calculate the new flux-corrected densities.
          do 340 i=1,n
      340     rhon(i)=rln(i)*(lnrhot(i)-(flxh(i+1)-flxh(i)))
          lsourc=.false.
          do 350 i=1,np
      350     source(i)=0.0
          return
c
c     ---------------------------------------------------------------
c
          entry velocd(uh,npa,dt,ncol)
c
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
cd
cd    velocd (uh, npa, dt)
cd    description:  this entry calculates all velocity-dependant coeffs.
cd    for multicoefficient fct
cd
cd    arguments:
cd    uh      real array(npa)   flow velocity at cell interfaces       i
cd    npa     integer           number of cell interfaces = n + 1      i
cd    dt      real              stepsize for the time integration      i
cd    ncol    integer that specifies the number of n values
cd
cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
c     calculate the interface area x velocity differential x dt.
          np=npa
          n=np-1
          do 360 i=1,np
      360     adudth(i)=ah(i)*dt*uh(i)-adugth(i)
c
c     calculate the half-cell epsilon (v*dt/dx)
          do 370 i=1,np
      370     epsh(i)=adudth(i)*rlh(i)
c
c     next calculate the diffusion and antidiffusion coefficients.
c     variation with epsilon means fourth-order accurate phases.
          if (ncol.eq.n) go to 400
          do 380 i=1,np
          gamma(i)=0.2+0.2*epsh(i)*epsh(i)
          nulh(i)=-c4+c3*epsh(i)**2
      380 mulh(i)=c1-c1*epsh(i)**2
          if (ncol.eq.0) go to 420
          do 390 i=1,ncol
          nulh(i)=c1+c2*epsh(i)*epsh(i)
          mulh(i)=0.25-0.5*nulh(i)
      390 gamma(i)=0.0
          go to 420
      400 continue
          do 410 i=1,np
          nulh(i)=c1+c2*epsh(i)*epsh(i)
          mulh(i)=0.25-0.5*nulh(i)
          gamma(i)=0.0
      410 continue
      420 continue
          rdt=1.0/dt
          do 430 i=1,np
          diff(i)=uh(i)*rdt*(rnh(i)-rnh(i))
          nulh(i)=lh(i)*nulh(i)
          gamma(i)=lh(i)*gamma(i)
      430 mulh(i)=lh(i)*mulh(i)
```

```
1473 c
1474 c     now calculate vdtodr for cnvfct.
1475           dt2=2.0*dt
1476           dt4=4.0*dt
1477           do 440 i=2,n
1478    440    vdtodr(i)=dt4*diff(i)/(rnh(i+1)-rnh(i-1)+roh(i+1)-roh(i-1))
1479           vdtodr(1)=dt2*diff(1)/(rnh(2)-rnh(1)+roh(2)-roh(1))
1480           vdtodr(np)=dt2*diff(np)/(rnh(np)-rnh(n)+roh(np)-roh(n))
1481           return
1482 c
1483 c     ------------------------------------------------------------------
1484 c
1485       entry setbd(ind,nsh)
1486 c
1487 cd    ******************************************************************
1488 cd
1489 cd    setbd (ind, nsh)
1490 cd    description:  this entry sets some velocity dependent parameters
1491 cd    to zero at the various material boundries to inhibit diffusion
1492 cd    of material across the boundries.
1493 cd
1494 cd    arguments:
1495 cd    ind     integer array(nsh)   end boundary indices for material   i
1496 cd                                 interfaces.
1497 cd    nsh     integer              number of material interfaces.      i
1498 cd
1499 cd    ******************************************************************
1500 c
1501 c     loop over interfaces.
1502           do 450 is=1,nsh
1503           i=ind(is)
1504           adudth(i)=0.0
1505           nuih(i)=0.0
1506           muih(i)=0.0
1507           gamma(i)=0.0
1508    450    continue
1509 c
1510           return
1511 c
1512 c     ------------------------------------------------------------------
1513 c
1514       entry ngridd(radh,npa,alpha)
1515 c
1516 cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1517 cd
1518 cd    ngride (radh, npa, alpha)
1519 cd    description:  this entry sets new geometry variables and coeffs.
1520 cd
1521 cd    arguments:
1522 cd    radh    real array(npa)   new cell interface positions           i
1523 cd    npa     integer           number of cell interfaces = n + 1      i
1524 cd    alpha   integer           = 1   for cartesian geometry           i
1525 cd                              = 2   for cylindrical geometry         i
1526 cd                              = 3   for spherical geometry           i
1527 cd                              = 4   for nozzle geometry              i
1528 cd
1529 cd    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1530 c
1531 c     calculate the new half-cell positions and grid changes.
1532           np=npa
1533           n=np-1
1534           malpha=iabs(alpha)
1535           do 460 i=1,np
1536    460    rnh(i)=radh(i)
```

```
1537            do 470 i=1,n
1538    470     rnc(i)=0.5*(rnh(i)+rnh(i+1))
1539 c
1540 c       calculate the three coordinate systems.
1541            go to (480,530,560), malpha
1542 c
1543 c       cartesian coordinates.
1544    480     do 490 i=1,np
1545    490     ah(i)=1.0
1546            if (alpha.gt.0) go to 510
1547            do 500 i=1,n
1548    500     ln(i)=rnh(2)-rnh(1)
1549            go to 600
1550    510     do 520 i=1,n
1551    520     ln(i)=rnh(i+1)-rnh(i)
1552            go to 600
1553 c
1554 c       cylindrical coordinates.
1555    530     do 540 i=1,np
1556            diff(i)=rnh(i)*rnh(i)
1557    540     ah(i)=pi*(roh(i)+rnh(i))
1558            do 550 i=1,n
1559    550     ln(i)=pi*(diff(i+1)-diff(i))
1560            go to 600
1561 c
1562 c       spherical coordinates.
1563    560     do 570 i=1,np
1564            diff(i)=rnh(i)*rnh(i)*rnh(i)
1565    570     scrh(i)=(roh(i)+rnh(i))*roh(i)
1566            do 580 i=1,np
1567    580     ah(i)=ftpi*(scrh(i)+rnh(i)**2)
1568            do 590 i=1,n
1569    590     ln(i)=ftpi*(diff(i+1)-diff(i))
1570 c
1571 c       now the geometric variables which are system independant.
1572    600     do 610 i=2,n
1573    610     lh(i)=amin1(ln(i),ln(i-1))
1574            lh(1)=amin1(ln(1),ln(2))
1575            lh(np)=amin1(ln(n),ln(n-1))
1576            do 620 i=1,n
1577    620     rln(i)=1.0/ln(i)
1578            if (alpha.gt.0) go to 640
1579            do 630 i=1,np
1580    630     adugth(i)=ah(i)*(rnh(i)-roh(i))
1581            go to 660
1582    640     do 650 i=1,np
1583    650     adugth(i)=ah(i)*(rnh(i)-roh(i))
1584    660     do 670 i=2,n
1585    670     rlh(i)=0.5*(rln(i)+rln(i+1))
1586            rlh(1)=rln(1)
1587            rlh(np)=rln(n)
1588            return
1589 c
1590 c       ---------------------------------------------------------------
1591 c
1592        entry sourcd(na,dt,modes,c,d,dr,dl)
1593 c
1594 cd      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1595 cd
1596 cd      sourcd (na, dt, modes, c, d, dr, dl)
1597 cd      description:  this entry accumulates different source terms.
1598 cd
1599 cd      arguments.
1600 cd      na      integer             number of cells in the system ( = n)    i
```

```
1601 cd     dt      real                stepsize for the time integration      |
1602 cd     modes   integer             = 1   computes + div (d)               |
1603 cd                                 = 2   computes + c*grad (d)            |
1604 cd                                 = 3   adds + d to the sources          |
1605 cd                                 = 4   + div(d) at interfaces           |
1606 cd                                 = 5   + c*grad(d) at interfaces        |
1607 cd                                 = 6   dummy for now                    |
1608 cd     c       real array(na)      array of source variables at grid pts  |
1609 cd     d       real array(na)      array of source variables at grid pts  |
1610 cd     dr      real                right boundary value of d (if used)    |
1611 cd     dl      real                left boundary value of d (if used)     |
1612 cd
1613 cd     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
1614 c
1615             n=na
1616             np=n+1
1617             dth=0.5*dt
1618             dtq=0.25*dt
1619             go to (680,710,750,770,800,840), modes
1620 c
1621 c       + div(d) is computed conservatively and added to the sources.
1622    680      do 690 i=2,n
1623    690      scrh(i)=dth*ah(i)*(d(i)+d(i-1))
1624             scrh(1)=dth*ah(1)*(dl+dl)
1625             scrh(np)=dth*ah(np)*(dr+dr)
1626             do 700 i=1,n
1627    700      source(i)=source(i)+(scrh(i+1)-scrh(i))
1628             lsourc=.true.
1629             return
1630 c
1631 c       + c*grad(d) is computed efficiently and added to the sources.
1632    710      do 720 i=2,n
1633    720      scrh(i)=dtq*(d(i)+d(i-1))
1634             scrh(1)=dtq*(dl+dl)
1635             scrh(np)=dtq*(dr+dr)
1636             do 730 i=1,n
1637    730      diff(i)=scrh(i+1)-scrh(i)
1638             do 740 i=1,n
1639    740      source(i)=source(i)+c(i)*(ah(i+1)+ah(i))*diff(i)
1640             lsourc=.true.
1641             return
1642 c
1643 c       + d is added to the sources in an explicit formulation.
1644    750      do 760 i=1,n
1645    760      source(i)=source(i)+dt*lo(i)*d(i)
1646             lsourc=.true.
1647             return
1648 c
1649 c       + div(d) is computed conservatively from interface data.
1650    770      do 780 i=1,np
1651    780      scrh(i)=dt*ah(i)*d(i)
1652             do 790 i=1,n
1653    790      source(i)=source(i)+(scrh(i+1)-scrh(i))
1654             lsourc=.true.
1655             return
1656 c
1657 c       + c*grad(d) is computed using interface data for d.
1658    800      do 810 i=1,np
1659    810      scrh(i)=dth*d(i)
1660             do 820 i=1,n
1661    820      diff(i)=scrh(i+1)-scrh(i)
1662             do 830 i=1,n
1663    830      source(i)=source(i)+c(i)*diff(i)*(ah(i)+ah(i+1))
1664             lsourc=.true.
```

```
1665            return
1666 c
1667 c       a dummy for future source terms.
1668 c           . . . .
1669 c           . . . .
1670    840     continue
1671            return
1672 c
1673 c       ---------------------------------------------------------------
1674 c
1675        entry ogridd(npa)
1676 c
1677 cd      . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1678 cd
1679 cd      ogridd (npa)
1680 cd      description:  this entry copies old grid and geometry variables
1681 cd      into arrays designated to hold them when new values are defined.
1682 cd
1683 cd      arguments:
1684 cd      npa     integer           number of interfaces in the system     i
1685 cd
1686 cd      . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1687 c
1688 c       copy the previously new grid values to be used as the old grid.
1689            n=npa-1
1690            np=npa
1691            do 850 i=1,n
1692            ro(i)=rn(i)
1693            roc(i)=rnc(i)
1694    850     rlo(i)=rln(i)
1695            roh(1)=rlo(1)
1696            do 860 i=1,np
1697    860     roh(i)=rnh(i)
1698            return
1699 c
1700 c       ---------------------------------------------------------------
1701 c
1702        entry consrd(rho,na,csum)
1703 c
1704 cd      . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1705 cd
1706 cd      consrd (rho, na, csum)
1707 cd      description:  this entry computes the ostensibly conserved sum.
1708 cd
1709 cd      arguments:
1710 cd      rho     real array(na)    cel values for physical variable (rho) i
1711 cd      na      integer           number of cells in the system          i
1712 cd      csum    real              value of the conservation sum of rho   o
1713 cd
1714 cd      . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1715 cd
1716 c       compute the ostensibly conserved total mass (beware your b.c.)
1717            n=na
1718            csum=0.0
1719            do 870 i=1,n
1720    870     csum=csum+ln(i)*rho(i)
1721            return
1722        end
1723            subroutine dtset (istep,time,radl,ensum)
1724 c       ---------------------------------------------------------------
1725 c       calculates time step as one-half the courant time step
1726 c       to insure positivity for FCT.
1727 c       ---------------------------------------------------------------
1728            parameter (mn=150, my=200, mx=202)
```

```
      parameter (mnxx=1, mnyy=1)
      integer alpha
      logical lmgrid, lstat, lnuc, therml, lgrav, leos
      real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
      real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
      common nx, ny, nxp, nyp, nspec, idump, idtag, lmgrid
      common lstat, lnuc, therml, lgrav, leos
      common alpha, gamma, gm1, dt, dx, dy, rhomin
      common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
      common /grids/ xcor, xcoro, ycor, ycoro, unit
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1   rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2   , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
      common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
     1   jbt, iadd, jadd, factor
c
      real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
      parameter (jthm=1)
      real fsky(mxy), rhtl(mnx,jthm)
      common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
     1   , jtherm
c
      real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
      equivalence (rscr,rmin), (pscr,rmax), (gscr,rvr), (scr1,sgrv)
      equivalence (scr2,scrh)
      real gams(mxy), vels(mxy), deltat(mxy)
      equivalence (sgam,gams), (srvx,vels), (deltat,srvy)
c
      common /tstep/ dtmin, dtmax
c
      dtold=dt
      vfluid=0.0
      vsound=0.0
      tmstep=dtmax
      it=0
      jt=0
c
c     get delta x and delta y.
      do 10 i=ixl,ixr
   10 delx(i)=xcor(i+1)-xcor(i)
      do 20 j=jyb,jyt
   20 dely(j)=ycor(j+1)-ycor(j)
c
      do 70 j=jyb,jyt
c
c     maximum fluid velocity and sound speed calculation.
      do 30 i=ixl,ixr
      rhom(i)=1.0/rho(i,j)
      vel(i)=(rvx(i,j)*rvx(i,j)+rvy(i,j)*rvy(i,j))*rhom(i)
      ergk(i)=amax1(0.0,erg(i,j)-0.5*vel(i)-rho(i,j)*gp00(j+1))
   30 vel(i)=sqrt(vel(i)*rhom(i))
      if (leos) then
```

```
1793 c           call eos (ilr,nspec,rho(ixl,j),ergk(ixl),gams(ixl),pres(ixl)
1794 c     1     ,rscr(ixl),pscr(ixl),gscr(ixl),scr1(ixl),scr2(ixl),rha0(ixl)
1795 c     2     ,rhb0(ixl))
1796           call eospi (rho(ixl,j),ergk(ixl),ilr,gams(ixl),pres(ixl))
1797           do 40 i=ixl,ixr
1798    40     vels(i)=sqrt(gams(i)*pres(i)*rhom(i))
1799           else
1800           do 50 i=ixl,ixr
1801           pres(i)=gmi*ergk(i)
1802    50     vels(i)=sqrt(gamma*pres(i)*rhom(i))
1803           endif
1804           do 60 i=ixl,ixr
1805    60     deltat(i)=amin1(delx(i),dely(j))/(vel(i)+vels(i))
1806           it=ismin(ilr,deltat(ixl),1)+ixl-1
1807           if (deltat(it).gt.tmstep) go to 70
1808           tmstep=deltat(it)
1809           vfluid=vel(it)
1810           vsound=vels(it)
1811           jt=j
1812    70     continue
1813 c
1814 c     determine the time step
1815           dt=amax1(dtmin,amin1(1.1*dtold,amin1(dtmax,0.5*tmstep)))
1816           dt=cvmgt(0.1*(float(istep)*dt+float(10-istep)*dtmin),dt,istep
1817      1     .lt.10)
1818           time=time+dt
1819           write (6,80) istep,time,dt,vfluid,vsound,it,jt,radl,ixr,jxr
1820      1     ,ixl,jxl,iyt,jyt,iyb,jyb
1821           return
1822 c
1823    80 format ('s',i5,1x,'t=',1pe10.3,'dt=',1pe10.3,'vfs=',2(1pe10.3),'at
1824      1 ',2i4,'r',1pe10.3,'ijrltb',8i4)
1825       end
1826           subroutine restrt (istep,time,restf,labol)
1827 c-------------------------------------------------------
1828 c     reads data from dump file to restart calculation.
1829 c-------------------------------------------------------
1830           parameter (mnx=150, mny=200, mxy=202)
1831           character *8 restf, dumpf, dfile
1832           character *25 path
1833           character *40 glab
1834           character *40 label
1835           character *40 labol
1836           logical lsave
1837           integer otape, ofile
1838           common /inout/ itape, otape, ifile, ofile, lsave, npath, nlab
1839 c
1840           open (unit=itape,file=restf,status='old')
1841           read (itape) nfile,dfile,nstep,time,labol
1842           write (6,60)
1843           write (6,50) ifile,nstep,time,nfile,dfile,labol
1844           if (ifile.ne.nfile) go to 30
1845           if (istep.ne.nstep+1) go to 40
1846           call datain (itape)
1847           close (unit=itape,status='keep')
1848           write (6,60)
1849           return
1850 c
1851           entry dumper(istep,time,dumpf,label,path,glab)
1852 c-------------------------------------------------------
1853 c     this entry dumps data at the specified time step.
1854 c-------------------------------------------------------
1855           write (dfile,100) ofile
1856           do 10 i=5,7
```

```
1857            if (dfile(1:1).eq.' ') dfile(1:1)='0'
1858     10     continue
1859            dfile(1:8)=dumpf(1:4)//dfile(5:8)
1860            open (unit=otape,file=dfile,status='new')
1861            write (6,60)
1862            write (6,90) ofile,istep,time,dfile,label
1863            write (otape) ofile,dfile,istep,time,label
1864 c
1865            call pressg
1866            call datout (otape)
1867            close (unit=otape,status='keep')
1868            if (.not.lsave) go to 20
1869            write (glab,80) path(1:npath),dfile
1870            ier=1000
1871            call mass ('save','\wait=on\',0,glab,ierr)
1872            write (6,70) dfile,ierr
1873     20     ofile=ofile+1
1874            write (6,60)
1875            return
1876 c
1877 c      error exits follow.
1878     30     write (6,110) ifile,nfile
1879            stop
1880     40     write (6,120) istep,nstep
1881            stop
1882 c
1883     50 format ('* * *  restart off file ',i2,' at step no.',i6,' and time
1884       1 =',1pe12.4,/,'       reading file no.',i3,' which is labeled ',a,
1885       2 ': ',a,/)
1886     60 format ('1* * * * * * * * * * * * * ',)
1887     70 format (1x,'ierr for ',a,' =',i4/)
1888     80 format ('&',a,a,'&')
1889     90 format ('* * *  dumped onto file ',i2,' at step no.',i6,' and time
1890       1 =',1pe12.4,//,'       this file is labeled ',a,':',1x,a,//)
1891    100 format (i8)
1892    110 format ('0file required (',i2,') does not equal file found',' ('
1893       1 ,i2,')',//)
1894    120 format (1x,'minstp should be last dump step + 1'/1x,'minstp, last
1895       1dump step=',215/1x,'run aborted'/)
1896        end
1897            subroutine eos (np,nspec,rho,ein,gam,pre,rscr,pscr,gscr,scr1
1898       1    ,scrh,rha,rhb)
1899 c-----------------------------------------------------------------------
1900 c      calculates pressure and gamma for upto three species.
1901 c-----------------------------------------------------------------------
1902            real rho(np), ein(np), gam(np), pre(np), rscr(np), pscr(np)
1903            real gscr(np), scr1(np), scrh(np), rha(np), rhb(np)
1904 c
1905 c      initialize.
1906            do 10 jk=1,np
1907            rscr(jk)=rho(jk)
1908            pre(jk)=0.0
1909            gam(jk)=1.0
1910     10     scr1(jk)=1.0/rscr(jk)
1911            go to (60,40,20), nspec
1912 c
1913 c      species 3.
1914     20     call eosp3 (rho,ein,np,gscr,pscr)
1915            do 30 jk=1,np
1916            scrh(jk)=rhb(jk)*scr1(jk)
1917            scrh(jk)=cvmgt(0.0,scrh(jk),scrh(jk).lt.0.1)
1918            pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
1919            gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
1920     30     rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rhb(jk),scrh(jk).eq.0.0)
```

```
1921 c
1922 c     species 2.
1923    40      call eosp2 (rho,ein,np,gscr,pscr)
1924            do 50 jk=1,np
1925            scrh(jk)=rho(jk)*scr1(jk)
1926            scrh(jk)=cvmgt(0.0,scrh(jk),scrh(jk).lt.0.1)
1927            pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
1928            gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
1929    50      rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rho(jk),scrh(jk).eq.0.0)
1930 c
1931 c     species 1.
1932    60      call eosp1 (rho,ein,np,gscr,pscr)
1933            do 70 jk=1,np
1934            scr1(jk)=rscr(jk)*scr1(jk)
1935            pre(jk)=amax1(0.0,pre(jk)+pscr(jk)*scr1(jk))
1936    70      gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr1(jk)
1937            return
1938        end
1939            subroutine eosp1 (rrr,eee,n,gamma,ppp)
1940 c-------------------------------------------------------------
1941 c     equation of state routine for air.
1942 c
1943 c     input varible definitions.
1944 c           rrr = mass density
1945 c           eee = internal energy per unit volume
1946 c                 (converted for internal use to energy per unit mass)
1947 c             n = number of entries in arrays rrr & eee
1948 c
1949 c     this routine calculates gamma and ppp as follows:
1950 c         gamma = 1.0 + p/(rho*e)
1951 c         ppp   = pressure(units of eee)
1952 c-------------------------------------------------------------
1953            parameter (m=64)
1954 c
1955        dimension rrr(n), eee(n), gamma(n), ppp(n)
1956        dimension t11(m), t12(m), t21(m), t22(m), rho(m), e(m), int(m)
1957        dimension p(m), omp(m), q(m), i(m), j(m), tem(m)
1958        dimension g(8,105), jcy(m), js(m)
1959        dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
1960        dimension g7(120), gf(840)
1961 c
1962        equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
1963        equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
1964        equivalence (g7(1),g(1,91)), (int(1),tem(1)), (gf(1),g(1,1))
1965 c
1966            data rzero1 /774.413/
1967        data x116e /2.772588722239774483568908181041479110717734375/
1968        data r116e /0.360673760222240957773465197533369064331054687 5/
1969 c-------------------------------------------------------------
1970 c     g = gamma - 1.0 is stored for 32 bit word machines in powers of
1971 c     16 across for mass density variation and intermediate values
1972 c     1 - 16 for powers of 16 vertically which represent the internal
1973 c     energy variation.
1974 c
1975 c     16**(2) .ge. rho .ge. 16**(-6)
1976 c     16**(8) .le. e   .le. 16**(15)
1977 c ------------------ ------- -------- ---------- -------
1978        data g1 /8*.4222,8*.4152,8*.4110,8*.4081,8*.4058,8*.4040,8*.4024,8
1979       1 *.4011,8*.3998,8*.3988,8*.3978,8*.3969,8*.3961,8*.3953,8*.3935/
1980        data g2 /.3918,.3918,.3918,.3918,.3918,.3918,.3918,.3723,.37
1981       1 15,.3707,.3699,.3690,.3680,.3663,.3637,.3555,.3538,.3522,.3502,.3
1982       2 476,.3430,.3344,.3238,.3370,.3370,.3370,.3364,.3347,.3277,.3099,.
1983       3 2885,.3257,.3227,.3201,.3134,.3062,.3014,.2884,.2591,.3166,.3110,
1984       4 .3063,.2946,.2831,.2787,.2677,.2358,.3111,.3006,.2940,.2787,.2635
```

```
1985      5 ,.2588,.2502,.2236,.3075,.2906,.2810,.2665,.2466,.2418,.2350,.213
1986      6 1,.3043,.2819,.2695,.2554,.2317,.2269,.2216,.2038,.2929,.2740,.25
1987      7 93,.2455,.2206,.2136,.2097,.1955,.2840,.2672,.2500,.2366,.2166,.2
1988      8 015,.1988,.1879,.2764,.2611,.2429,.2285,.2125,.1890,.1890,.1811,.
1989      9 2714,.2555,.2384,.2210,.2079,.1818,.1799,.1747,.2669,.2504,.2343,
1990      $ .2141,.2037,.1822,.1709,.1689,.2624,.2473,.2304,.2096,.1998,.1828
1991      $ ..1684,.1639/
1992       data g3 /.2599,.2446,.2268,.2087,.1961,.1834,.1673,.1601,.2401,.21
1993      1 91,.1972,.1775,.1592,.1444,.1358,.1203,.2002,.1960,.1749,.1536,.1
1994      2 376,.1252,.1107,.1044,.1911,.1829,.1633,.1420,.1266,.1101,.1012,.
1995      3 0933,.1950,.1781,.1566,.1415,.1241,.1118,.1009,.0948,.2001,.1789,
1996      4 .1594,.1443,.1306,.1189,.1095,.1013,.2040,.1826,.1657,.1494,.1338
1997      5 ,.1177,.1081,.0980,.2034,.1854,.1683,.1497,.1322,.1169,.1051,.094
1998      6 6,.1969,.1855,.1685,.1487,.1304,.1149,.1024,.0916,.1899,.1837,.16
1999      7 77,.1475,.1287,.1126,.1002,.0900,.1841,.1817,.1667,.1464,.1272,.1
2000      8 109,.0983,.0888,.1800,.1800,.1659,.1455,.1262,.1097,.0965,.0878,.
2001      9 1779,.1787,.1657,.1450,.1254,.1087,.0949,.0868,.1773,.1778,.1656,
2002      $ .1447,.1250,.1080,.0939,.0859,.1783,.1778,.1658,.1448,.1248,.1076
2003      $ ,.0933,.0851/
2004       data g4 /.1808,.1781,.1667,.1451,.1248,.1074,.0930,.0843,.2134,.20
2005      1 40,.1978,.1782,.1565,.1368,.1206,.1074,.2210,.2072,.1957,.1739,.1
2006      2 516,.1312,.1137,.1000,.2245,.2109,.1989,.1772,.1563,.1390,.1247,.
2007      3 1133,.2299,.2132,.2017,.1795,.1579,.1384,.1221,.1090,.2350,.2157,
2008      4 .2023,.1798,.1575,.1370,.1197,.1057,.2397,.2194,.2034,.1796,.1572
2009      5 ,.1372,.1205,.1070,.2452,.2227,.2050,.1805,.1576,.1379,.1236,.111
2010      6 8,.2510,.2256,.2069,.1814,.1581,.1383,.1231,.1103,.2560,.2282,.20
2011      7 91,.1822,.1585,.1385,.1226,.1083,.2605,.2312,.2111,.1829,.1588,.1
2012      8 386,.1222,.1070,.2677,.2358,.2129,.1836,.1592,.1386,.1218,.1071,.
2013      9 2759,.2403,.2145,.1857,.1598,.1389,.1219,.1078,.2834,.2445,.2160,
2014      $ .1878,.1603,.1394,.1223,.1084,.2905,.2484,.2175,.1898,.1613,.1399
2015      $ ..1226,.1090/
2016       data g5 /.2963,.2531,.2199,.1918,.1625,.1407,.1230,.1096,.4323,.35
2017      1 82,.3109,.2889,.2803,.2706,.2410,.2224,.4610,.4026,.3624,.3212,.2
2018      2 926,.2551,.2375,.2015,.4199,.3837,.3401,.2979,.2623,.2318,.2108,.
2019      3 1854,.3924,.3642,.3194,.2760,.2427,.2157,.1902,.1721,.3794,.3479,
2020      4 .3025,.2673,.2311,.2019,.1842,.1613,.3674,.3448,.2961,.2593,.2255
2021      5 ,.1994,.1785,.1594,.3573,.3443,.2910,.2517,.2293,.2006,.1843,.167
2022      6 9,.3661,.3438,.2935,.2597,.2336,.2225,.2143,.2116,.3674,.3435,.30
2023      7 80,.2728,.2606,.2577,.2573,.2573,.3685,.3453,.3210,.3014,.2942,.2
2024      8 933,.2932,.2932,.3814,.3612,.3341,.3276,.3257,.3253,.3252,.3252,.
2025      9 3903,.3752,.3570,.3522,.3513,.3510,.3506,.3496,.4012,.3899,.3782,
2026      $ .3751,.3743,.3741,.3734,.3713,.4155,.4057,.3956,.3930,.3920,.3913
2027      $ ,.3907,.3890/
2028       data g6 /.4290,.4205,.4118,.4092,.4077,.4065,.4059,.4047,.5411,.53
2029      1 85,.5359,.5353,.5351,.5350,.5350,.5350,.5823,.5812,.5801,.5797,.5
2030      2 796,.5797,.5797,.5797,.6096,.6090,.6085,.6082,.6082,.6083,.6083,.
2031      3 6083,.6308,.6306,.6305,.6303,.6303,.6305,.6305,.6305,.6481,.6483,
2032      4 .6485,.6483,.6484,.6486,.6487,.6487,.6627,.6632,.6637,.6636,.6637
2033      5 ,.6640,.6640,.6640,.6754,.6761,.6769,.6768,.6770,.6773,.6773,.677
2034      6 3,.6866,.6875,.6885,.6884,.6886,.6890,.6890,.6890,.6966,.6977,.69
2035      7 89,.6989,.6991,.6995,.6995,.6995,.7056,.7070,.7083,.7083,.7085,.7
2036      8 090,.7090,.7090,.7139,.7154,.7169,.7169,.7172,.7176,.7177,.7177,.
2037      9 7214,.7231,.7248,.7248,.7251,.7256,.7256,.7256,.7285,.7303,.7321,
2038      $ .7321,.7325,.7330,.7330,.7330,.7350,.7370,.7390,.7390,.7393,.7398
2039      $ ,.7399,.7399/
2040       data g7 /.7411,.7432,.7453,.7454,.7457,.7463,.7463,.7463,.8069,.81
2041      1 03,.8138,.8139,.8145,.8152,.8153,.8153,.8454,.8496,.8538,.8540,.8
2042      2 547,.8556,.8557,.8557,.8727,.8774,.8822,.8825,.8832,.8842,.8843,.
2043      3 8843,.8938,.8990,.9042,.9046,.9054,.9064,.9065,.9065,.9111,.9166,
2044      4 .9222,.9226,.9235,.9246,.9247,.9247,.9258,.9316,.9374,.9379,.9387
2045      5 ,.9399,.9400,.9400,.9384,.9445,.9506,.9511,.9520,.9532,.9533,.953
2046      6 3,.9496,.9559,.9622,.9627,.9637,.9649,.9650,.9650,.9596,.9661,.97
2047      7 27,.9731,.9741,.9754,.9755,.9755,.9686,.9753,.9821,.9826,.9836,.9
2048      8 849,.9850,.9850,.9769,.9837,.9906,.9912,.9922,.9936,.9917,.9937,.
```

```
     9 9845,.9915,.9986,.9991,4*.9999,.9915,.9987,6*.9999,.9981,7*.9999/
c-----------------------------------------------------------------
c     real air eos, table lookup on gilmore data.  (no temp. model)
c     to avoid costly logrithmic functions the table "g" is stored in a
c     form so that the hexadecimal word structure of a 32 bit machine
c     may be exploited.
c     this logic may be transfered to other machines by recalculating
c     the table "g" appropriate to the word arcitecture of that machine.
c     machine dependent functions and key numbers must also be changed.
c-----------------------------------------------------------------
      ist=0
      nr=n
   10 nst=min0(nr,m)
c
      do 20 ire=1,nst
      rho(ire)=rzeroi*rrr(ist+ire)
      e(ire)=amax1(3.e8,eee(ist+ire)/rrr(ist+ire))
c
c     calculate mass density variation index "i".
      tem(ire)=alog(rho(ire))*rl16e+500.0
      i(ire)=tem(ire)
      omp(ire)=tem(ire)-i(ire)
      i(ire)=502-i(ire)
      p(ire)=1.0-omp(ire)
      i(ire)=max0(i(ire),1)
c
c     calculate internal energy variation index "j".
      tem(ire)=alog(e(ire))*rl16e
      jcy(ire)=ifix(tem(ire))
      tem(ire)=tem(ire)-jcy(ire)
      tem(ire)=exp(xl16e*tem(ire))
      jcy(ire)=jcy(ire)-7
      js(ire)=tem(ire)
      q(ire)=tem(ire)-js(ire)
      j(ire)=js(ire)+15*jcy(ire)
      j(ire)=min0(j(ire),104)
      j(ire)=i(ire)*8+j(ire)
   20 i(ire)=j(ire)-8
      do 30 ire=1,nst
      t11(ire)=gf(i(ire))
      t21(ire)=gf(i(ire)+1)
      t12(ire)=gf(j(ire))
   30 t22(ire)=gf(j(ire)+1)
c
c     calculate gamma by linear interpolation.
      do 40 ire=1,nst
      t12(ire)=t12(ire)-t11(ire)
      t22(ire)=t22(ire)-t21(ire)
   40 gamma(ist+ire)=1.0+omp(ire)*(t11(ire)+q(ire)*t12(ire))+p(ire)*
     1 (t21(ire)+q(ire)*t22(ire))
c
      nr=nr-nst
      ist=ist+nst
      if (nr.gt.0) go to 10
c
c     calculate presure in units of ees
      do 50 ire=1,n
   50 ppp(ire)=amax1(0.0,eee(ire)*(gamma(ire)-1.0))
c
      return
      end
      subroutine eosp2 (rrr,eee,n,gamma,ppp)
c-----------------------------------------------------------------
c     equation of state for dust.
```

```
2113 c-----------------------------------------------------------------------
2114           real rrr(n), eee(n), gamma(n), ppp(n)
2115 c
2116           do 10 i=1,n
2117           gamma(i)=1.0
2118    10     ppp(i)=0.0
2119           return
2120       end
2121           subroutine radlos (t,xcor,ycor,radl,fracn)
2122 c-----------------------------------------------
2123 c     subroutines radlos and fbloss perform calculations to account
2124 c     for the radiative energy loss from the fireball to the
2125 c     surroundings.
2126 c
2127 c     the calculations involve the following steps:
2128 c     1. call subroutine fbloss to compute the radiative energy loss and
2129 c     2. subtract this energy proportionately from the entire grid.
2130 c
2131 c     arguments:
2132 c      t    = time,
2133 c      dt   = time step,
2134 c      xcor = x coordinates of grid points,
2135 c      ycor = y coordinates of grid points,
2136 c      rho  = fluid density,
2137 c      rvx  = x momentum,
2138 c      rvy  = y momentum,
2139 c      erg  = energy per unit volume and
2140 c      radl = cumulative radiation energy loss.
2141 c      fracn= fraction of yield deposited in grid.
2142 c-----------------------------------------------------------------------
2143           parameter (mnx=150, mny=200, mxy=202)
2144           parameter (mnxx=1, mnyy=1)
2145           integer alpha
2146           logical lmgrid, lstat, lnuc, therml, lgrav, leos
2147           real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
2148           real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
2149           real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
2150           common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
2151           common lstat, lnuc, therml, lgrav, leos
2152           common alpha, gamma, gm1, dt, dx, dy, rhomin
2153           common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
2154           real xcor(mxy), ycor(mxy)
2155 c
2156           real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
2157           real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
2158           real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
2159           real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
2160           real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
2161           real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
2162           real rmin(mxy), rmax(mxy), rvr(mxy)
2163           common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
2164      1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
2165      2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
2166 c
2167           real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
2168           parameter (jthm=1)
2169           real fsky(mxy), rhtl(mnx,jthm)
2170           common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
2171      1    , jtherm
2172           common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
2173      1    jbt, iadd, jadd, factor
2174 c
2175 c     compute the energy radiated by the fireball in dt seconds.
2176           call fblos1 (t,dt,f)
```

```
2177            f=f*fracn
2178            radl=radl+f
2179            if (f.eq.0.0) return
2180 c
2181 c     subtract this energy proportionately from the entire grid.
2182            tsum=0.0
2183            do 20 j=jyb,jyt
2184            do 10 i=ixl,ixr
2185            rhom(i)=1.0/rho(i,j)
2186            ergk(i)=rhom(i)*amax1(0.0,erg(i,j)-0.5*rhom(i)*(rvx(i,j)*rvx(i
2187       1    ,j)+rvy(i,j)*rvy(i,j))-rho(i,j)*gp00(j+1))
2188      10    ergk(i)=ergk(i)*ergk(i)*ergk(i)*ergk(i)
2189            tsum=tsum+ssum(ixr-ixl+1,ergk,1)
2190      20    continue
2191            fact=f/tsum
2192 c
2193            do 30 j=jyb,jyt
2194            do 30 i=ixl,ixr
2195            uh(i)=(xcor(i+1)-xcor(i))*(ycor(j+1)-ycor(j))*3.1415926*(xcor
2196       1    (i)+xcor(i+1))
2197            rhom(i)=1.0/rho(i,j)
2198            ergk(i)=rhom(i)*amax1(0.0,erg(i,j)-0.5*rhom(i)*(rvx(i,j)*rvx(i
2199       1    ,j)+rvy(i,j)*rvy(i,j))-rho(i,j)*gp00(j+1))
2200            erg(i,j)=cvmgt(erg(i,j),erg(i,j)-fact*ergk(i)*ergk(i)*ergk(i)
2201       1    *ergk(i)/uh(i),erg(i,j).lt.1.1*er00(j+1))
2202      30    continue
2203 c
2204            return
2205       end
2206            subroutine fbloss (t,dt,f,w,sigma)
2207 c-----------------------------------------------------
2208 c     fireball loss routine scaled to sputter data by al sharp.
2209 c                11july72
2210 c
2211 c     t      time in seconds of last hydro cycle
2212 c     dt     hydro time step, tt is the subroutine time after burst.
2213 c     f      energy %ergs< radiated at time t in dt seconds.
2214 c     w      device yield in kt.
2215 c     sigma ratio of density at burst height to density at sea level
2216 c-----------------------------------------------------
2217 c     this entry point is for initialization
2218 c
2219            real m
2220 c
2221            if (w.eq.0.0) return
2222            rhoa=1.225e-03*sigma
2223            m=1.31737*rhoa**(-0.0432)
2224            con=0.17e14*sigma**(-0.6244)
2225            psem=con*w**(0.53)
2226            tsem=(1.6828e-04-7.62326e-06*alog(sigma)-3.72132e-06*alog
2227       1    (sigma)**2-1.51933e-07*alog(sigma)**3)*w**(0.3123)
2228            bmr=2.0*sigma**0.1008
2229            bml=sigma**0.1189
2230            tmax=0.038*w**(0.44)*sigma**(0.36)
2231            tmin=0.00256*w**(0.39)*sigma**(-0.062)
2232            tmin=amin1(0.99*tmax,tmin)
2233            pmax=1.49e13*w**(0.59)*sigma**( 0.45)
2234            pmin=6.82e11*w**(0.54)*sigma**(-1.02)
2235            pmin=amin1(0.99*pmax,pmin)
2236            bsem=psem/pmin-1.0
2237            bnr=11.2586*rhoa**(0.17827)
2238            bnl=10.0
2239            if (rhoa.lt.4.0e-06) bnl=5971.13*rhoa**(0.51428)
2240            btime=0.91981*tmax**rhoa**(-0.072126)
```

```
2241           bhite=1.0/(0.8*sigma**(0.3372)) 1.0
2242           tr=tmin/tmax
2243           c=(alog(tmax)-alog(tmin))*tmax/(tmax-tmin)
2244           b=m*exp(c)/c
2245           hintmn=pmin/(tr**(-m)*exp(-b*exp(-c*tr)))
2246           hintmx=pmax/(exp(-b*exp(-c)))
2247           return
2248 c
2249           entry fblosi(t,dt,f)
2250 c
2251 c     this entry point is called at the start of each hydro cycle.
2252 c
2253           if (w.eq.0.0) go to 60
2254           tt=t+0.5*dt
2255           if (tt.lt.1.0e-07) go to 50
2256           tsmax=tt/tmax
2257           if (tt.lt.tmin) go to 20
2258           if (tsmax-1.0) 10,30,30
2259     10    betah=1.35
2260           alph=3.5
2261           teta=1.0-alog(1.0/tsmax)/alog(1.0/tr)
2262           tetab=3.1415926*teta**betah
2263           hint=hintmx+(hintmn-hintmx)*((1.0+cos(tetab))/2.0)**alph
2264           go to 40
2265     20    fac=2.0*bhite/((btime/tt)**bnl+(tt/btime)**bnr)
2266           bfac=1.0/(fac+1.0)
2267           facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
2268 c
2269 c    this statement converts power out to energy radiated
2270 c
2271                f=pmin*bfac*facm*1.0e07*dt
2272           return
2273     30    hint=hintmx
2274     40    fac=2.0*bhite/((btime/tt)*bnl+(tt/btime)**bnr)
2275           bfac=1.0/(fac+1.0)
2276           facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
2277           pwt=hint*(tsmax**(-m)*exp(-b*exp(-c*tsmax)))*bfac*facm
2278                f=pwt*1.0e07*dt
2279           return
2280     50    if (tt.lt.1.0e-08) go to 60
2281           facm=1.0+(2.0*bsem/((tsem/tt)**bml+(tt/tsem)**bmr))
2282                f=facm*10.0**((alog10(tt)+8.0)*(alog10(pmin)-1.0)+1.0)
2283           return
2284     60         f=0.0
2285           return
2286        end
2287           subroutine staton (k,nstep,time)
2288 c-------------------------------------------------------------
2289 c     calculates pressures and dynamic pressures at eulerian stations.
2290 c-------------------------------------------------------------
2291           parameter (mnx=150, mny=200, mxy=202)
2292           parameter (mnxx=1, mnyy=1)
2293           parameter (mxx=1, nsta=1)
2294 c
2295           integer alpha
2296           logical lmgrid, lstat, lnuc, therml, lgrav, leos
2297           real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
2298           real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
2299           real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
2300           common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
2301           common lstat, lnuc, therml, lgrav, leos
2302           common alpha, gamma, gm1, dt, dx, dy, rhomin
2303           common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
2304 c
```

```
2305            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
2306            common /grids/ xcor, xcoro, ycor, ycoro, unit
2307 c
2308            common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
2309           1    jbt, iadd, jadd, factor
2310 c
2311            real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
2312            parameter (jthm=1)
2313            real fsky(mxy), rhtl(mnx,jthm)
2314            common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
2315           1    , jtherm
2316 c
2317            real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
2318            real prs(mxx,nsta), xs(nsta), ys(nsta)
2319            common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
2320           1    ixx
2321 c
2322            ixx=ixx+1
2323            tme(ixx)=time
2324            imin=cvmgt(ixl,1,nstep.ne.1)
2325            imax=cvmgt(ixr,mnx,nstep.ne.1)
2326            jmin=cvmgt(jyb,1,nstep.ne.1)
2327            jmax=cvmgt(jyt,mny,nstep.ne.1)
2328            do 200 j=1,nsta
2329            if (xs(j).gt.xcor(imax).or.xs(j).lt.xcor(imin)) go to 200
2330            if (ys(j).gt.ycor(jmax).or.ys(j).lt.ycor(jmin)) go to 200
2331            do 190 i=imin,imax
2332            if (xs(j).le.xcor(i).or.xs(j).gt.xcor(i+1)) go to 190
2333 c-----------------------------------------------------------------
2334 c     now inside the column, must find the right row
2335 c     must also interpolate the y direction
2336 c     find the index for the y coordinate of the station
2337 c-----------------------------------------------------------------
2338            do 10 js=jmin,jmax
2339       10   if (ys(j).ge.ycor(js).and.ys(j).lt.ycor(js+1)) l=js
2340            yfrac=(ys(j)-ycor(l))/(ycor(l+1)-ycor(l))
2341            yfrac=yfrac-0.5
2342            if (yfrac) 20,30,40
2343       20   m=l
2344            l=l-1
2345            if (m.eq.1) l=1
2346            go to 50
2347       30   m=l
2348            go to 50
2349       40   m=l+1
2350       50   continue
2351            fay=(ys(j)-ycor(l))/(ycor(2)-ycor(1))
2352            if (m.eq.1) go to 60
2353            ya=(ycor(m+1)+ycor(m))*0.5
2354            yb=(ycor(m-1)+ycor(m))*0.5
2355            fay=(ys(j)-yb)/(ya-yb)
2356 c
2357 c     must get the x fraction of the cell
2358       60   xfrac=(xs(j)-xcor(i))/(xcor(i+1)-xcor(i))
2359            xfrac=xfrac-0.5
2360            if (xfrac) 70,110,150
2361 c
2362 c     lower part of the cell
2363       70   if (i.eq.1) go to 110
2364            xl=(xcor(i)+xcor(i-1))*0.5
2365            xr=(xcor(i)+xcor(i+1))*0.5
2366            fac=(xs(j)-xl)/(xr-xl)
2367            p1=erg(i-1,l)+(erg(i,l)-erg(i-1,l))*fac
2368            p2=erg(i-1,m)+(erg(i,m)-erg(i-1,m))*fac
```

```
2369            ergs=p1+(p2-p1)*fay
2370            r1=rho(i-1,l)+(rho(i,l)-rho(i-1,l))*fac
2371            r2=rho(i-1,m)+(rho(i,m)-rho(i-1,m))*fac
2372            rhos=r1+(r2-r1)*fay
2373            rhas=0.0
2374            rhbs=0.0
2375            rhcs=rhos
2376            go to (100,90,80), nspec
2377     80     r1=rhb(i-1,l)+(rhb(i,l)-rhb(i-1,l))*fac
2378            r2=rhb(i-1,m)+(rhb(i,m)-rhb(i-1,m))*fac
2379            rhbs=r1+(r2-r1)*fay
2380            rhcs=rhcs-rhbs
2381     90     r1=rha(i-1,l)+(rha(i,l)-rha(i-1,l))*fac
2382            r2=rha(i-1,m)+(rha(i,m)-rha(i-1,m))*fac
2383            rhas=r1+(r2-r1)*fay
2384            rhcs=amax1((rhcs-rhas),0.0)
2385    100     continue
2386            u1=rvx(i-1,l)+(rvx(i,l)-rvx(i-1,l))*fac
2387            u2=rvx(i-1,m)+(rvx(i,m)-rvx(i-1,m))*fac
2388            rvxs=u1+(u2-u1)*fay
2389            v1=rvy(i-1,l)+(rvy(i,l)-rvy(i-1,l))*fac
2390            v2=rvy(i-1,m)+(rvy(i,m)-rvy(i-1,m))*fac
2391            rvys=v1+(v2-v1)*fay
2392            go to 190
2393 c-------------------------------------------------------
2394 c     here we are at cell center, so no interpolation is needed
2395 c     in the x direction, but alas must do the y
2396 c-------------------------------------------------------
2397    110     p1=erg(i,l)
2398            p2=erg(i,m)
2399            ergs=p1+(p2-p1)*fay
2400            r1=rho(i,l)
2401            r2=rho(i,m)
2402            rhos=r1+(r2-r1)*fay
2403            rhas=0.0
2404            rhbs=0.0
2405            rhcs=rhos
2406            go to (140,130,120), nspec
2407    120     r1=rhb(i,l)
2408            r2=rhb(i,m)
2409            rhbs=r1+(r2-r1)*fay
2410            rhcs=rhcs-rhbs
2411    130     r1=rha(i,l)
2412            r2=rha(i,m)
2413            rhas=r1+(r2-r1)*fay
2414            rhcs=amax1((rhcs-rhas),0.0)
2415    140     continue
2416            u1=rvx(i,l)
2417            u2=rvx(i,m)
2418            rvxs=u1+(u2-u1)*fay
2419            v1=rvy(i,l)
2420            v2=rvy(i,m)
2421            rvys=v1+(v2-v1)*fay
2422            go to 190
2423 c
2424 c          the upper part of the cell in the x-direction.
2425    150     xl=(xcor(i)+xcor(i+1))*0.5
2426            xr=(xcor(i+1)+xcor(i+2))*0.5
2427            fac=(xs(j)-xl)/(xr-xl)
2428            p1=erg(i,l)+(erg(i+1,l)-erg(i,l))*fac
2429            p2=erg(i,m)+(erg(i+1,m)-erg(i,m))*fac
2430            ergs=p1+(p2-p1)*fay
2431            r1=rho(i,l)+(rho(i+1,l)-rho(i,l))*fac
2432            r2=rho(i,m)+(rho(i+1,m)-rho(i,m))*fac
```

```
2433            rhos=r1+(r2-r1)*fay
2434            rhas=0.0
2435            rhbs=0.0
2436            rhcs=rhos
2437            go to (180,170,160), nspec
2438    160     r1=rhb(i,l)+(rhb(i+1,l)-rhb(i,l))*fac
2439            r2=rhb(i,m)+(rhb(i+1,m)-rhb(i,m))*fac
2440            rhbs=r1+(r2-r1)*fay
2441            rhcs=rhcs-rhbs
2442    170     r1=rha(i,l)+(rha(i+1,l)-rha(i,l))*fac
2443            r2=rha(i,m)+(rha(i+1,m)-rha(i,m))*fac
2444            rhas=r1+(r2-r1)*fay
2445            rhcs=amax1((rhcs-rhas),0.0)
2446    180     continue
2447            u1=rvx(i,l)+(rvx(i+1,l)-rvx(i,l))*fac
2448            u2=rvx(i,m)+(rvx(i+1,m)-rvx(i,m))*fac
2449            rvxs=u1+(u2-u1)*fay
2450            v1=rvy(i,l)+(rvy(i+1,l)-rvy(i,l))*fac
2451            v2=rvy(i,m)+(rvy(i+1,m)-rvy(i,m))*fac
2452            rvys=v1+(v2-v1)*fay
2453    190     continue
2454            rhs(k,j)=rhos
2455            rhom=1.0/rhos
2456            vls(k,j)=0.5*(rvxs*rvxs+rvys*rvys)*rhom
2457            eint=amax1(0.0,ergs-vls(k,j))
2458            if (leos) then
2459 c           call eos (1,nspec,rhos,eint,gms(k,j),prs(k,j),rscr,pscr,gscr
2460 c     1     ,scr1,scr2,rhas,rhbs)
2461            call eospl (rhos,eint,1,gms(k,j),prs(k,j))
2462            else
2463            prs(k,j)=gm1*eint
2464            gms(k,j)=gamma
2465            endif
2466            vls(k,j)=sqrt(2.0*vls(k,j)*rhom)
2467    200     continue
2468            return
2469        end
2470            subroutine blufxy (bcl,bcr,bcb,bct)
2471 c-----------------------------------------------------------------------
2472 c     hydro advance for grid containing bluff bodies.
2473 c-----------------------------------------------------------------------
2474            parameter (mnx=150, mny=200, mxy=202)
2475            parameter (nnbd=2, nbp1=nnbd+1)
2476 c
2477            logical lscan, rigt, ltop, left, botm
2478            common /scan/ lscan, rigt, left, ltop, botm
2479            common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
2480      1     jbt, iadd, jadd, factor
2481 c
2482            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
2483            common /grids/ xcor, xcoro, ycor, ycoro, unit
2484 c
2485            real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
2486            real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
2487            real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
2488            real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
2489            real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
2490            real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
2491            real rmin(mxy), rmax(mxy), rvr(mxy)
2492            common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
2493      1     rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
2494      2     , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
2495 c
2496            integer nlef(mny,nbp1), nrig(mny,nbp1), imn(nnbd), imx(nnbd)
```

```
2497            integer nbot(mnx,nbp1), ntop(mnx,nbp1), jmn(nnbd), jmx(nnbd)
2498            common /bodint/ nlef, nrig, imn, imx, nbot, ntop, jmn, jmx
2499 c
2500 c       when active grid does not include bodies.
2501            if (ixr.lt.imin-1.or.jyt.lt.jmin-1.or.ixl.gt.imax+1.or.jyb.gt
2502           1     .jmax+1) then
2503            call bndrx (bcr,bcl)
2504            call integx
2505            call bndry (bct,bcb)
2506            call integy
2507            return
2508            endif
2509 c
2510 c       adjust active grid boundaries if necessary.
2511            if (iscan) then
2512            ixl=cvmgt(max0(imin-iadd,1),ixl,imin.le.ixl.and.imax.ge.ixl)
2513            ixr=cvmgt(min0(imax+iadd,mnx),ixr,imin.le.ixr.and.imax.ge.ixr)
2514            jyb=cvmgt(max0(jmin-jadd,1),jyb,jmin.le.jyb.and.jmax.ge.jyb)
2515            jyt=cvmgt(min0(jmax+jadd,mny),jyt,jmin.le.jyt.and.jmax.ge.jyt)
2516 c
2517 c       reset boundary conditions.
2518            abcl=cvmgt(bcl,0.0,ixl.eq.1)
2519            abcr=cvmgt(bcr,0.0,ixr.eq.mnx)
2520            abcb=cvmgt(bcb,0.0,jyb.eq.1)
2521            abct=cvmgt(bct,0.0,jyt.eq.mny)
2522            endif
2523 c
2524 c       save grid boundaries.
2525            isavl=ixl
2526            isavr=ixr
2527            jsavb=jyb
2528            jsavt=jyt
2529 c
2530 c       row integration.
2531 c       ****************
2532 c       region below bodies.
2533            if (jmin.gt.jsavb) then
2534            jyt=jmin-1
2535            jbt=jyt-jyb+1
2536            call bndrx (abcr,abcl)
2537            call integx
2538            endif
2539 c
2540 c       region including the bodies.
2541            do 10 j=jmin,jmax
2542            jyb=j
2543            jyt=j
2544            jbt=1
2545            do 10 n=1,nbodp1
2546            if (nlef(j,n).ne.0) then
2547            ixl=nlef(j,n)
2548            ixr=nrig(j,n)
2549            ilr=ixr-ixl+1
2550            aabcl=cvmgt(abcl,-1.0,ixl.eq.isavl)
2551            aabcr=cvmgt(abcr,-1.0,ixr.eq.isavr)
2552            call bndrx (aabcr,aabcl)
2553            call integx
2554            endif
2555       10   continue
2556 c
2557 c       region above bodies.
2558            if (jmax.lt.jsavt) then
2559            ixl=isavl
2560            ixr=isavr
```

```
2561            ilr=ixr-ixl+1
2562            jyb=jmax+1
2563            jyt=jsavt
2564            jbt=jyt-jyb+1
2565            call bndrx (abcr,abcl)
2566            call integx
2567            endif
2568 c
2569 c       column integration.
2570 c       ******************
2571 c       region left of bodies.
2572            if (imin.gt.isavl) then
2573            ixl=isavl
2574            ixr=imin-1
2575            ilr=ixr-ixl+1
2576            jyb=jsavb
2577            jyt=jsavt
2578            jbt=jyt-jyb+1
2579            call bndry (abct,abcb)
2580            call integy
2581            endif
2582 c
2583 c       region including the bodies.
2584            do 20 i=imin,imax
2585            ixl=i
2586            ixr=i
2587            ilr=1
2588            do 20 n=1,nbodpt
2589            if (nbot(i,n).ne.0) then
2590            jyb=nbot(i,n)
2591            jyt=ntop(i,n)
2592            jbt=jyt-jyb+1
2593            aabcb=cvmgt(abcb,-1.0,jyb.eq.jsavb)
2594            aabcb=cvmgt(2.0,aabcb,i.lt.imn(i))
2595            aabct=cvmgt(abct,-1.0,jyt.eq.jsavt)
2596            call bndry (aabct,aabcb)
2597            call integy
2598            endif
2599       20   continue
2600 c
2601 c       region to right of bodies.
2602            if (imax.lt.isavr) then
2603            jyb=jsavb
2604            jyt=jsavt
2605            jbt=jyt-jyb+1
2606            ixl=imax+1
2607            ixr=isavr
2608            ilr=ixr-ixl+1
2609            call bndry (abct,abcb)
2610            call integy
2611            endif
2612 c
2613 c       reset active grid boundaries.
2614            ixr=isavr
2615            ixl=isavl
2616            jyb=jsavb
2617            jyt=jsavt
2618            ilr=ixr-ixl+1
2619            jbt=jyt-jyb+1
2620            return
2621 c
2622            entry bluf(nbod)
2623 c-----------------------------------------------------------------
2624 c       initialize bluff body parameters.
```

```
c-----------------------------------------------------------------------
          nbodp1=nbod+1
          imin=1
          imax=mnx
          jmin=1
          jmax=mny
c
c     body boundaries.
          imn(1)=18
          imx(1)=24
          jmn(1)=1
          jmx(1)=105
          imn(2)=18
          imx(2)=52
          jmn(2)=158
          jmx(2)=164
c
c     initialize body integers to zero.
          do 50 n=1,nbodp1
          do 30 j=1,mny
          nlef(j,n)=0
   30     nrig(j,n)=0
          do 40 i=1,mnx
          nbot(i,n)=0
   40     ntop(i,n)=0
   50     continue
c
c     body integers.
c     left, segment 1.
          do 60 j=1,mny
   60     nlef(j,1)=1
c
c     left and right, segment 2.
          do 70 n=1,nbod
          do 70 j=jmn(n),jmx(n)
          nlef(j,2)=imx(n)+1
   70     nrig(j,2)=mnx
c
c     right, segment 1.
          do 80 n=1,nbod
          do 80 j=jmn(n),jmx(n)
   80     nrig(j,1)=imn(n)-1
          do 90 j=jmx(1)+1,jmn(2)-1
   90     nrig(j,1)=mnx
          do 100 j=jmx(2)+1,mny
  100     nrig(j,1)=mnx
c
c     bottom, segment 1.
          do 110 i=1,imn(1)-1
  110     nbot(i,1)=1
          do 120 i=imn(1),imx(1)
  120     nbot(i,1)=jmx(1)+1
          do 130 i=imx(1)+1,mnx
  130     nbot(i,1)=1
c
c     bottom and top, segment 2.
          do 140 i=imn(2),imx(2)
          nbot(i,2)=jmx(2)+1
  140     ntop(i,2)=mny
c
c     top, segment 1.
          do 150 i=1,imn(1)-1
  150     ntop(i,1)=mny
          do 160 i=imn(2),imx(2)
```

```
      160     ntop(1,1)=jmn(2)-1
              do 170 i=imx(2)+1,mnx
      170     ntop(i,1)=mny
              return
          end
              subroutine dpshok (rho,rvx,rvy,erg)
c-----------------------------------------------------------------------
c     deposits shock in shock tube.
c-----------------------------------------------------------------------
              parameter (mnx=150, mny=200, mxy=202)
              parameter (nnbd=2, nbp1=nnbd+1)
              real rho(mnx,mny), erg(mnx,mny)
              real rvx(mnx,mny), rvy(mnx,mny), mach
c
              integer nlef(mny,nbp1), nrig(mny,nbp1), jmn(nnbd), jmx(nnbd)
              integer nbot(mnx,nbp1), ntop(mnx,nbp1), jmn(nnbd), jmx(nnbd)
              common /bodint/ nlef, nrig, imn, imx, nbot, ntop, jmn, jmx
              data p1, p2, r1 /1.013e6,3.42e6,1.19e-3/
              logical leos
              common /bbc/ rhobbc, rvxbbc, rvybbc, ergbbc
c
c     set up shock.
              do 10 j=1,jmx(1)
              do 10 i=1,imn(1)-1
              rho(i,j)=rhobbc
              rvx(i,j)=rvxbbc
              rvy(i,j)=rvybbc
      10      erg(i,j)=ergbbc
              return
c
              entry jump(gamma,leos,ergamb)
c-----------------------------------------------------------------------
c     calculate conditions behind shock.
c-----------------------------------------------------------------------
c     compute internal energy, mach number and density.
              p2op1=p2/p1
              ergbbc=p2/(gamma-1.0)
              mach=sqrt(((gamma+1.0)*p2op1+(gamma-1.0))/(2.0*gamma))
              rhobbc=r1*(gamma+1.0)*mach*mach/((gamma-1.0)*mach*mach+2.0)
c
c     iterate to get better gamma and energy.
              gam=gamma
              pre=p1
              if (.not.leos) go to 30
              kount=0
      20      if (kount.eq.50) then
              write (6,40)
              stop
              endif
              call eosp1 (rhobbc,ergbbc,1,gam,pre)
              kount=kount+1
              ergbbc=p2/(gam-1.0)
              mach=sqrt(((gam+1.0)*p2op1+(gam-1.0))/(2.0*gam))
              rhobbc=r1*(gam+1.0)*mach*mach/((gam-1.0)*mach*mach+2.0)
              if (abs(pre-p2).gt.1.e-6*p2) go to 20
c
c     determine ambient sound speed.
              call eosp1 (r1,ergamb,1,gam,pre)
      30      camb=sqrt(gam*pre/r1)
c
c     determine conditions behind shock.
              rvxbbc=0.0
              rvybbc=mach*camb*(rhobbc/r1)
              ergbbc=ergbbc-0.5*rvybbc*rvybbc/rhobbc
```

```
2753            write (6,50) rhobbc,rvxbbc,rvybbc,ergbbc,mach
2754            return
2755 c
2756     40 format (1x,'no convergence to shock conditions')
2757     50 format (1x,'rho,rvx,rvy,erg,mach=',5e12.5)
2758        end
2759            subroutine initial (bcl,bcr,bcb,bct)
2760            parameter (mnx=150, mny=200, mxy=202)
2761            parameter (mnxx=1, mnyy=1)
2762            integer alpha
2763            logical lmgrid, lstat, lnuc, therml, lgrav, leos
2764            real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
2765            real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
2766            real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
2767            common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
2768            common lstat, lnuc, therml, lgrav, leos
2769            common alpha, gamma, gm1, dt, dx, dy, rhomin
2770            common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
2771 c
2772            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
2773            common /grids/ xcor, xcoro, ycor, ycoro, unit
2774 c
2775            real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
2776            real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
2777            real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
2778            real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
2779            real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
2780            real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
2781            real rmin(mxy), rmax(mxy), rvr(mxy)
2782            common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
2783          1 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
2784          2 , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
2785 c
2786            real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
2787            parameter (jthm=1)
2788            real fsky(mxy), rhtl(mnx,jthm)
2789            common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
2790          1 , jtherm
2791            data g0, r0 /980.,6.48e8/
2792 c
2793            real gams(mxy), ptest(mxy), ygp(mxy)
2794            equivalence (gams,sgam), (ptest,scrh), (ygp,sgrv)
2795 c
2796 c       compute coordinates & set them into common /grids/.
2797            call xgrid0
2798            call ygrid0
2799            write (6,220)
2800            write (6,210) (xcor(i),i=1,nxp)
2801            write (6,230)
2802            write (6,210) (ycor(j),j=1,nyp)
2803 c
2804 c       set up ambient atmosphere.
2805 c       first get height in KM.
2806            nypp=nyp+1
2807            do 10 j=1,mny
2808     10     ygp(j+1)=0.5*(ycor(j)+ycor(j+1))*1.e-5
2809            ygp(1)=1.e-5*(ycor(1)-0.5*(ycor(2)-ycor(1)))
2810            ygp(nypp)=1.e-5*(ycor(nyp)+0.5*(ycor(nyp)-ycor(ny)))
2811            jlim=cvmgt(nypp,2,lgrav)
2812 c
2813 c       get density and pressure.
2814            do 20 j=1,jlim
2815            call cav61 (ygp(j),rh00(j),tam,amass)
2816            rh00(j)=amax1(rh00(j),rhomin)
```

```
2817            pr00(j)=rh00(j)*8.3144e7*tam/amass
2818     20     continue
2819 c
2820 c       get gravitational acceleration and potential.
2821            gr2=cvmgt(g0*r0*r0,0.0,lgrav)
2822            do 30 j=1,nypp
2823     30     ygp(j)=ygp(j)+1.e5
2824            do 40 j=1,jlim
2825            gravy(j)=-gr2/(r0+ygp(j))**2
2826     40     gp00(j)=gr2*(ygp(j)-ygp(1))/((r0+ygp(1))*(r0+ygp(j)))
2827 c
2828 c       use real air EOS to get better gamma and energy.
2829            do 50 j=1,jlim
2830     50     er00(j)=pr00(j)/(gamma-1.0)
2831            if (.not.leos) go to 90
2832            kount=0
2833     60     if (kount.eq.50) then
2834            write (6,260)
2835            stop
2836            endif
2837            call eospt (rh00,er00,jlim,gams,ptest)
2838            kount=kount+1
2839            do 70 j=1,jlim
2840     70     er00(j)=pr00(j)/(gams(j)-1.0)
2841            do 80 j=1,jlim
2842            if (abs(pr00(j)-ptest(j)).gt.1.0e-6*pr00(j)) go to 60
2843     80     continue
2844     90     do 100 j=1,jlim
2845    100     er00(j)=er00(j)+rh00(j)*gp00(j)
2846 c
2847 c       set values in remaining cells.
2848            if (bcb.eq.-1.0.or..not.lgrav) then
2849            jlim=cvmgt(1,nypp,lgrav)
2850            do 110 j=1,jlim
2851            rh00(j)=rh00(2)
2852            pr00(j)=pr00(2)
2853            gravy(j)=gravy(2)
2854    110     er00(j)=er00(2)
2855            endif
2856 c
2857 c       compute "falling sky" term.
2858            if (lgrav) then
2859            fsky(1)=pr00(1)
2860            fsky(2)=pr00(2)
2861            do 120 j=2,nypp-1
2862    120     fsky(j+1)=fsky(j-1)+rh00(j)*gravy(j)*(ygp(j+1)-ygp(j-1))
2863            else
2864            do 130 j=1,nypp
2865    130     fsky(j)=pr00(j)
2866            endif
2867 c
2868 c       print ambient data.
2869            write (6,240)
2870            do 140 j=1,nypp
2871            jj=nypp-j+1
2872            write (6,250) jj,ygp(jj),rh00(jj),pr00(jj),er00(jj),gp00(jj)
2873           1     ,gravy(jj),fsky(jj)
2874    140     continue
2875 c
2876 c       fill fluid variable arrays
2877            do 150 j=1,mxy
2878            do 150 i=1,mxx
2879            rho(i,j)=rh00(j+1)
2880            ry(i,j)=0.0
```

```
2881            rvy(i,j)=0.0
2882    150     erg(i,j)=er00(j+1)
2883            go to (200,180,160), nspec
2884    160     do 170 j=1,mny
2885            do 170 i=1,mnx
2886    170     rhb(i,j)=0.0
2887    180     do 190 j=1,mny
2888            do 190 i=1,mnx
2889    190     rha(i,j)=rho(i,j)
2890    200     continue
2891 c
2892            return
2893 c
2894    210 format (1p10e13.4)
2895    220 format ('0xcor(i) - the x coordinate grid locations ',/,3x)
2896    230 format ('0ycor(j) - the y coordinate grid locations ',/,3x)
2897    240 format (3x,'j',7x,'y',17x,'rho(gm/cc)',3x,'pre(dynes/cm**2)',3x,'e
2898       1rg(ergs/cc)',3x,'gr pot(erg/gm)',3x,'gr acc(cm/sec**2)',3x,'fsky'/
2899       2 )
2900    250 format (1x,i3,7(5x,e12.5))
2901    260 format (1x,'failure to converge in subroutine inital')
2902        end
2903            subroutine burst (xcor,ycor,erg)
2904 c-----------------------------------------------------------------------
2905 c     deposits burst energy in 3 zones.
2906 c-----------------------------------------------------------------------
2907            parameter (mnx=150, mny=200, mxy=202)
2908            real erg(mnx,mny), xcor(mxy), ycor(mxy)
2909 c
2910            parameter (mxx=1, nsta=1)
2911            real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
2912            real prs(mxx,nsta), xs(nsta), ys(nsta)
2913            common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
2914       1    ixx
2915            data pi /3.1415926/, engin/0.0/
2916            namelist /blast/ burstn,eblast,nicel,njc1,njc2
2917 c
2918 c     read namelist data.
2919            read (5,blast)
2920            write (6,blast)
2921 c
2922 c     determine energy per cell.
2923            vol=pi*(xcor(nicel+1)**2*(ycor(njc2+1)-ycor(
2924       x     njc1)))
2925            ez=eblast*4.2e19*burstn/vol
2926            fracn=1.0
2927 c
2928 c     distribute burst energy.
2929            do 10 j=njc1,njc2
2930            delty=ycor(j+1)-ycor(j)
2931            do 10 i=1,nicel
2932             vol=pi*(xcor(i+1)+xcor(i))*(xcor(i+1)-xcor(i))*delty
2933            engin=engin+ ez*vol
2934    10         erg(i,j)=erg(i,j)+ez
2935           write(6,99) engin
2936    99     format(1x,'energy deposited=',1pe12.5,' ergs'/)
2937            return
2938        end
2939            subroutine cav61 (alt,r,t,m)
2940 c-----------------------------------------------------------
2941 c     calculate mass density, temperature and mean molecular weight
2942 c     based on the average model in the "cira 1961" tables.
2943 c
2944 c     data is stored in the following manner
```

```
c       0 - 120 km in intervals of 5 km,
c     120 - 300 km in intervals of 10 km and
c     300 - 800 km in intervals of 50km.
c-----------------------------------------------------------------
          real rh(53), tn(53), am(53), m
          data rh /1.23e 03,7.48e-04,4.19e-04,2.02e-04,8.89e-05,4.06e-05
     1    ,1.84e-05,8.69e-06,4.07e-06,2.02e-06,1.02e-06,5.54e-07,3.04e-0
     2    7,1.67e-07,8.84e-08,4.37e 08,1.94e-08,8.12e-09,3.12e-09,1.23e-
     3    09,4.78e-10,2.07e-10,9.49e-11,4.77e-11,2.44e-11,6.95e-12,3.07e
     4    -12,1.69e-12,1.11e-12,8.26e 13,6.59e-13,4.73e 13,3.61e-13,2.77
     5    e-13,2.14e-13,1.67e 13,1.30e-13,1.03e 13,8.11e-14,6.45e 14,5.1
     6    5e-14,4.14e-14,3.34e-14,1.23e-14,5.09e-15,2.33e-15,1.17e-15,6.
     7    15e-16,3.45e-16,2.00e-16,1.19e-16,7.22e-17,4.60e-17/
          data tn /289.3,256.7,222.4,214.1,217.1,222.5,228.8,236.4,247.9
     1    ,260.6,269.6,268.9,257.8,238.6,217.0,198.1,184.6,178.9,181.1,1
     2    92.5,212.1,237.6,265.0,297.6,343.0,569.4,799.1,1015.,1155.,117
     3    7.,1193.,1211.,1227.,1243.,1259.,1274.,1288.,1302.,1314.,1326.
     4    ,1338.,1348.,1359.,1401.,1436.,1466.,7*1474./
          data am /19*28.97,28.91,28.85,28.78,28.72,28.67,28.60,28.43,28
     1    .25,28.09,27.90,27.70,27.47,27.25,27.00,26.75,26.48,26.18,25.8
     2    7,25.55,25.21,24.85,24.50,24.11,23.74,21.80,20.00,18.55,17.48,
     3    16.31,15.70,15.10,14.62,14.02,13.88/
c
c     calculate appropriate index for table look-up.
          p1=.20*amin1(alt,120.0001)
          p2=.10*amin1(amax1(alt-120.0,0.0),180.0001)
          p3= 02*amax1(alt-300.0,0.0)
          i1=p1
          i2=p2
          i3=p3
          i=i1+i2+i3+1
          i=max0(i,1)
          i=min0(i,52)
c
c     calculate atmospheric quantities by linear interpolation.
          p=(p1-i1)+(p2-i2)+(p3-i3)
          omp=1.0-p
          r=rh(i)*omp+rh(i+1)*p
          t=tn(i)*omp+tn(i+1)*p
          m=am(i)*omp+am(i+1)*p
          return
        end
          subroutine dposit (time)
c-----------------------------------------------------------------
c     sets up a nuclear blast at (x0,y0,t0) or a HE
c     blast at (x0,y0,0.0) in a 2D cartesian or cylindrical grid.
c-----------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (mnxx=1, mnyy=1)
          parameter (nop=1)
          parameter (mxx=1, nsta=1)
c
          integer alpha
          logical lmgrid, lstat, lnuc, therml, lgrav, leos
          real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy),
          real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
          real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
          common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
          common lstat, lnuc, therml, lgrav, leos
          common alpha, gamma, qm1, dt, dx, dy, rhomin
          common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
          real rhom(mxy), pres(mxy), vol(mxy), ubl(mxy)
          real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
```

```
3009            real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
3010            real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3011            real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
3012            real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
3013            real rmin(mxy), rmax(mxy), rvr(mxy)
3014            common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
3015           1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
3016           2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3017 c
3018            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
3019            common /grids/ xcor, xcoro, ycor, ycoro, unit
3020 c
3021            real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
3022            parameter (jthm=1)
3023            real fsky(mxy), rhtl(mnx,jthm)
3024            common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
3025           1    , jtherm
3026 c
3027            real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
3028            real prs(mxx,nsta), xs(nsta), ys(nsta)
3029            common /stat/ rhs, vls, gms, prs, tme, xs, ys, fracn, eblast,
3030           1    lxx
3031 c
3032            parameter (mmr=2200)
3033            real pbl(mmr), rbl(mmr), vbl(mmr), gbl(mmr), ebl(mmr)
3034            real rad(mmr), vlc(mmr), ein(mmr), den(mmr), pre(mmr)
3035            real gma(mmr), pss(mmr)
3036            equivalence (pbl,pre), (rbl,den), (gbl,rad)
3037            equivalence (vbl,vlc), (ebl,ein)
3038 c
3039            real pav(mxy), rav(mxy), vav(mxy), gav(mxy), eav(mxy)
3040            real gams(mxy)
3041            integer inmin(mxy), inmax(mxy)
3042            equivalence (rav,rhoh), (pav,rvxh), (vav,rvyh)
3043            equivalence (gav,ergh), (gams,sgam), (eav,srvx)
3044 c
3045            logical part
3046            real xp(nop), yp(nop)
3047            common /part/ part, xp, yp, nopp
3048 c
3049            logical lanl
3050            namelist /depo/ x0,y0,t0,r0,eblast,lanl
3051 c
3052            read (5,depo)
3053            write (6,depo)
3054 c-----------------------------------------------------------
3055 c     add up the ambient energy in the grid  this will enable a
3056 c     comparsion of the final energy, and thus what the energy is in the
3057 c     explosion.
3058 c-----------------------------------------------------------
3059            call ngridd (xcor,nxp,alpha)
3060            call ogridd (nxp)
3061            call ngridd (xcor,nxp,alpha)
3062            do 10 j=1,mny
3063            call consrd (erg(1,j),nx,erg0(j))
3064     10     continue
3065            call ngridd (ycor,nyp,1)
3066            call ogridd (nyp)
3067            call ngridd (ycor,nyp,1)
3068            call consrd (erg0,ny,ambsum)
3069            if (.not.lnuc) go to 30
3070            if (abs(eblast-1000.).gt.10.) lanl=.false.
3071            if (lanl) go to 50
3072 c-----------------------------------------------------------
```

```
c     calculate a scale factor for the radius based on the blast energy
c     measured in kt. then evaluate ktnuc at radial intervals
c     of 1/4 the smallest cell dimension. scale r0 and dr
c     according to blast energy.
c-------------------------------------------------------------------
          rscale=(1.0/eblast)**(1.0/3.0)
          dr=0.25*amin1(dx,dy)*rscale
          nr=(1.2*rscale*r0/dr)+1
          nrp=nr+1
          radd=0.0
          do 20 i=1,nrp
          call ktnucl (t0,radd,ppp,dens,eee,vvv,gamma)
          pbl(i)=ppp
          rbl(i)=dens
          vbl(i)=vvv
          gbl(i)=gamma
   20     radd=radd+dr
          go to 50
c-------------------------------------------------------------------
c     deposit HE blast energy in the grid using CJDET at
c     1/4 the smallest cell dimension.
c-------------------------------------------------------------------
   30     rscale=1.0
          dr=0.25*amin1(dx,dy)
          nr=(1.1*r0/dr)+1
          nrp=nr+1
          x=-0.5*dr
          r0inv=1.0/r0
          do 40 i=1,nrp
          x=x+dr
          xpass=x*r0inv
          call cjdet (xpass,dens,vvv,eee,ppp,gamma)
          pbl(i)=ppp
          ebl(i)=eee
          rbl(i)=dens
          vbl(i)=vvv
   40     gbl(i)=gamma
   50     continue
c
c     calculate cell boundaries relative to (x0, y0). place in delx
c     and dely.
          do 60 i=1,nxp
   60     delx(i)=xcor(i)-x0
          do 70 j=1,nyp
   70     dely(j)=ycor(j)-y0
c
c     compute lengths of position vectors (radii) to cell corners.
          do 80 i=1,nxp
   80     rha0(i)=delx(i)*delx(i)
          do 90 j=1,nyp
   90     rhb0(j)=dely(j)*dely(j)
c
c     now calculate cell centers relative to (x0, y0). place in delx and
c     dely.
          do 100 i=1,mnx
  100     uh(i)=0.5*(delx(i)+delx(i+1))
          do 110 i=1,mnx
  110     delx(i)=uh(i)
          do 120 j=1,mny
  120     vh(j)=0.5*(dely(j)+dely(j+1))
          do 130 j=1,mny
  130     dely(j)=vh(j)
c
c     compute lengths of position vectors of the cell centers
```

```
3137            do 140 i=1,mnx
3138   140      scrh(i)=delx(i)*delx(i)
3139            do 150 j=1,mny
3140   150      sgrv(j)=dely(j)*dely(j)
3141            if (.not.lnuc) go to 160
3142            if (lanl) go to 330
3143 c-----------------------------------------------------------------
3144 c       calculate flow field at each point by interpolation onto grid.
3145 c       ktnuc provides pressure, density, and velocity, assuming
3146 c       ambient values of 1.01325e6, 1.225e-3, and 0.0, respectively.
3147 c       we scale the ktnuc values to the ambient values set up in inital.
3148 c-----------------------------------------------------------------
3149            pkt=1.01325e6
3150            rhokt=1.225e-3
3151   160      r00=r0*rscale/dr+1
3152            do 290 j=1,mny
3153            do 170 i=1,mnx
3154   170      rvr(i)=scrh(i)+sgrv(j)
3155 c
3156 c       pick up variables from 2D arrays.
3157            do 180 i=1,mnx
3158            rvx0(i)=rvx(i,j)
3159            rvy0(i)=rvy(i,j)
3160            rho0(i)=rho(i,j)
3161            rhom(i)=1.0/rho0(i)
3162   180      erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
3163 c-----------------------------------------------------------------
3164 c       interpolate the blast values onto the grid.
3165 c-----------------------------------------------------------------
3166 c       find maximum and minimum radius vectors to corners of each cell.
3167 c       scale according to blast energy.
3168            do 190 i=1,mnx
3169            vel(i)=rha0(i)+rhb0(j)
3170            uh(i)=rha0(i+1)+rhb0(j)
3171            vh(i)=rha0(i)+rhb0(j+1)
3172            rmin(i)=amin1(vel(i),uh(i),vh(i))
3173            rmax(i)=amax1(vel(i),uh(i),vh(i))
3174            vel(i)=rha0(i+1)+rhb0(j+1)
3175            rmin(i)=amin1(vel(i),rmin(i))
3176            rmax(i)=amax1(vel(i),rmax(i))
3177            rmin(i)=sqrt(rmin(i))*rscale
3178   190      rmax(i)=sqrt(rmax(i))*rscale
3179 c
3180 c       interpolate onto grid by averaging between rmin and rmax.
3181            do 200 i=1,mnx
3182            rmax(i)=rmax(i)/dr+1.0
3183            rmin(i)=rmin(i)/dr+1.0
3184            inmax(i)=int(rmax(i))
3185   200      inmin(i)=int(rmin(i))
3186 c
3187            do 220 i=1,mnx
3188            kmin=inmin(i)
3189            kminp=kmin+1
3190            kmax=inmax(i)
3191            kmaxp=kmax+1
3192            eav(i)=0.0
3193            pav(i)=0.0
3194            rav(i)=0.0
3195            vav(i)=0.0
3196            gav(i)=0.0
3197 c
3198            if (rmin(i).gt.r00) go to 220
3199 c
3200            do 210 k=kminp,kmax
```

```
3201            eav(i)=eav(i)+ebl(k)
3202            pav(i)=pav(i)+pbl(k)
3203            rav(i)=rav(i)+rbl(k)
3204            vav(i)=vav(i)+vbl(k)
3205  210       gav(i)=gav(i)+gbl(k)
3206 c
3207 c       interpolate at ends of interval (rmin,rmax).
3208            flow=1.0-rmin(i)+float(inmin(i))
3209            fhigh=rmax(i)-float(inmax(i))
3210            eav(i)=eav(i)+ebl(kmin)*flow+ebl(kmaxp)*fhigh
3211            pav(i)=pav(i)+pbl(kmin)*flow+pbl(kmaxp)*fhigh
3212            rav(i)=rav(i)+rbl(kmin)*flow+rbl(kmaxp)*fhigh
3213            vav(i)=vav(i)+vbl(kmin)*flow+vbl(kmaxp)*fhigh
3214            gav(i)=gav(i)+gbl(kmin)*flow+gbl(kmaxp)*fhigh
3215 c
3216 c       divide by number of points in interval (rmin,rmax).
3217            fpoint=rmax(i)-rmin(i)+1.0
3218            eav(i)=eav(i)/fpoint
3219            pav(i)=pav(i)/fpoint
3220            rav(i)=rav(i)/fpoint
3221            vav(i)=vav(i)/fpoint
3222            gav(i)=gav(i)/fpoint
3223  220       continue
3224            if (.not.lnuc) go to 250
3225 c
3226 c       now calculate new values for fluid variables.
3227            do 230 i=1,mnx
3228  230       ergk(i)=amax1(0.0,erg0(i)-0.5*rhom(i)*(rvx0(i)*rvx0(i)+rvy0(i)
3229     1       *rvy0(i)))
3230            call eospl (rho0,ergk,nx,gams,pres)
3231            do 240 i=1,mnx
3232            if (rmin(i).gt.r00) go to 240
3233            rho0(i)=(rav(i)/rhokt)*rho0(i)
3234            rho0(i)=amax1(rho0(i),rhomin)
3235            uh(i)=sqrt(rvr(i))
3236            rvr(i)=rho0(i)*vav(i)
3237            rvx0(i)=rvx0(i)+rvr(i)*delx(i)/uh(i)
3238            rvy0(i)=rvy0(i)+rvr(i)*dely(j)/uh(i)
3239            pres(i)=(pav(i)/pkt)*pres(i)
3240            gams(i)=gav(i)
3241 c
3242            rhom(i)=1.0/rho0(i)
3243            ergk(i)=(rvx0(i)*rvx0(i)+rvy0(i)*rvy0(i))*0.5*rhom(i)
3244            erg0(i)=pres(i)/(gams(i)-1.0)+ergk(i)
3245  240       continue
3246            go to 270
3247  250       do 260 i=1,mnx
3248            if (rmin(i).gt.r00) go to 260
3249            rho0(i)=amax1(rav(i),rhomin)
3250            uh(i)=sqrt(rvr(i))
3251            rvr(i)=rho0(i)*vav(i)
3252            rvx0(i)=rvx0(i)+rvr(i)*delx(i)/uh(i)
3253            rvy0(i)=rvy0(i)+rvr(i)*dely(j)/uh(i)
3254            erg0(i)=eav(i)
3255  260       continue
3256 c
3257 c       refill fluid variable arrays.
3258  270       do 280 i=1,mnx
3259            rho(i,j)=rho0(i)
3260            rvx(i,j)=rvx0(i)
3261            rvy(i,j)=rvy0(i)
3262  280       erg(i,j)=erg0(i)+rho0(i)*gp00(i,j)
3263  290       continue
3264            go to (320, 300, 300), nspec
```

```
3265   300      do 310 j=1,mny
3266            do 310 i=1,mnx
3267            rha(i,j)=cvmgt(rho(i,j),0.0,rho(i,j).gt.rha(i,j))
3268   310      continue
3269   320      continue
3270 c
3271            if (part) call partin (xcor,ycor,nx,ny,y0)
3272 c
3273 c      set time at start of calculation.
3274            time=0.0
3275            if (inuc) time=t0/rscale
3276            go to 480
3277 c
3278 c      read lanl data for a 1 megaton blast.
3279   330      continue
3280            open (unit=7,file='mt500',status='old')
3281            read (7,500) time,nrp
3282            rscale=(1.0/eblast)**(1.0/3.0)
3283            if (abs(time-t0).gt.0.01*time) stop 2222
3284            tkt=1.25*time*rscale
3285            do 340 k=1,nrp
3286            read (7,510) rad(k),vlc(k),ein(k),den(k),pre(k)
3287 c           radd=rad(k)*rscale
3288 c           call ktnucl (tkt,radd,pre(k),den(k),ein(k),vlc(k),gma(k))
3289 c           ein(k)=pre(k)/((gma(k)-1.0)*den(k))
3290   340      continue
3291            close (unit=7,status='keep')
3292 c
3293 c      recalculate gamma and energy using equation of state for air.
3294            do 350 k=1,nrp
3295   350      ein(k)=ein(k)*den(k)
3296            kount=0
3297   360      kount=kount+1
3298            if (kount.gt.40) stop 8888
3299            call eospi (den,ein,nrp,gma,pss)
3300            do 370 k=1,nrp
3301   370      ein(k)=pre(k)/(gma(k)-1.0)
3302            do 380 k=1,nrp
3303   380      if (abs(pre(k)-pss(k)).gt.1.e-8*pre(k)) go to 360
3304 c
3305 c      store data in terms of momentum and total energy.
3306            do 390 k=1,nrp
3307            ein(k)=ein(k)+0.5*vlc(k)*vlc(k)*den(k)
3308   390      vlc(k)=den(k)*vlc(k)
3309            do 400 k=2,nrp
3310   400      pre(k)=1.0/(rad(k)-rad(k-1))
3311 c
3312            do 470 j=1,mny
3313            if (abs(dely(j)).gt.rad(nrp)) go to 470
3314            do 410 i=1,mnx
3315   410      rvr(i)=sqrt(scrh(i)+sgrv(j))
3316 c
3317 c      pick up variables from 2D arrays.
3318            do 420 i=1,mnx
3319            rvx0(i)=rvx(i,j)
3320            rvy0(i)=rvy(i,j)
3321            rho0(i)=rho(i,j)
3322   420      erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
3323 c-------------------------------------------------------------
3324 c      interpolate the blast values onto the grid.
3325 c-------------------------------------------------------------
3326            do 450 i=1,mnx
3327            if (rvr(i).gt.rad(nrp)) go to 450
3328            kp=1
```

```
3329            do 430 k=2,nrp
3330            if (rvr(i).lt.rad(k-1).or.rvr(i).gt.rad(k)) go to 430
3331            kp=k
3332            go to 440
3333    430     continue
3334    440     fact=pre(kp)*(rvr(i)-rad(kp-1))
3335            rho0(i)=den(kp-1)+(den(kp)-den(kp-1))*fact
3336            rvx0(i)=vlc(kp-1)+(vlc(kp)-vlc(kp-1))*fact
3337            rvy0(i)=rvx0(i)*dely(j)/rvr(i)
3338            rvx0(i)=rvx0(i)*delx(i)/rvr(i)
3339            erg0(i)=ein(kp-1)+(ein(kp)-ein(kp-1))*fact
3340    450     continue
3341 c
3342 c       refill 2D arrays.
3343            do 460 i=1,mnx
3344            rho(i,j)=rho0(i)
3345            rvx(i,j)=rvx0(i)
3346            rvy(i,j)=rvy0(i)
3347    460     erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
3348    470     continue
3349 c
3350 c       calculate fraction of eblast deposited in the grid.
3351    480     continue
3352            call ngridd (xcor,nxp,alpha)
3353            call ogridd (nxp)
3354            call ngridd (xcor,nxp,alpha)
3355            do 490 j=1,mny
3356            call consrd (erg(1,j),nx,erg0(j))
3357    490     continue
3358            call ngridd (ycor,nyp,1)
3359            call ogridd (nyp)
3360            call ngridd (ycor,nyp,1)
3361            call consrd (erg0,ny,dpsum)
3362            fracn=(dpsum-ambsum)/(eblast*4.2e19)
3363            write (6,520) ambsum,dpsum,fracn
3364 c
3365            return
3366 c
3367    500 format (e12.6,16)
3368    510 format (5(1x,e14.8))
3369    520 format (1x,'energy in ambient mesh=',e12.5,' ergs'/1x,'energy afte
3370       1r deposition=',e12.5,' ergs'/1x,'fraction of eblast deposited (nuc
3371       2lear case only)=',e12.5//)
3372        end
3373            subroutine cjdet (x,rho,u,e,p,gamma)
3374 c-------------------------------------------------------------
3375 c       evaluation of the cj detonation flow field (spherical)
3376 c       x = r/rc (radius/charge radius)
3377 c       rho = density (g/cc)
3378 c       u = velocity (cm/sec)
3379 c       e = internal energy (ergs/g)
3380 c       p = pressure (dynes/cm**2)
3381 c-------------------------------------------------------------------------
3382              common /jwl/ r0, pcj, dcj, e0cj, gcj, a, beos, c, r1, r2, w
3383              dimension xx(56), rhorhn(56), uun(56), een(56)
3384            data xx /0.00,0.46,0.47,0.48,0.49,0.50,0.51,0.52,0.53,0.54,0.55
3385         1   ,0.56,0.57,0.58,0.59,0.60,0.61,0.62,0.63,0.64,0.65,0.66,0.67,0.
3386         2   68,0.69,0.70,0.71,0.72,0.73,0.74,0.75,0.76,0.77,0.78,0.79,0.80,
3387         3   0.81,0.82,0.83,0.84,0.85,0.86,0.87,0.88,0.89,0.90,0.91,0.92,0.9
3388         4   3,0.94,0.95,0.96,0.97,0.98,0.99,1.00/
3389            data rhorhn /0.58336908,0.58329052,0.58353359,0.58547372,0.5877
3390         1   1509,0.59020287,0.59290010,0.59578818,0.59884679,0.60206354,0.6
3391         2   0542828,0.60893458,0.61257368,0.61633992,0.62022847,0.62423760,
3392         3   0.62836248,0.63260019,0.63695151,0.64141273,0.64598429,0.650666
```

```
     4   24,0.65545791,0.66036117,0.66537666,0.67050570,0.67575240,0.681
     5   11676,0.68660545,0.69221991,0.69796503,0.70384878,0.70987409,0.
     6   71605092,0.72238618,0.72888911,0.73557264,0.74244702,0.74952847
     7   ,0.75683498,0.76438427,0.77220094,0.78031456,0.78875953,0.79757
     8   631,0.80681324,0.81654400,0.82684541,0.83783180,0.84965658,0.86
     9   254716,0.87683612,0.89311200,0.91249108,0.93788856,1.00000000/
         data uun /0.00000000,0.00000000,0.00482083,0.01035851,0.0166617
     1   4,0.02356024,0.03094239,0.03874840,0.04691900,0.05541681,0.0642
     2   1171,0.07328302,0.08260766,0.09216960,0.10195599,0.11195995,0.1
     3   2217087,0.13258135,0.14319155,0.15399441,0.16499040,0.17617987,
     4   0.18756266,0.19914278,0.21092318,0.22290809,0.23510681,0.247521
     5   95,0.26016891,0.27305350,0.28618804,0.29959211,0.31327447,0.327
     6   26002,0.34156698,0.35621879,0.37124780,0.38668105,0.40255913,0.
     7   41892713,0.43583086,0.45333302,0.47150749,0.49044237,0.51024151
     8   ,0.53102863,0.55299151,0.57632965,0.60133791,0.62841576,0.65815
     9   610,0.69143391,0.72980273,0.77623236,0.83853370,1.00000000/
         data een /0.39426151,0.39416990,0.39445356,0.39672121,0.3993476
     1   9,0.40227160,0.40545160,0.40886858,0.41250038,0.41633454,0.4203
     2   6077,0.42457420,0.42896569,0.43353057,0.43826503,0.44316903,0.4
     3   4823870,0.45347190,0.45887235,0.46443692,0.47016850,0.47606906,
     4   0.48213992,0.48838571,0.49480942,0.50141495,0.50821042,0.515197
     5   40,0.52238870,0.52978772,0.53740394,0.54525214,0.55333781,0.561
     6   67936,0.57028919,0.57918406,0.58838654,0.59791517,0.60779816,0.
     7   61806703,0.62875247,0.63989770,0.65155333,0.66377920,0.67664534
     8   ,0.69023508,0.70467544,0.72009885,0.73670238,0.75475061,0.77461
     9   543,0.79692996,0.82264954,0.85371912,0.89516944,1.00000000/
c-----------------------------------------------------------------
c     r0 = ambient charge density (g/cc)
c     pcj = cj pressure (megabars)
c     dcj = cj detonation velocity (cm/microsecond)
c     e0cj = cj energy (mbars-cc/cc)
c     gcj,a,beos,c,r1,r2,w=  jwl eos prameters
c-----------------------------------------------------------------
c     calculate cj jump conditions
          g=gcj
          wn=dcj*1.0e+06
          q=e0cj/r0*1.0e+12
          rhon=r0*(g+1.0)/g
          pn=r0*wn*wn/(g+1.0)
          en=q+0.5*wn*wn/((g+1.0)**2)
          un=wn/(g+1.0)
          if (x.ge.0.0.and.x.le.1.0) go to 10
          rho=1.229e-3
          u=0.
          e=2.542e6
          go to 20
c
c      evaluate self-similar cj detonation flow field
   10     rho=rhon*terpl(x,xx,rhorhn,56,0.0,0.0,3hrho,0)
          u=un*terpl(x,xx,uun,56,0.0,0.0,3hvel,0)
          e=en*terpl(x,xx,een,56,0.0,0.0,6henergy,0)
   20     call eosp2 (rho,e,1,gamma,p)
          return
      end
          function terpl (x,a,b,n,c1,c2,z,nj)
c-----------------------------------------------------------------
c     linear interpolation subprogram
c     input
c       x      abcissa input
c       a      abcissa breakpoints, ascending order
c       b      ordinate breakpoints
c       n      no. a,b data points
c       c1     abcissa data increments (mode 3)
c       c2     abcissa data starting pt (mode 3)
```

```
c       z     data overflow message. blank gives no message.
c       nj    interpolation integer for mode 2 operation
c     output
c       terpl linear interpolation result
c     three modes of operation
c       1   normal.  x,a,b known. nj unknown. do loop search reqd.
c       2   x,a,b,nj known. no search reqd.
c       3   quick interpolation scheme for equally spaced abcissa data
c     calls  none
c-----------------------------------------------------------------------
          dimension a(n), b(n)
          nm1=n-1
          j=c1*x+c2
          j=j+nj
          if (j.gt.nm1) j=nm1
          if (j.gt.0) go to 20
          do 10 i=1,nm1
          j=i
          if (x.le.a(i+1)) go to 20
   10     continue
          print 40, z,x
   20     terpl=a(j+1)-a(j)
          if (terpl.eq.0.) go to 30
          terpl=b(j)+(b(j+1)-b(j))*(x-a(j))/terpl
          return
   30     print 50, z
          terpl=alog(0.)
          stop
c
   40 format (40h range exceeded during interpolation of ,a6,12h data.
     1x = ,e15.7)
   50 format (10x,12hterpl error ,a6,24h data. zero denominator./10x,30h
     1error trace by alog(0.) below.)
      end
          subroutine tlayer (rho,rvx,rvy,erg,rhomin,sum)
c-----------------------------------------------------------------------
c     thermal layer model using sound speed data.
c-- ---------------------------------- ---------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (jthm=1)
c
          real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
          common /grids/ xcor, xcoro, ycor, ycoro, unit
c
          real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
          real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
          real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
          real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
          real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
          real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
          real rmin(mxy), rmax(mxy), rvr(mxy)
          common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
          real gamo(mxy), gamn(mxy), eint(mxy), cs(mxy)
          real xkt(mxy), ratio(mxy), pscr(mxy)
          equivalence (gamo,delx), (gamn,dely)
          equivalence (eint,vel), (cs,vh), (pscr,ergh)
          equivalence (xkt,rha0), (ratio,rhb0)
c
          real csamb(jthm)
          real rho(mnx,mny), rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
c
```

```
3521           real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
3522           real fsky(mxy), rhtl(mnx,jthm)
3523           common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
3524          1     , jtherm
3525 c
3526 c      calculate sound speed in ambient atmosphere.
3527           mnx1=mnx-1
3528           do 10 j=1,jtherm
3529           rho1=rho(mnx,j)
3530           rvx1=rvx(mnx,j)
3531           rvy1=rvy(mnx,j)
3532           erg1=erg(mnx,j)
3533 c
3534           eint1=erg1-0.5*(rvx1*rvx1+rvy1*rvy1)/rho1-rho1*gp00(j+1)
3535           call eospl (rho1,eint1,1,gam1,pres1)
3536           csamb(j)=sqrt(gam1*pres1/rho1)
3537      10   continue
3538           sum=0.0
3539 c
3540 c      set up thermal layer.
3541           do 80 j=1,jtherm
3542           do 20 i=1,mnx
3543      20   xkt(i)=0.5*(xcor(i)+xcor(i+1))
3544           call bag (xkt,csamb,j,cs)
3545           do 30 i=1,mnx
3546           rho0(i)=rho(i,j)
3547           rvx0(i)=rvx(i,j)
3548           rvy0(i)=rvy(i,j)
3549           erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
3550           ergk(i)=0.5*(rvx0(i)*rvx0(i)+rvy0(i)*rvy0(i))/rho0(i)
3551      30   eint(i)=erg0(i)-ergk(i)
3552           call eospl (rho0,eint,mnx,gamn,pres)
3553           icount=0
3554      40   icount=icount+1
3555           if (icount.gt.40) stop 5555
3556           do 50 i=1,mnx
3557           gamo(i)=gamn(i)
3558           rho0(i)=pres(i)*gamo(i)/(cs(i)*cs(i))
3559      50   eint(i)=pres(i)/(gamo(i)-1.0)
3560           call eospl (rho0,eint,mnx,gamn,pscr)
3561           do 60 i=1,mnx
3562           if (abs(gamo(i)-gamn(i)).gt.1.e-8*gamo(i)) go to 40
3563      60   continue
3564           do 70 i=1,mnx
3565           erg0(i)=eint(i)+ergk(i)
3566           sum=sum+(erg0(i)-erg(i,j))
3567           rho(i,j)=amax1(rho0(i),rhomin)
3568           rhtl(i,j)=rho(i,j)
3569      70   erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
3570      80   continue
3571           return
3572         end
3573           subroutine bag (xkt,csamb,j,cs)
3574 c-------------------------------------------------------------------
3575 c      sound speed vs ground range for MINISCALE.
3576 c-------------------------------------------------------------------
3577           parameter (mnx=150, mny=200, mxy=202)
3578           parameter (jthm=1)
3579           real xkt(mnx), cs(mnx), csamb(jthm)
3580           data csthm, xmin, xmax /8.327e4,2073.,.6645./
3581 c
3582           do 10 i=1,mnx
3583      10   cs(i)=cvmgt(csthm,csamb(j),xkt(i).ge.xmin.and.xkt(i).le.xmax)
3584           return
```

```
      end
          subroutine griddx (nreg,nxp,xval,scrh)
c-----------------------------------------------------------------
c     calculates x - coordinate values for variable grid size.
c-----------------------------------------------------------------
          parameter (mxy=202)
c
          real xbnd(5), dxr(4), ybnd(5), dyr(2)
          common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1    xbnd, ybnd, dxr, dyr, xleft
c
          real ibnd(11), xval(nxp), scrh(mxy)
c
c     set up the region boundary indices.
          nx=nxp-1
          ibnd(1)=1
          nreg1=nreg+1
          do 10 ireg=1,nreg
   10     ibnd(ireg+1)=ibnd(ireg)+(xbnd(ireg+1)-xbnd(ireg))/dxr(ireg)
          ibnd(nreg1)=float(nxp)
c
c     now the values of xval are interpolated in the region tables.
          ireg=1
          do 30 i=1,nxp
          do 20 ireg=1,nreg1
   20     scrh(ireg)=cvmgt(float(ireg),float(nreg1),float(i).le.ibnd
     1    (ireg+1))
          ireg=int(scrh(ismin(nreg1,scrh,1)))
c
c     i is in the region ibnd(ireg) to ibnd(ireg+1).
          xval(i)=xbnd(ireg)+(xbnd(ireg+1)-xbnd(ireg))*((float(i)-ibnd
     1    (ireg))/(ibnd(ireg+1)-ibnd(ireg)+1.0e-5))
   30     continue
c
c     smooth the discontinuities in grid size.
          do 60 ismth=1,nsmth
          do 40 i=2,nx
   40     scrh(i)=0.25*(xval(i+1)+2.0*xval(i)+xval(i-1))
          do 50 i=2,nx
   50     xval(i)=scrh(i)
   60     continue
c
          return
      end
          subroutine xgrid (jtherm)
c-----------------------------------------------------------------
c     calculates interface locations for moving grid problems.
c-----------------------------------------------------------------
          parameter (mnx=150, mny=200, mxy=202)
          parameter (mnxx=1, mnyy=1)
          integer alpha
          logical lmgrid, lstat, lnuc, therml, lgrav, leos
          real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
          real tem(mnxx,mnyy), sc1(mnxx,mnyy), rho(mnx,mny)
          real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
          common nx, ny, nxp, nyp, nspec  idump, idiag, lmgrid
          common lstat, lnuc, therml, lgrav, leos
          common alpha, gamma, gml, dt, dx, dy, rhomin
          common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
c
          real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
          common /grids/ xcor, xcoro, ycor, ycoro, unit
c
          real xbnd(5), dxr(4), ybnd(5), dyr(2)
```

```
      common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
     1    xbnd, ybnd, dxr, dyr, xleft
c
      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
      real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
      real rmin(mxy), rmax(mxy), rvr(mxy)
      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
     1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
     2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
c
c     save old x - coordinate values.
      do 10 i=1,nxp
   10 xcoro(i)=xcor(i)
c
c     determine shock-ambient interface.
      ihld=1
      jhld=0
      do 30 j=jtherm+2,mny
      do 20 i=1,mnx
   20 scrh(i)=cvmgt(float(i),0.0,erg(i,j).gt.1.1*erg(mnx,j))
      klim=ismax(mnx,scrh,1)
      ihld=max0(ihld,klim)
      jhld=cvmgt(j,jhld,ihld.eq.klim)
   30 continue
      xhld=xcor(ihld)+20.*dx
      xshk0=xbnd(3)-float(nahed)*dx
      if (xhld.lt.xshk0) return
      xtry=amin1(xshk0+dx,xhld)
      xbnd(3)=amin1(xtry+float(nahed)*dx,xbnd(5))
      xbnd(4)=amin1(xbnd(3)+float(nx-ndx)*hx,xbnd(5))
      xbnd(2)=amax1(xbnd(3)-float(ndx)*dx,xbnd(1))
c
c     compute the new grid as it slides along with the shock.
      call griddx (4,nxp,xcor,scrh)
      if (therml) call rhlyr
      return
c
      entry ygrid
      do 40 j=1,nyp
   40 ycoro(j)=ycor(j)
      return
c
      entry xgrid0
      xcor(1)=0.0
      if (lmgrid) go to 60
      do 50 i=2,nxp
   50 xcor(i)=xcor(i-1)+dx
      go to 70
c
c     standard moving grid for hob problems.  shock edge falls within
c     fine grid.
   60 continue
      nhx=int((xfine-float(ndx)*dx)/hx)+1
      nhx=max0(0,nhx)
      xbnd(1)=xleft
      xbnd(2)=xbnd(1)+float(nhx)*hx
      xbnd(3)=xbnd(2)+float(ndx)*dx
      xbnd(4)=xbnd(1)+float(ndx)*dx+float(nx-ndx)*hx
      xbnd(5)=xbnd(4)
      dxr(1)=hx
```

```
3713            dxr(2)=dx
3714            dxr(3)=hx
3715            dxr(4)=dx
3716            call griddx (4,nxp,xcor,scrh)
3717     70     do 80 i=1,nxp
3718     80     xcoro(i)=xcor(i)
3719            return
3720 c
3721            entry ygrid0
3722            ycor(1)=alt
3723            if (lmgrid) go to 100
3724 c
3725 c       for a nonmoving grid, we use uniform cells in the y-direction.
3726            do 90 j=2,nyp
3727     90     ycor(j)=ycor(j-1)+dy
3728            go to 130
3729 c-----------------------------------------------------------------
3730 c       for a moving grid, the lowest mdy cells are uniform and the rest
3731 c       increase geometrically with cell index. hy is the fractional in-
3732 c       crease from cell to cell.
3733 c-----------------------------------------------------------------
3734    100     continue
3735            mdyp=min0((mdy+1),nyp)
3736            do 110 j=2,mdyp
3737    110     ycor(j)=ycor(1)+float(j-1)*dy
3738 c
3739            fact=1.0+hy
3740            mdyp2=min0((mdyp+1),nyp)
3741            dylrg=dy
3742            do 120 j=mdyp2,nyp
3743            dylrg=fact*dylrg
3744    120     ycor(j)=ycor(j-1)+dylrg
3745    130     do 140 j=1,nyp
3746    140     ycoro(j)=ycor(j)
3747            return
3748        end
3749            subroutine dustm(istep,time)
3750 c-----------------------------------------------------------------------
3751 c       adds dust so as to model the dust pickup behind the shock front.
3752 c       the SOUR model used is based upon Mirel's model. the dust mass
3753 c       added at each step is mdot*dt*area*u*rho(air).
3754 c-----------------------------------------------------------------------
3755            parameter (mnx=150, mny=200, mxy=202)
3756            parameter (mnxx=1, mnyy=1)
3757 c
3758            integer alpha
3759            logical lmgrid, lstat, lnuc, therml, lgrav, leos
3760            real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
3761            real tem(mnxx,mnyy), scl(mnxx,mnyy), rho(mnx,mny)
3762            real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
3763            common nx, ny, nxp, nyp, nspec, idump, idtag, lmgrid
3764            common lstat, lnuc, therml, lgrav, leos
3765            common alpha, gamma, gm1, dt, dx, dy, rhomin
3766            common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
3767 c
3768            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
3769            common /grids/ xcor, xcoro, ycor, ycoro, unit
3770 c
3771            real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
3772            real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
3773            real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
3774            real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3775            real dnew(mxy)
3776            real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
```

```
3777            real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
3778            real rmin(mxy), rmax(mxy), rvr(mxy)
3779            common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
3780           1     rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
3781           2     , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3782 c
3783            real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
3784            parameter (jthm=1)
3785            real fsky(mxy), rhtl(mnx,jthm)
3786            common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
3787           1     , jtherm
3788 c
3789            real dirt(mxy), rair(mxy), rhae(mxy), eint(mxy), qnew(mxy)
3790            real delta(mxy),theta(mxy),delta2(mxy),deltas(mxy)
3791            real mdot(mxy),tauw(mxy), visrat(mxy), phi(mxy), mach(mxy)
3792            equivalence (dirt,rhb0), (rair,rhc0), (rhae,rho0)
3793            equivalence (eint,srvy) , (phi,rmin), (mach,rmax)
3794            data con3, ff, edp0 /3.509e-7,0.1,50./
3795            data beta/0.0064/, sigp/2.5/, G/980./, dia/0.01/
3796            data amua/620./ ,tw/300./, ff/0.01/
3797 c
3798 c     determine location of shock front. return if shock front has
3799 c     propagated less than 8 zones to the right.
3800 c           do 10 i=1,mnx
3801 c    10     scrh(i)=cvmgt(float(i),0.0,rho(i,1).gt.1.2*rho(mnx,1))
3802 c           ishk=ismax(mnx,scrh,1)
3803 c           sum=ssum(mnx,scrh,1)
3804 c           if (sum.lt.1.0) return
3805 c           inject=max0(1,ishk-8)
3806              inject=mnx
3807              if (inject.eq.1) return
3808 c
3809 c      mirels model for blowing dust layer
3810 c      solve the ode for delta the boundary layer thickness
3811 c
3812 c      next we need to compute the new mass added per unit time
3813 c      mdot = cft/2xB  where B defined as ln(1+B)=Bxcft/cf0
3814 c       and cft = 2beta x sigp x G x diameter/rhoair/uair**2
3815 c
3816 c           Luck has it that beta, sigp, g and diameter are constants
3817 c              beta = 0.0064
3818 c              sigp = particle density = 2.5 g/cm**3
3819 c              G = gravity = 980 cm/sec**2
3820 c              diameter = particle dia = 0.01 cm
3821 c                    well almost constant...
3822 c
3823 c           cf0 = 2tauw/rhoa/uair**2
3824 c       next tauw is defined as
3825 c            tauw = 0.0225 [ mua/rhoa/uair/delta]**0.25 x phi x rhoa x uair**2
3826 c       now you ask where does phi come from??
3827 c       Finally our final definition
3828 c          phi = [rhom/rhoa]**0.75 x [mum/mua]**0.25
3829 c
3830 c
3831              do 20 i=1,mnx
3832              rhom(i)=1.0/rho(i,1)
3833              rhoh(i)=rho(i,1)
3834              rair(i)=rho(i,1)*rha(i,1)
3835              vel(i) = rvx(i,1)*rhom(i)
3836              qnew(i)=(deltas(i)-delta2(i))*rair(i)*vel(i)
3837              ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))*rhom(i)
3838     20       eint(i)=erg(i,1)-ergk(i)-rho(i,1)*gp00(2)
3839              call eospi (rhoh,eint,inject,sgam,pres)
3840 c
```

```
c       obtain the ratio of viscosities, mum/mua
c
         do 25 i=1,mnx
           pres(i)=amax1(pres(i),1.e5)
         mach(i)=vel(i)/sqrt(sgam(i)*pres(i)/rair(i))
          visrat(i)=0.524*abs(mach(i))
 25      continue
c
c       begin the painful process of integration
c
           call ngridd(xcoro,nx+1,2)
           call ogridd(nx+1)
           call ngridd(xcor,nx+1,2)
c
c      obtain the velocities at cell interfaces
c
            do 26 i=2,nx
 26         uh(i)=(vel(i)+vel(i-1))*0.5
            uh(1)=0.0
            uh(nx+1)=uh(nx)
           call velocd(uh,nx+1,dt,nx+1)
c
c        next do the source terms
c
           do 1 i=1,mnx
 1         rha0(i)=-qnew(i)
           call sourcd(mnx,dt,5,rha0,vel,0.0,0.0)
           call sourcd(mnx,dt,5,delta2,pres,pres(mnx),pres(1))
           do 2 i=1,mnx
 2         rha0(i)=-theta(i)*vel(i)**2*rair(i)
           call sourcd(mnx,dt,1,rha0,rha0,0.0,0.0)
           do 3 i=1,mnx
 3         rha0(i)=qnew(i)*vel(i)
           call sourcd(mnx,dt,1,rha0,rha0,0.0,0.0)
           do 4 i=1,mnx
 4         rha0(i)=tauw(i)
           call sourcd(mnx,dt,3,rha0,rha0,0.0,0.0)
c
           call fryfct(qnew,rhae,mnx,1.0,0.0,1.0,0.0)
c
c          update the qnew array and compute the boundary layer
c          thicknesses,  deltas and delta.
c
          delty=ycor(2)-ycor(1)
           do 5 i=1,mnx
           qnew(i)=rhae(i)
          if(vel(i).eq.0.0) vel(i)=1.e 30
           deltas(i)=qnew(i)/(rair(i)*vel(i)) + delta2(i)
 5         delta(i)=amax1(.02*delty,8.*deltas(i))
c
           do 6 i=1,mnx
           mdot(i)=0.0
           if(vel(i).lt.1.e 4) go to 6
           ta=con3*pres(i)/rair(i)
           rhor=0.5*(tw/ta+1.0) + 0.11*(sgam(i)-1.0)*mach(i)**2
           phi(i)=(1.0/rhor)**0.75*(visrat(i))**0.25
           if(delta(i).lt.1.e-4)  then
           tauw(i)=0.0
           else
           tauw(i)=0.0225*(amua/(rair(i)*vel(i)*delta(i))**0.25)*
     x     phi(i)*rair(i)*vel(i)**2
            endif
           cfo=2.*tauw(i)/rair(i)/vel(i)**2
           cft=2.*beta*sigp*Gydia(rair(i)*vel(i)**2)
```

```
c      solve for B , ln(1+B)=B*cft/cfo
            raat=cfo/cft
              if(raat.le.1.0) go to 6
           B=raat*15.
           do 7 k=1,25
           a=B/(alog(1.0+B))
           B=raat-a*B
 7         continue
            mdot(i)=abs(cft*0.5*B*(rair(i)*vel(i)))
 6          continue
c
c      deposit dust in the flow field behind shock front.
           do 30 i=1,inject
           if(delta(i).lt.1.e-4) then
           dirt(i)=0.0
           else
           dirt(i)=mdot(i)*ff*dt/delty
           endif
           dnew(i)=rha(1,1)+dirt(i)
           rha(1,1)=cvmgt(dnew(i),rha(1,1),ergk(i).gt.edp0)
           rho(1,1)=rair(i)+rha(1,1)
           erg0(i)=amax1(0.0,eint(i)+1.e7*dirt(i)*con3*pres(i)/rho(1,1))
           erg(1,1)=cvmgt(erg0(i)+rho(1,1)*gp00(2)+ergk(i)/(rhom(i)*rho(i
     1     ,1)),erg(1,1),ergk(i).gt.edp0)
    30     continue
           if(mod(istep-1,100).eq.0) then
           write(6,57) istep, time
 57        format(1x,'dirt for step',i5,' and time ',1pe12.5)
             write(6,56) (dirt(i),i=11,30)
             else
             endif
 56         format(1x,1p8e12.5)
           return
c
           entry dust01
c
c      initialize dust routine by setting dust density to zero.
           do 40 j=1,mny
           do 40 i=1,mnx
    40     rha(i,j)=0.0
           do 50 i=1,mnx
           delta(i)=0.02*(ycor(2)-ycor(i))
           deltas(i)=delta(i)/8.0
           delta2(i)=0.0
           theta(i)=delta(i)*7./32.
           mdot(i) = 0.0
           tauw(i) = 0.0
 50        continue
 167         continue
       return
       end
           subroutine duster(istep,time)
c-----------------------------------------------------------------------
c      adds dust so as to model the dust pickup behind the shock front.
c      the SOUR model used is based upon the S**3 model. the dust mass
c      added at each step is .08*area*u*rho(air).
c-----------------------------------------------------------------------
           parameter (mnx=150, mny=200, mxy=202)
           parameter (mnxx=1, mnyy=1)
c
           integer alpha
           logical lmgrid, lstat, lnuc, therml, lgrav, leos
           real rha(mnx,mny), rhb(mnxx,mnyy), rhc(mnxx,mnyy)
           real tem(mnxx,mnyy), sct(mnxx,mnyy), rho(mnx,mny)
```

```
3969            real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
3970            common nx, ny, nxp, nyp, nspec, idump, idiag, lmgrid
3971            common istat, lnuc, therml, lgrav, leos
3972            common alpha, gamma, gm1, dt, dx, dy, rhomin
3973            common tem, sc1, rho, rvx, rvy, erg, rha, rhb, rhc
3974 c
3975            real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
3976            common /grids/ xcor, xcoro, ycor, ycoro, unit
3977 c
3978            real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
3979            real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
3980            real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
3981            real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3982            real rhoh(mxy), rvxh(mxy), rvyh(mxy), ergh(mxy)
3983            real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
3984            real rmin(mxy), rmax(mxy), rvr(mxy)
3985            common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
3986           1    rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
3987           2    , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3988 c
3989            real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravy(mxy)
3990            parameter (jthm=1)
3991            real fsky(mxy), rhtl(mnx,jthm)
3992            common /grav/ pr00, rh00, er00, gp00, gravy, fsky, rhtl
3993           1    , jtherm
3994 c
3995            real dirt(mxy), rair(mxy), rhae(mxy), eint(mxy), dnew(mxy)
3996            equivalence (dirt,rhb0), (rair,rha0), (rhae,rhc0), (dnew,srvx)
3997            equivalence (eint,srvy)
3998            data con3, ff, edp0 /3.509e-7,0.1,50./
3999 c
4000 c     determine location of shock front. return if shock front has
4001 c     propagated less than 8 zones to the right.
4002 c          do 10 i=1,mnx
4003 c    10    scrh(i)=cvmgt(float(i),0.0,rho(i,1).gt.1.2*rho(mnx,1))
4004 c          ishk=ismax(mnx,scrh,1)
4005 c          sum=ssum(mnx,scrh,1)
4006 c          if (sum.lt.1.0) return
4007 c          inject=max0(1,ishk-8)
4008            inject=mnx
4009            if (inject.eq.1) return
4010 c
4011 c     calculate kinetic energy.
4012            do 20 i=1,inject
4013            rhom(i)=1.0/rho(i,1)
4014            rair(i)=rho(i,1)-rha(i,1)
4015            ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))*rhom(i)
4016    20      eint(i)=erg(i,1)-ergk(i)-rho(i,1)*gp00(2)
4017 c
4018 c     get pressure.
4019            call eospi (rho,eint,inject,sgam,pres)
4020 c
4021 c     deposit dust in the flow field behind shock front.
4022            dy12=(ycor(2)-ycor(1))
4023            do 30 i=1,inject
4024            dirt(i)=0.08*rair(i)*dt*ff*rhom(i)*abs(rvx(i,1))/dy12
4025            dnew(i)=rha(i,1)+dirt(i)
4026            rha(i,1)=cvmgt(dnew(i),rha(i,1),ergk(i).gt.edp0)
4027            rho(i,1)=rair(i)+rha(i,1)
4028            erg0(i)=amax1(0.0,eint(i)+1.e7*dirt(i)*con3*pres(i)/rho(i,1))
4029            erg(i,1)=cvmgt(erg0(i)*rho(i,1)+gp00(2)+ergk(i)/(rhom(i)*rho(i
4030           1    ,1)),erg(i,1),ergk(i).gt.edp0)
4031    30      continue
4032               if(mod(istep,100).eq.0) then
```

```
4033            write(6,78)  istep,time
4034   78       format(1x,'dirt  at step',i4,' and time',1pe12.5)
4035             write(6,79) (dirt(i),i=11,30)
4036   79        format(1x,1p8e12.5)
4037                 else
4038                 endif
4039             return
4040 c
4041             entry dust0
4042 c
4043 c       initialize dust routine by setting dust density to zero.
4044             do 40 j=1,mny
4045             do 40 i=1,mnx
4046     40      rha(i,j)=0.0
4047         end
```

APPENDIX B

ABSTRACT FOR DNA

DUST SYMPOSIUM

Three-Dimensional Effects in Dust Clouds- A Numerical Approach

Mark A. Fry
Science Applications International

Three-dimensional numerical simulation codes combining minimal diffusion algorithms which are exceptionally fast on today's supercomputers have been used to investigate nuclear dust cloud scenarios. Results show persistent multi-dimensional effects for long periods of time. Conclusions based solely on two- dimensional(2-D) calculations are misleading and sometimes incorrect. Modeling of high explosive dust cloud scenarios has also been performed and a unique high explosive and numerical simulation program for dust clouds is proposed. When more than one explosion occurs with proximity, multi- dimensional effects become important. Separation distances up to many fireball radii produce highly complex flows which result in asymmetrical dust cloud geometries. Even separation distances less than a fireball radius can influence cloud shape. The need to perform parametric analysis of these effects has led to the development of a very fast non-diffusive 3-D code, FAST3DD with dust entrainment. Pertinent aspects of the physics modeled will be explained. High explosive dust cloud simulation provides a research path when combined with numerical calculations can lead to new understanding of entrainment processes and cloud develoment. Improvements in the understanding of the dust lofting mechanism have been made with implementation of a turbulent of a turbulent boundary layer model for use with in 2 and 3-D codes. Multi-cloud experiments with HE are proposed with direction from 3-D calculations. Spherical charges of 1000 pounds or more that can be remotely detonated and launched into a proper configuration would be used. A sample scenario has been computed and results will be shown.

APPENDIX C

CHEMICAL EXPLOSIVE CHARGE STUDY USING 3-D FCT CODE

Distributed Explosive Charge Array (DECA)

Annual Report

Science Applications International Corporation
McLean, Virginia

and

Naval Research Laboratory

Sponsor: Defense Advanced Research Projects Agency

Project Officers: Col. Alexander H. Lancaster and Maj. Randy Lundberg

Program Code: 5G10/6G10

Program Element Code: 62702E

Project Name: DECA-Distributed Explosive Charge Array

Period covered: 1 August 1985 - 31 July 1986

Principal Investigators: Dr. Shmuel Eidelman (SAIC) and Dr. Joseph Baum (NRL)

NRL, Code 4040, Tel. (202) 767-3254

Work performed by: Shmuel Eidelman, David L. Book and Joseph Baum

TABLE OF CONTENTS

SECTION PAGE

Introduction

This report describes the results of the one year feasibility study of the Distributed Explosive Charge Array (DECA) concept, as applied to antitank mine neutralization. The DECA concept was proposed as an area weapon which would have the maximum blast effect for a given weight of explosive. The goal of our study was to substantiate this claim, by detailed numerical simulations and blast experiments. In the current study we considered DECA arrays formed by line charges of explosive positioned with selected distances between charges. The simulation of the blast wave propagation from a multitude of finite length explosive charges was done on a CRAY-XMP supercomputer, using a fully 3–Dimensional, time dependent numerical model. Parametrical study of the maximum pressure and impulse dependence on surface density was done for DECA formed from strands of Primacord. The formation and propagation of blast waves from the DECA was studied for arrays suspended above the ground, and lying on the ground, with different spacings of the Iremite or Primacord charges.

Experiments were conducted at SRI International remote test facility at Corral Hollow, California. Experiments were done with full size DECA arrays of Iremite or Primacord explosives, and were designed to follow numerical simulations. The experimental data was used to verify the computer simulations. To illustrate the effectivness of DECA against antitank mines, in two tests M–15 mines were desroyed by DECA arrays, suspended 25cm and 50cm above the ground.

1. Numerical Simulations

For numerical simulation of the blast waves of complex structures generated by DECA explosions, we used FAST3D computer code. This code was developed recently in the Laboratory for Computational Physics at NRL, and it solves unsteady and fully three–dimensional flow problems, with good resolution of the shock waves and contact discontinuities. This code is vectorized and optimized for the most efficient use on the Cray X–MP supercomputer. More details about this code structure and the numerical method used are given in Ref.1. This reference is added as Appendix A to the current report.

The purpose of the current study was to achieve levels of overpressure and impulse loads which cause collapse of the antitank mine casing. According to the BRL study (Ref 2.), published in 1983, pressure levels of above 2000psi and the blast wave impulse of above 950psi · sec will cause rupture of the M–15 mine casing, in the area of the central fuse cavity. This datum was taken as a reference point for defining the range of explosive surface density able to produce the load needed to destroy antitank (AT) mines.

The size of the DECA array was first choosen to be 4m × 4m. This is a relevant size for potential applications, and it allowed us to estimate the magnitude of the edge effects (nonplanar relief of the blast energy at the edges of the DECA charge). Later, it was realized that the edge effects are small at the time when most of the blast load affects the ground, and it was decided to study arrays of 2m × 2m in order to reduce expenses of the experiments, and improve the accuracy of numerical simulations.

In order to assure good resolution of the blast waves in all spatial directions, all the simulations were done on an evenly spaced computational mesh (60 × 50 × 50). The linear geometry of the individual charges was chosen because of the simplicity of its implementation, both in experiments and simulations. Only one fourth of the physical plan was simulated numerically, because of symmetry considerations.

For a typical simulation, at the time $t = 0$ we deposited energy and density (corresponding to levels observed for the explosives under consideration) into cells of the computational domain, located along straight lines and evenly spaced. Then an inital value problem was solved explicitly for the system of three-dimensional Euler equations. The pressure, density, velocities (in X,Y and Z directions) and energy were recorded in the form of binary arrays of values for every grid point in the computational domain. Because of the large size of these arrays, they were recorded only for a given increment of time or integration step (usualy 8 times per simulation). However, all physical parameters were recorded at every integration step for 144 points, located in the most relevant places of the computational domain (including the points corresponding to the location of the pressure transducers in the experiments).

For analysis of the complex three-dimensional flow field developing after a typical DECA explosion, a package of graphic subroutines was developed. The plots produced by these subroutines are demonstrated below, where we discuss the results of our simulations. Here we list the various functions of the graphics routines:

1. Contour plots of X–Y cross sections of the domain;
2. Contour plots of X–Z cross sections of the domain;
3. & 4. The same as 1. & 2. in color;
5. Time history of pressure, impulse, density and temperature (or total pressure);
6. Contour plots of the impulse distribution on the ground;
7. Graphs of pressure and impulse distribution on the ground.

The first set of computations and experiments was done with Iremite explosive. The surface density of the explosive was about 2.6kg/m^2. Figure 1.1 shows a schematic layout of a typical simulation. Initially, the explosive arrays consisted of nine 4m long lines of Primacord spaced 0.5m and placed on the ground. Because of symmetry, in the computational domain we have 4.5 lines (the line at the center of the array is cut in half by the symmetry plane Z–Y) and the lines are

2m long (again, physical lines of explosive are cut in half by the symmetry plane Z–X). In Figures 1.2(a,b,c,d), pressure contours are shown in the X–Z plane for the blast wave propagating after the explosion of a 100lb charge of DECA explosive formed by nine 4m charges of Primacord. The pressure contours are shown for $t = 0.13$ms, $t = 0.30$ms, $t = 0.49$ms, $t = 0.78$ms after the detonation. It is noticeable that in this case, because of the 0.5m space between the charges, the load on the ground is not uniform. Also, the edge effects are significant in this case. In Figures 1.3(a,b,c,d), pressure contours are shown for the X–Y plane (the ground). The maximum pressure in these plots is allocated along the lines; although this maximum is moving along the ground in time (compare Fig. 1.3b and Fig. 1.3c), this is not enough to issure an even load on the ground. In Figures 1.4(a,b,c) the contour plots of the impulse on the ground are shown for $t = 0.25$ms, $t = 0.50$ms, $t = 1.0$ms, after detonation. The maximum impulse achieved in this simulation was ≈ 1500psi $\cdot$ ms. These levels were observed directly under the lines of explosives. In the areas between the lines of Primacord, the impulse is around 450psi $\cdot$ ms. These areas of small impulse are ≈ 20cm wide. The blast wave contibutes most significantly to the impulse in the first 0.5ms after the detonation.

In Figures 1.5(a,b), graphs of pressure, impulse, temperature and density are shown for the same simulation, as functions of time elapsed after the detonation. (The coordinates of the station where the recording was made are given in the third line of the graph headings.) The station coresponding to Fig. 1.5a is located under the explosive. That leads to very high presssure and impulse. However, for the station located between the lines of explosives shown in Fig. 1.5b, the maximum pressure is only about 1.5kpsi and maximum impulse is 500psi $\cdot$ ms. Initially, the uneven load distribution was not considered a negative factor, since it was assumed that to destroy an AT mine it would be sufficient to produce a very high load on part of the mine . Later, it was learned that this concept could be applied only to AT mines which are activated by large pressure plates, and it would be inefficient for blast resistant mines. For this reason, later in our study we tried to get more

even distribution of the blast wave load on the ground.

Let's consider in more detail the dynamics of the blast wave for DECA. Simultaneous detonation of a Distributed Explosive Charge Array formed by line charges of explosive will at first lead to cylindrical blast waves in the vicinity of the charges. At a later time, cylindrical blast waves from single charges of the array collide and after multiple reflections, a pseodo planar wave would form. Since the main advantage of the DECA concept is the use of planar blast waves to efficiently produce blast loads on the ground, earlier formation of the planar blast wave leads to increased efficiency. At the limit, the distance between the charges could be so large that DECA will not produce a greater blast load than single line charges of the same weight detonated separately. That will take place when collision of the blast waves from two parallel charges of the array will produce lower pressure than the target threshold. At another extreme, a large amount of very closely spaced lines of explosive will be difficult to detonate simultaneously, and if the line is thinner than a critical radius it will not detonate at all. A feasible DECA device would have parameters laying between these two limits, and one of the tasks of our proposed second-year development program will be to determine the range of DECA parameters where this concept is most effective. In our current study, the main task was to show the feasibility of DECA, and to correlate simulations with experimental tests. For this reason, we worked only with two spacial distances between charges: 50cm and 25cm.

In order to smooth the blast load distribution on the ground it was first decided to use different explosives. Iremite is a very "slow" explosive, with a density of only about $0.9g/cm^3$. Its detonation velocity is 3500m/sec. It is also produced in strands and it could be used is the same way as Primacord. It was also decided to use 10 lines of explosive instead of 9, so the Z-Y symmetry plane would lie between the lines of explosive. That will extend the size of the array to 4.5m × 4m. In Figure 1.6 the results of this simulation are shown in the form of pressure and impulse distribution on the ground. From the graphs in Fig. 1.6 it could be concluded that

substituting explosive from Primacord to Iremite would not improve the blast load distribution.

In order to achieve a more even blast load on the ground, the distance between was reduced from 50cm to 25cm. For the array of 4m × 4m, 25cm spacing between the charges will lead to 17 lines of explosive. The surface density of explosive in this simulation was same as in previouse simulation with Iremite. In Figures 1.7(a,b) and 1.8(a,b) contour plots of pressure, for X–Z and X–Y cross sections corespondingly, are shown for $t = 0.15$ms and $t = 0.35$ms. It is easy to see in these figures that for that case, the planar front forms almost imediatly after detonation and few contour lines close to the ground in Fig. 1.7a indicate that the blast was originated from the line sources. However, more detailed examination of the pressure time history for the point under the line explosive, shown in Figure 1.9a, and for the point between the lines of explosives, shown in Figure 1.9b, demonstrates large variation of blast load on the ground. In Fig. 1.9a pressure reached the same maximum level as in the case of 10 lines of Iremite (see Figure 1.6), but because the lines of explosive are thinner, in the current simulation the maximum impulse is only 930psi·ms. Between the lines the maximum pressure is 4.1kpsi, twice as large as in the previouse case, and the impulse is 580psi · ms.

Another method to achieve an evenly distributed blast load on the ground is to detonate DECA at some distance from the ground. In that case, multiple collisions of the blast waves from neiboring charges and formation of the planar front would begin prior to the start of the reflection from the ground. Because the incident blast wave in this case would be smoother, its reflection from the ground will produce more evenly distributed load. In order to prove this concept we simulated detonation of the DECA aray of 17 line charges of Iremite (length and weight was the same as in previous simulation) suspended 20cm above the ground. In Figures 1.10(a,b,c,d), results for this simulation are given in the form of pressure, impulse, density and temperature time histories for the points located on the ground. All these figures show $\approx$ 10% variation in pressure and less than

5% variation in impulse. That shows that suspention is the most powerful practical method to achieve even distribution of the blast load on the ground. The maximum pressure observed in simulation of the detonation of 17 lines of Iremite 20cm above the ground is 3250psi and the maximum impulse was 650psi · sec. This blast load is not suficient for the neutralization of an AT mine. In order to increase the blast load we decided to use strands of Primacord explosive, since the Primacord explosive is about 50% more energetic than Iremite.

In Figures 1.11 and 1.12 pressure contours are shown for the X–Z and X–Y cross sections of the blast wave formed by detonation of an 2.25m × 2m DECA device formed of 10 lines of Primacord spaced 25cm apart and suspended 25cm above the ground. Total weight of explosives in that simulation was 30lb. The pressure contours in Fig. 1.11 are shown for t = 0.06ms, t = 0.12ms, and t = 0.27ms. Better grid resolution allows as to follow details of the evolution of the blast wave for this DECA detonation. At t = 0.06ms the blast wave is formed by cylindrical detonations from the parallel line charges, and colisions between the parallel waves have only started. The primary waves from the line charges can be clearly located in Fig.1.11 at time 0.06ms; at that time the front of the blast wave has not reached the ground. At t = 0.12ms the reflection from the ground begins. The reflected blast wave is stronger between the charges, because of colision of the blast waves from two parallel sources. At t = 0.27ms reflection of the blast wave is fully developed. As it could be seen in Fig.1.12 at that time pressure is distributed evenly on the ground. The levels of pressure and impulse could be examined in more details in Figures 1.13(a.b.c.d), where the time histories of pressure, impulse and density are shown for selected points on the ground. The maximum pressures shown in Figs.1.13 are in the range of 7kpsi to 4.5kpsi and the impulse $\approx$ 800psi · ms. These levels of blast loads should be suficient to destroy an AT mine. Following this conclusion, in our experiments with AT mines we used a DECA charge of the same geometry and surface density as in the last described simulation, and it was suspended 25cm above the ground.

We have studied the dependence of maximum pressure and impulse, produced by the blast wave on the ground, on surface density of Primacord explosive. In Figure 1.14 the results of this study are shown on the log(Pressure) vs. log(Impulse) graph. DECA charges for this study were about 4m × 4m in size, suspended 25cm above the ground. On the same graph, vulnerability limits are shown for mines and other military targets and graphs representing dependence of M–58 line charge and FAE on surface density (volume density for FAE) are shown. From Fig.1.14 it is possible to conclude that DECA gives distinct advantages over M–58 and FAE. The a dvantage of DECA over FAE is in higher pressure. According to Fig.1.14, DECA is more efficient than M–58, since it can produce the same pressure on the ground using about fourth of M–58 weight. Also, the impulse produced by DECA is larger than M–58's impulse.

2. EXPERIMENTAL

All the experimental work reported here was done by SRI International of Menlo Park, California. In this part of our report, we will use the data and information form Ref.3.

The main objectives of the experimental work described in this report were to provide data for correlation with the numerical simulations, and to perform tests on inert mines that confirm the feasibility of this approach to damage or destroy antitank mines.

Approach and Test Procedures

The experimental technique, to meet the objectives stated above, was to detonate arrays of explosive placed on or above the ground and to measure the airblast overpressure at several points above and to the side of the array. Figure 2.1 shows a schematic layout of a typical experiment. Initially, the explosive arrays consisted of ten 4.5m lines of Iremite spaced 0.5m apart and placed on the ground covering an area 4.5m × 4.5m (Figure 2.2). Each line, was made of two or three strands of 1.125 inch diameter Iremite. Each Iremite strand weighs about 0.59kg/m (1.18lb/ft) and the total charge weight was from 52.7kg (116lb) to 79.1kg (173.9lb). The surface density of the explosive arrays was from 2.6kg/m^2 (0.53lb/ft^2) to 3.9kg/m^2 (0.80lb/ft^2). The lines of Iremite were each initiated by a single 3.65m long (12ft) strand of 25gr/ft Primacord. These lead-in strands were bundled together at a single point and initiated by a single 9.1m long (30ft) strand of Primacord. This initiation scheme produces a detonation wave in each line. The detonation front from each line sweeps along the array simultaneously. The detonation velocity of the Iremite is 3500m/s (11,500ft/s). The entire lead-in and array is detonated within 3.1ms of initiation.

For tests 1 through 6, the Iremite was placed directly on a prepared soil bed. The soil bed consisted of a firm soil base that had been leveled. A thin layer (about

5cm) of sand was placed over the soil and packed in place using a hand tamper. The sand provided a smooth surface on which to lay the Iremite. In later tests (test 7), the Iremite array was suspended 0.25 m (10 inches) above the test bed. Columns of styrofoam were used at 4 points along the length of each line to ensure a known uniform distance from the explosive to the ground.

For tests 8, 9, and 10, the Iremite was replaced by bundles of 400gr/ft Primacord. The spacing between lines of explosive was reduced from 0.5m to 0.25m and the array dimensions were reduced to 2.5m × 2.5m. However, the total explosive surface density (2.6kg/m^2) was approximately the same. Therefore, each line of explosive consisted of 7 strands of 400gr/ft Primacord. Each line had a lineal density of 0.59kg/m (the same lineal density of a single strand of Iremite). With these Primacord lines on 0.25m spacings, the surface density of the array is 2.4kg/m^2 compared with 2.6kg/m^2 for the Iremite array.

The Primacord arrays were suspended 0.25m above the ground (tests 8 and 9) or 0.5m above the ground (test 10) in the same manner as the Iremite array in test 7. The Primacord arrays were initiated in the same way as the Iremite arrays.

Instrumentation for these tests consisted of up to 6 pressure gauges mounted in 3 steel poles (Figure 2.2) that were buried in the ground in the central area of the test bed. The poles were 10.2cm diameter (4in) pipes approximately 4.8m long (15.5ft) long that were buried 1.5m (5ft) leaving 3.3m (10.5ft) above the ground. Two of the poles were 1m apart and third pole was 2.25m to one side (Figure 3). Gauge adaptors were inserted in each pole. Gauges could be placed 1, 2, or 3m above the ground in each pole. For the tests described here, only gauge positions 1m and 2m above the ground were used. For tests 1, 2, and 3, piezoelectric pressure gauges (PCB Model 119) were used. Because of large transverse accelerations in the steel poles, these gauges produced unacceptable records. For tests 4 through 10, piezoresistive pressure gauges (Endevco Model 8530A) were used. These gauges produced fully acceptable records. The frequency response of these gauges is about 90 kHz, which means that measurements of pressure rise times of about 2μsec to

$3\mu s$ are possible. Figure 2.3 shows a layout of the array for Tests 4 and 5 showing gauge positions and gauge numbers, and the explosive positioning.

Pressure data were recorded on a high speed tape recorder that has a frequency response of 80 kHz. The analog tape was played back through a recording digital oscilloscope. The digitized data were stored on a floppy disk for later reduction by a computer to produce hard copy records.

Inert antitank mines were exposed to the blast in tests 8, 9, and 10. These last three tests were designed to establish whether such an explosive array could disable an antitank mine. The inert mines consisted of a mine body that was filled with candle wax to simulate the mine explosive (Figure 2.4a). Before each test, an inert booster and detonator were placed in the pressure plate assembly on the top of the mine and the arming cap was screwed into place.

On these tests, the explosive arrays were suspended either 0.25m (tests 8 and 9) or 0.5m (test 10) above the ground (Figure 2.4b). The antitank mine was placed in the center of the array. The explosion array was positioned so that the instrumentation pole containing gauges G5 and G8 was one mine diameter from the edge of the center mine ($\approx 30cm$). The other instrumentation pole containing gauges G4 and G7 was on the edge of the small array. The third pole was not instrumented. The mine was placed on the hard subsoil, about 5cm (2in) below the sand bed surface, and sand was packed around the mine.

Test Results

Table 1 summarizes the results of the 10 tests performed in this program. As mentioned earlier, tests 1, 2, and 3 produced no usable data, although these tests did demonstrate that the instrumentation poles and tests bed were able to withstand repeat testing. Tests 4 and 5 demonstrated the reproducibility of the technique. Figure 2.5 shows records from these two tests. Other than differences in the peak pressure of up to 35 %, the records are similar. The differences in the peak pressure can be attributed to the spacial gradient of the wave fronts as they pass the fixed gauge position. Small changes in positioning of the DECA explosives

on the ground could lead to a change in the effective gauge position, and that in turn could lead to a change in pressure record, because of the large spacial gradient of the blast wave pressure. Pressure gauges G7 and G8 are located symmetrically, and should produce similar pressure records. However, gauges G7 and G8 recorded substantially different levels of pressure. We will try to explain this difference in the next part of the report.

On test 6, the charge density was increased by 50% from 2.6kg/m^2 to 3.9kg/m^2 The pressure measured at the gauge 7 location 1m above the ground was 3964kPa, which is an increase of 31% from the pressure measured in test 5. An increase of 50% in pressure should not be expected in this case, since pressure is linearly proportional to blast energy only for a planar blast wave. In our case, at the distance of 1m above the ground, pressure is some complex function of energy in between the laws for planar and cylindrical blast waves. The increase in pressure is even smaller for gauges G4 and G5 for an unclear reason. More detailed measurements are needed in order to study the dynamics of the blast wave from DECA, and its dependence on the energy and geometry of the array.

In test 7, the nominal explosive array was suspended 0.25m above the ground. The effect of this configuration was to decrease peak pressure from about 3000kPa (435psi) to about 2860kPa (415psi) at the 1m point, and from about 2500kPa (363psi) to about 2240kPa (325psi) at the 2m point (Figure 2.6). The rise time of the pressure pulse was also increased when the explosive was suspended above the ground.

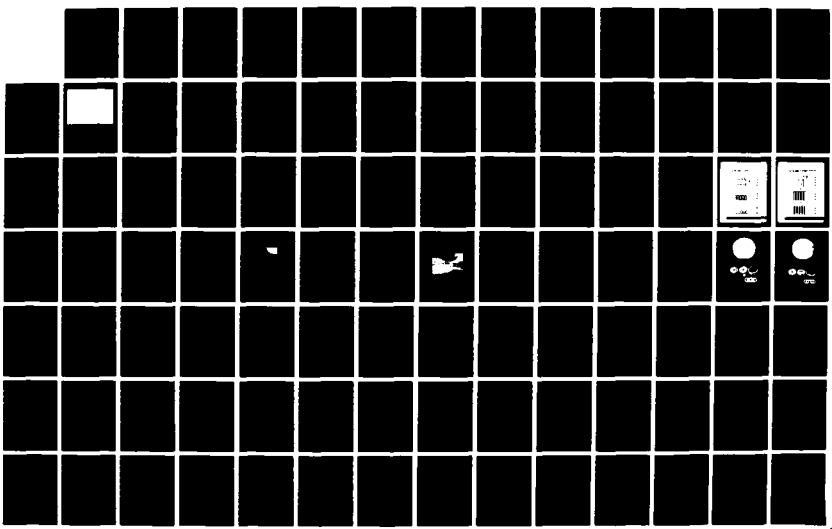
For tests 8, 9, and 10, the Iremite was replaced by bundels of 400gr/ft Primacord. The spacing between lines of explosive was reduced from 0.5m to 0.25m and the array dimentions were reduced to 2.5m × 2.25m. However, the average surface density of explosive ($2.6kg/m^2$) was maintained. The Primacord explosive (PETN) is about 50% more energetic than the Iremite explosive, so for the same surface density, a larger pressure is expected from a Primacord array than from an Iremite array. This proved to be the case, as evidenced by comparing the results of tests 7

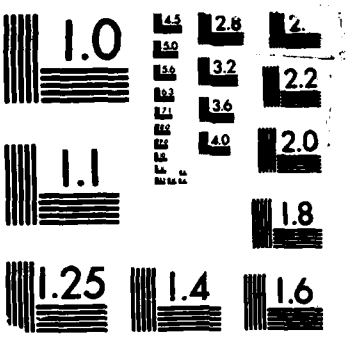
and 9. Test 7 used Iremite suspended 0.25m above the ground, whereas test 9 used the same areal density of Primacord suspended 0.25m above the ground. The peak pressure at the 1m point was 2S60kPa (415psi) for the Iremite array and 4S60 kPa (705 psi) for the Primacord array, a 70% increase in pressure. The large increase in pressure proves that for a smaller spacing of the line charges the planar wave will form and that will lead to higher overpressures per given weight of explosives. Also included in tests 9 and 10 were inert antitank mines. These mines included a mine casing that was filled with candle wax to simulate the mine explosive, a pressure plate assembly, and an inert booster and detonator. The mines were about 30cm (11.Sin.) in diameter and about 15cm (6in.) high. They were placed in the center of the Primacord array, which was 25cm about the ground surface. The mines were centered between lines of primacord that were spaced 25cm apart. The mines were buried so that their bases were on the firm soil subgrade about 5cm (2in.) below the sand bed surface. The upper 10cm (4in.) of the mine was above the ground level.

On test 9, the mine was severly crushed and the pressure plate, the detonator, and the booster assemblies were completely blown off of the mine. Figure 2.7 shows the damage to this mine. On test 10, the explosive array was raised to 50cm above the ground. On this test, the mine was crushed similarly to that in test 9, and the upper portion of the pressure plate and the detonator were blown away from the mine. The lower pressure plate and the rubber seal around the pressure plate along with the booster remained on the mine. This damage can be seen in Figure 2.Sa. When the mine was examined after the test, the booster and lower pressure plate assembly were easily removed from the mine. The pressure plate pieces, the detonator, and the booster are shown in figure 2.8b.

AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS 2/3
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

UNCLASSIFIED F/G 19/11 NN





3. Comparison Between Numerical Simulations and Experiments

One of the objectives of the current study was to compare results of numerical simulation of DECA to blast experiments with DECA charges. Because of budgetary constrains we could compare only pressure history in locations above the DECA charge, where pressure has been measured. Only in 6 experiments of the 10, were useful pressure measurments made, and data scatter for identical experiments is in a 30% range. In this environment, only limited number of valid comparisons could be made, and in this section we will discuss them.

First, simulation corresponding to experiments 4 and 5 will be compared with experimental data. In Figure 3.1, measured in test 4 and simulated pressure histories are shown for the location of gauge G7. The simulated pressure closely repeats the patern of the blast wave observed in the experiment. In this case the maximum pressure levels are predicted within 10% of the experimental values. The location of first, second and fourth pressure peaks are predicted accurately. However the third peak is missing in the experimental data. Study of the pressure records at points of the computational domain adjacent to the point G7 shows that at a distance of 10cm from the point G7, the third peak disapears. So, slight inaccuracies in the alignment of the explosive charges on the ground could produce the observed discrepancies in the pressure time–history graph.

Gauges G7 and G8 are symmetrically located relative to the vertical cross section of the DECA charge, and because we were modeling only assymetric part of the array, both of these gauges will correspond only to one point in the computational domain. In Figure 3.2 we compare the pressure record of the gauge G8 obtained in test 4 with numerically simulated pressure for the same point. The character of the graphs reperesenting calculation and experiment are similar; however, simulation predicts $\approx$ 70% higher peak pressure. The difference between the simulation and pressure record obtained by gauge G8 is the same as between gauges G7 and G8, which should show the same pressure, since they located symmetricaly. One of the reasons for this discrepancy could be the obliqueness of the blast wave in

the experiment. The tests 4 and 5 was done with Iremit which has relativly low detonation velocity of 3500m/sec At that velocity, the detonation wave will reach the center of the DECA array in ≈ 0.57ms. That is comparable to ≈ 0.42ms which takes to the blast wave to travel from the ground to the location of the gauge G7 (or G8). That means that at the time the blast wave will reach the pressure transducer it will be oblique to horisontal plane at 36 deg angle. So, small deviation in the orientation of gauges G7 and G8 can lead to large differences in measured pressure. At the same time numerical solution currently do not simulate runing detonation wave (that was proposed in our next years program) and its assumes that array detonates simultaneously. As we will see below in case of 2m $\times$ 2m Primacord array the obliqueness of the blast front is smaller and the described effect would be negligable. However, we think that more experimental and computational study is needed to assess in details formation of the oblique blast fronts and its effect on the ground blast load. That subject is specialy important for design of the initiation schemes for the DECA charges. Test 5 was an exact repeat of test 4 and results of pressure measurment in this test are very similar to results of test 4. However, the large difference in pressure measured in gauges G7 and G8 reported for test 4 was also observed in test 5. That means that this discrepancy is systematic.

In test 9, 10 lines of Primacord explosive was used. Spacing between the lines was 25cm. In Figure 3.3 we compare the pressure history measured by gauge G5 in test 9 and numericaly simulated results for the same point. For that point, the results of simulation and experiment are very close, although the simulation predicted a stronger than measured rarefraction wave behind the first shock wave. In Figure 3.4, simulation and experiment are compared at the point where the G7 gauge is located. Please note that for this experiment, gauges G5 and G7 are not located in symmetrical poisitions. As is shown in Fig. 3.4, gauge G5 is located 75cm above the center of the array and gauge G7 is located at the same height above the array's edge. In Fig. 3.4, the experimentally measured maximum pressure is about 30% larger than simulated. Also, simulation predicts two shock

fronts, and the experiment measured only one front for the G7 gauge. The reasons of these discrepancies are unclear, and we point again to the inaccuracies of laying the Primacord strands and of placing the gauge. More experiments and calculations are needed in order to provide a better data base for comparision. We believe that the qualitative results of simulation for gauge G7 are correct, since at least two shock fronts should be present as a result of colisions between the blast waves from adjacent parallel lines, equidistant from the point of measurment.

In Figures 3.5 and 3.6, results for simulation and experiment are compared for the points where gauges G4 and G5 were located. Both gauges were located at a distance of 1.75m above the ground. At that distance, as a result of the multiple colisions between the shock waves, the blast wave propagates with one front. In Figures 3.5 and 3.6, results of experimentally measured pressure and calculation are in the range of expected accuracy for experiments and calculations ($\pm 15\%$).

4. Summary and Conclusions

As a result of the one–year, DARPA supported (program code 5G10/6G10), exploratory effort to investigate the Distributed Explosive Charge Array (DECA) we come to the following main conclusions:

a. Both experiments and computer simulations point to substantial advantages of the DECA concept over known mine clearing methods.

b. The computer simulations have been found to be in general agreement with the experimental data, although the accuracy of the experiments should be improved. In particular, the ringing and heating of the pressure gauges should be reduced. Future experiments should be better instrumented, in order to provide a clearer picture of the blast propagation. Also a larger experimental data base is needed to clarify some inconsistencies in pressure records (such as differences in data from gauges G7 and G8 in experiments 4 and 5). Measurments on the ground will add very important information about pecularities of the reflection of the blast wave created by DECA.

c. Computer simulation indicated that 120 lb of DECA explosive will produce targeted impulse and overpressure on the ground ($I \approx 1\text{psi} \cdot \text{sec}, P \approx 4000\text{psi}$). This impulse and pressure is sufficient for AT mine destruction, and will clear mines over a 16m^2 area. That constitutes an improvement by more than a factor of five over the closest competetive device (M–58 line charge).

d. The ability of DECA to destroy AT mines was verified experimentally. In two experiments DECA charges were suspended at 25 cm and 50 cm above the ground. In both of these experiments AT mines lying on the ground were destroyed. This shows that relatively large variations in the distance from the target did not affect substantially the ability of the charge to destroy mines. This also indicates that DECA will not need to be deployed with high precision, which will make its production cost–effective. However, the threshold at which such mines can be

destoryed needs to be determined. The effects of burying the mine below ground or the effects of reducing charge density (lower charge weight, increased charge spacing. increased charge standoff, or less energetic explosive) are all parameters that need to be investigated to establish mine lethality for near surface explosive arrays.

References

1. M. Fry, P. Kamath and D. Book, *Three-Dimensional Algoruthm Desing for Hydrodynamic Calculations*, International Symposium on Computational Fluid Dynamics, Tokyo, 1985.

2. F.H. Gregory and A.D. Gupta, *The Use of ADINA for Analysis of Mines with Explosive Fills*, Computers & Structures, Vol 17, No. 5–6, pp. 625–633, 1983.

3. C.M. Romander, *Measurment of Airblast and Responce of Antitank Mines from a Planar Explosive Array*, Iterim Report, SRI–International, May 1986.

Figures and Table Captions

Figure 1.1 Schematic layout of a typical simulation.

Figure 1.2a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.2d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Z cross section.

Figure 1.3a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.3d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X–Y cross section.

Figure 1.4a Impulse on the ground at $t = 0.25$msec.

Figure 1.4b Impulse on the ground at $t = 0.50$msec.

Figure 1.4c Impulse on the ground at $t = 1.00$msec.

Figure 1.5a Time history of pressure, impulse, temperature and density on the ground under the explosive.

Figure 1.5b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

Figure 1.6 Maximum pressure and impulse distribution on the ground for 10 lines (116lb) of Iremite.

Figure 1.7a Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Z cross section.

Figure 1.7b Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Z cross section.

Figure 1.8a Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Y cross section.

Figure 1.8b Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Y cross section.

Figure 1.9a Time history of pressure, impulse, temperature and density on the ground under a line of explosive.

Figure 1.9b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

Figure 1.10a Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground. X=0cm, Y=0cm, Z=0cm.

Figure 1.10b Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground. X=10cm, Y=0cm, Z=0cm

Figure 1.10c Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground. X=25cm, Y=0cm, Z=0cm

Figure 1.10d Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground. X=40cm, Y=0cm, Z=0cm.

Figure 1.11 Pressure contours for 10 lines ($2.25m \times 2.0m$) Primacord detonation. X–Z cross section.

Figure 1.12 Pressure contours for 10 lines ($2.25m \times 2.0m$) Primacord detonation. X–Y cross section.

Figure 1.13a Time history for pressure, impulse, total pressure and density on the ground. X=0, Y=0, Z=0

Figure 1.13b Time history for pressure, impulse, total pressure and density on the ground. X=5, Y=0, Z=0

b . Computer simulation.

Figure 3.5 Comparison of data from gauge G4 (test 9) and simulation.

a . Experimental data

b . Computer simulation.

Figure 3.6 Comparison of data from gauge G5 (test 9) and simulation.

a . Experimental data

b . Computer simulation.

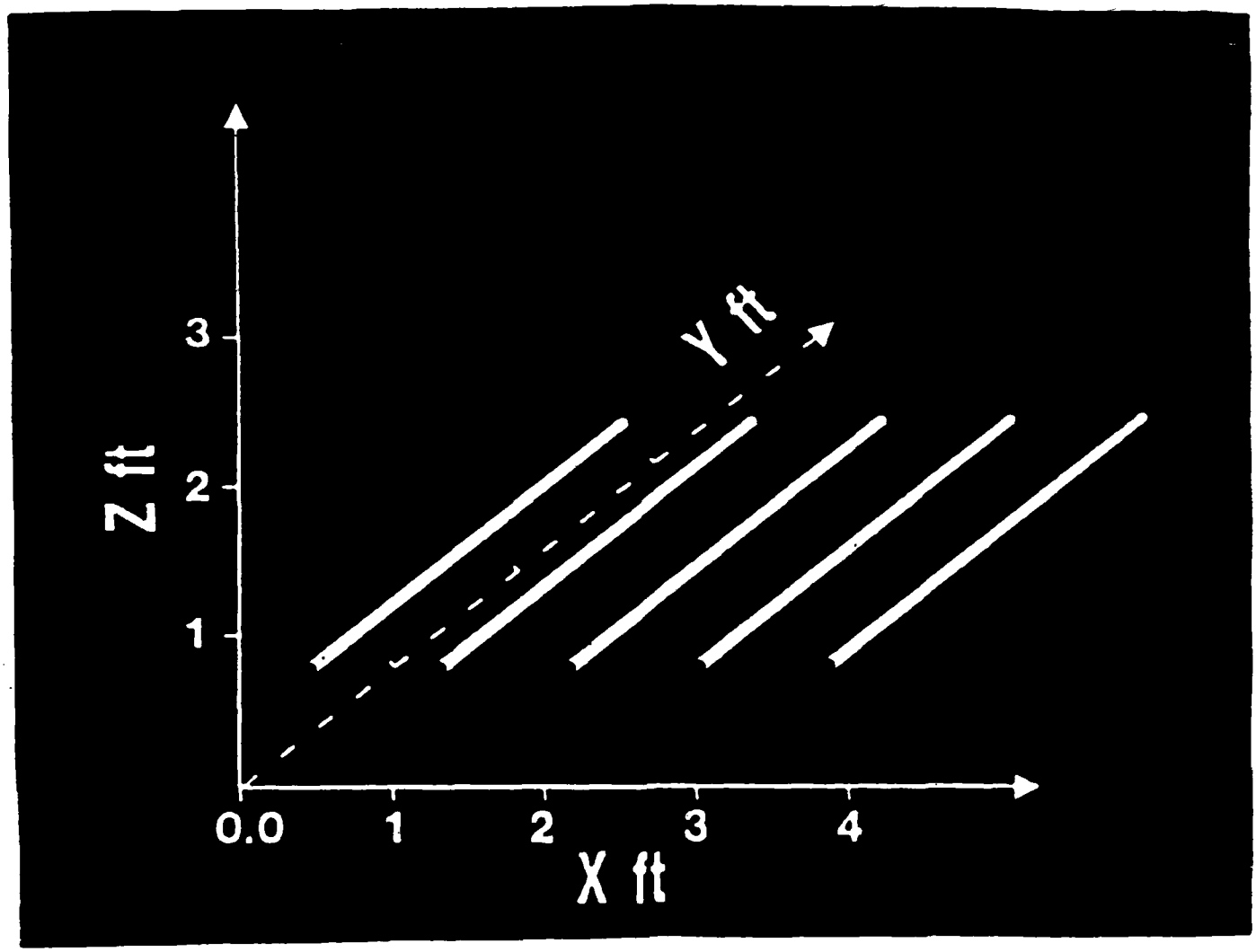


Figure 1.1 Schematic layout of a typical simulation.

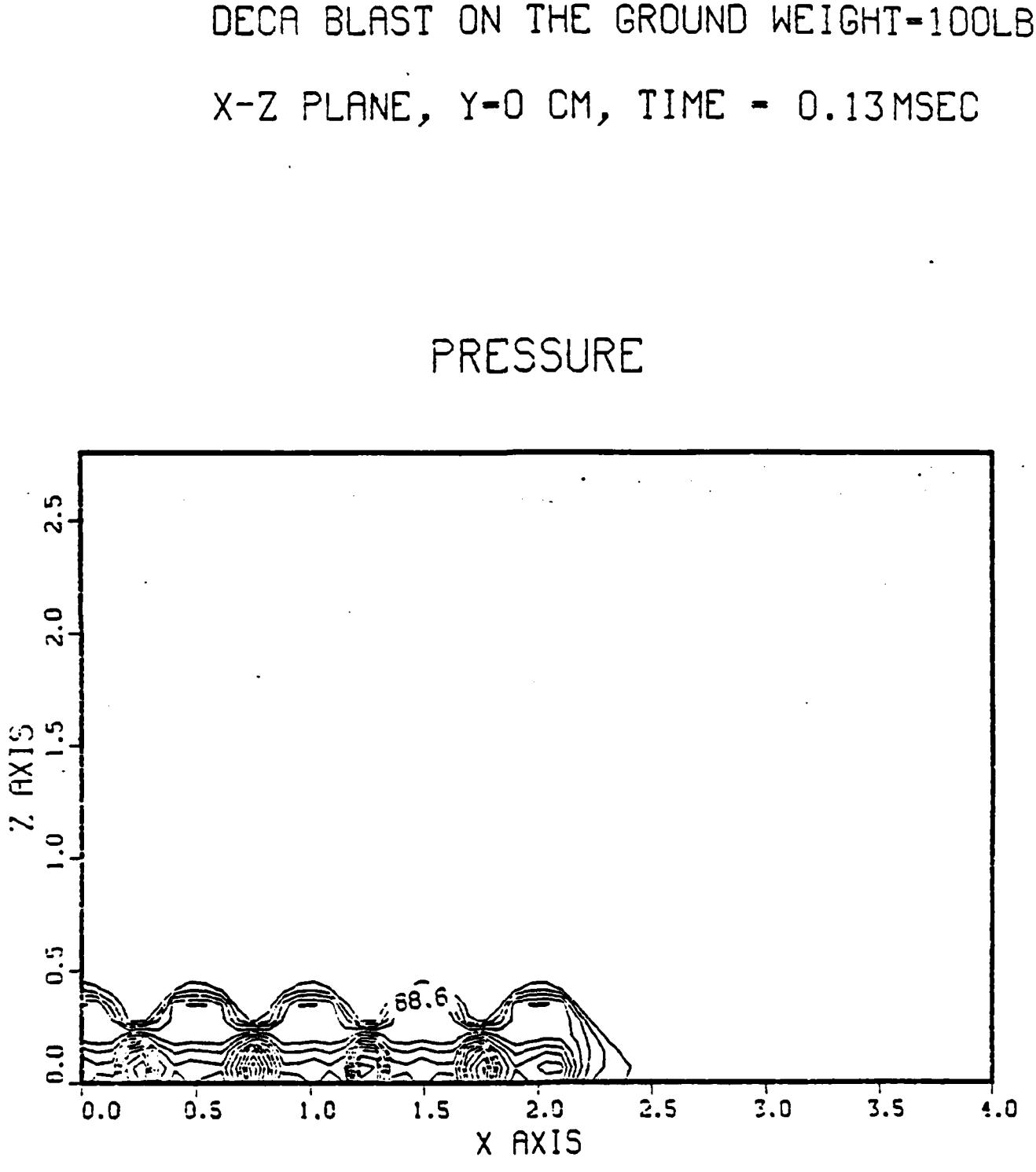


Figure 1.2a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Z cross section.

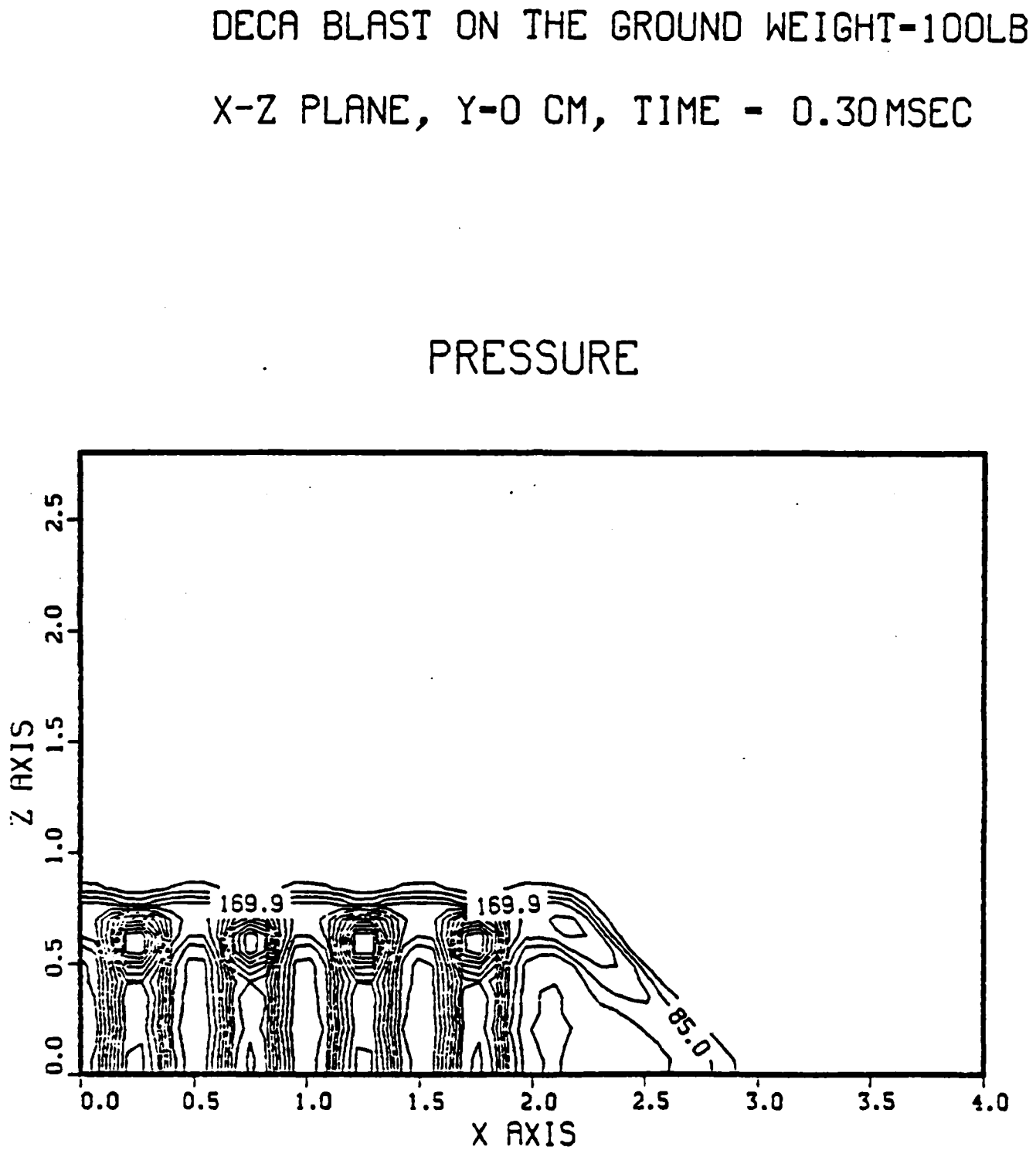


Figure 1.2b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Z cross section.

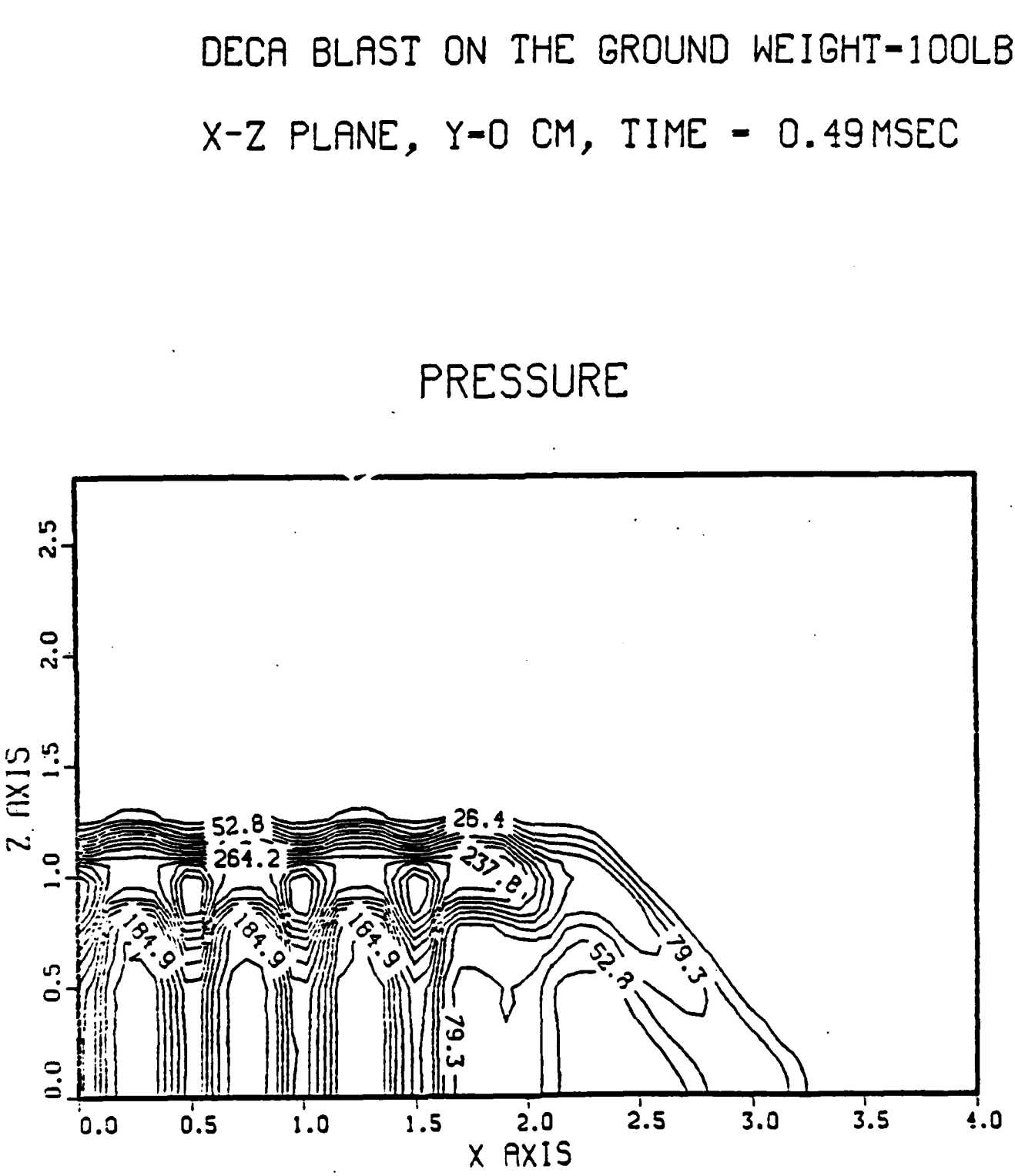


Figure 1.2c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Z cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB

X-Z PLANE, Y=0 CM, TIME = 0.72MSEC

PRESSURE

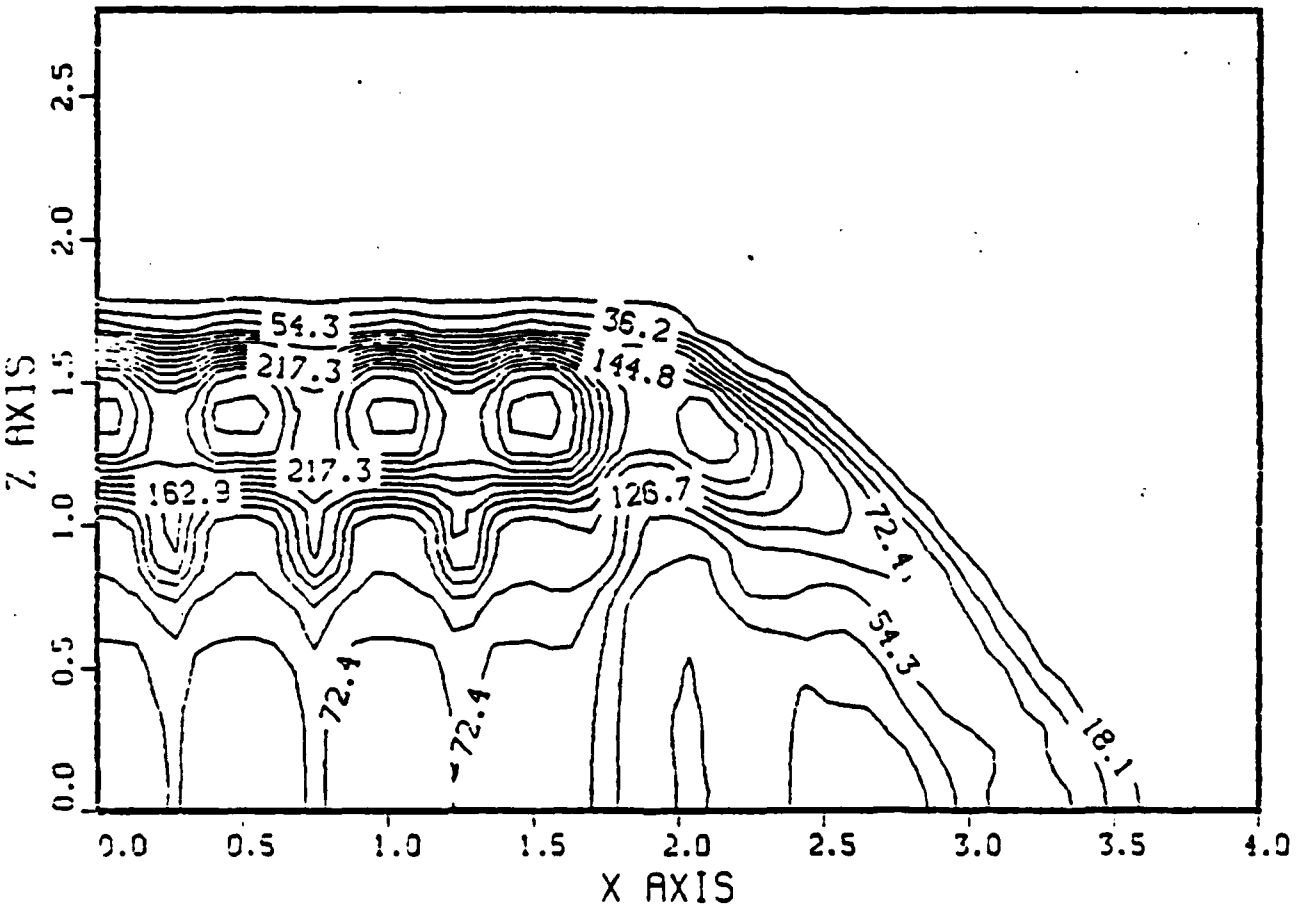


Figure 1.2d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Z cross section.

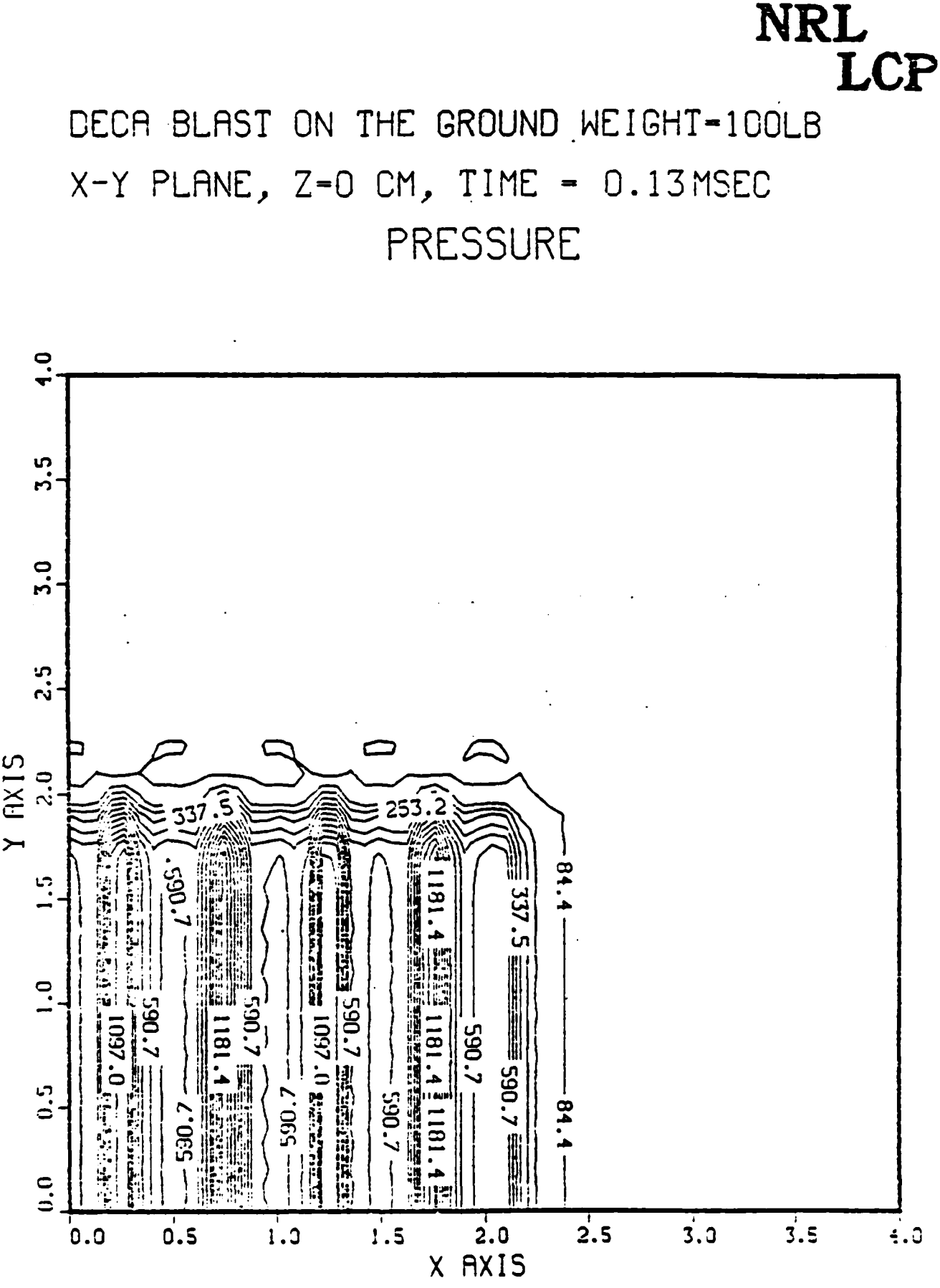


Figure 1.3a Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

NRL
LCP

DECA BLAST ON THE GROUND WEIGHT=100LB

X-Y PLANE, Z=0 CM, TIME = 0.30MSEC

PRESSURE

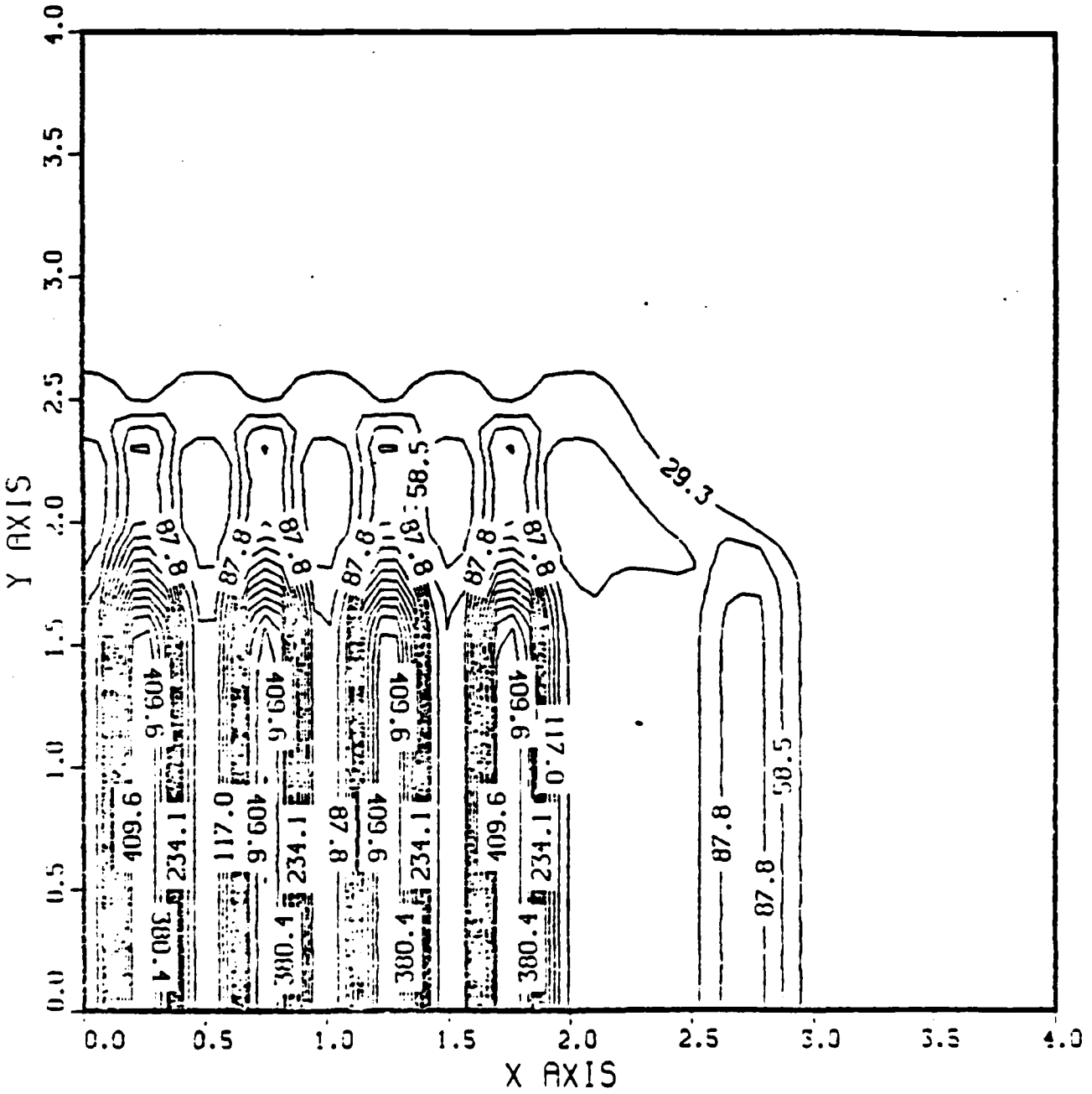


Figure 1.3b Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

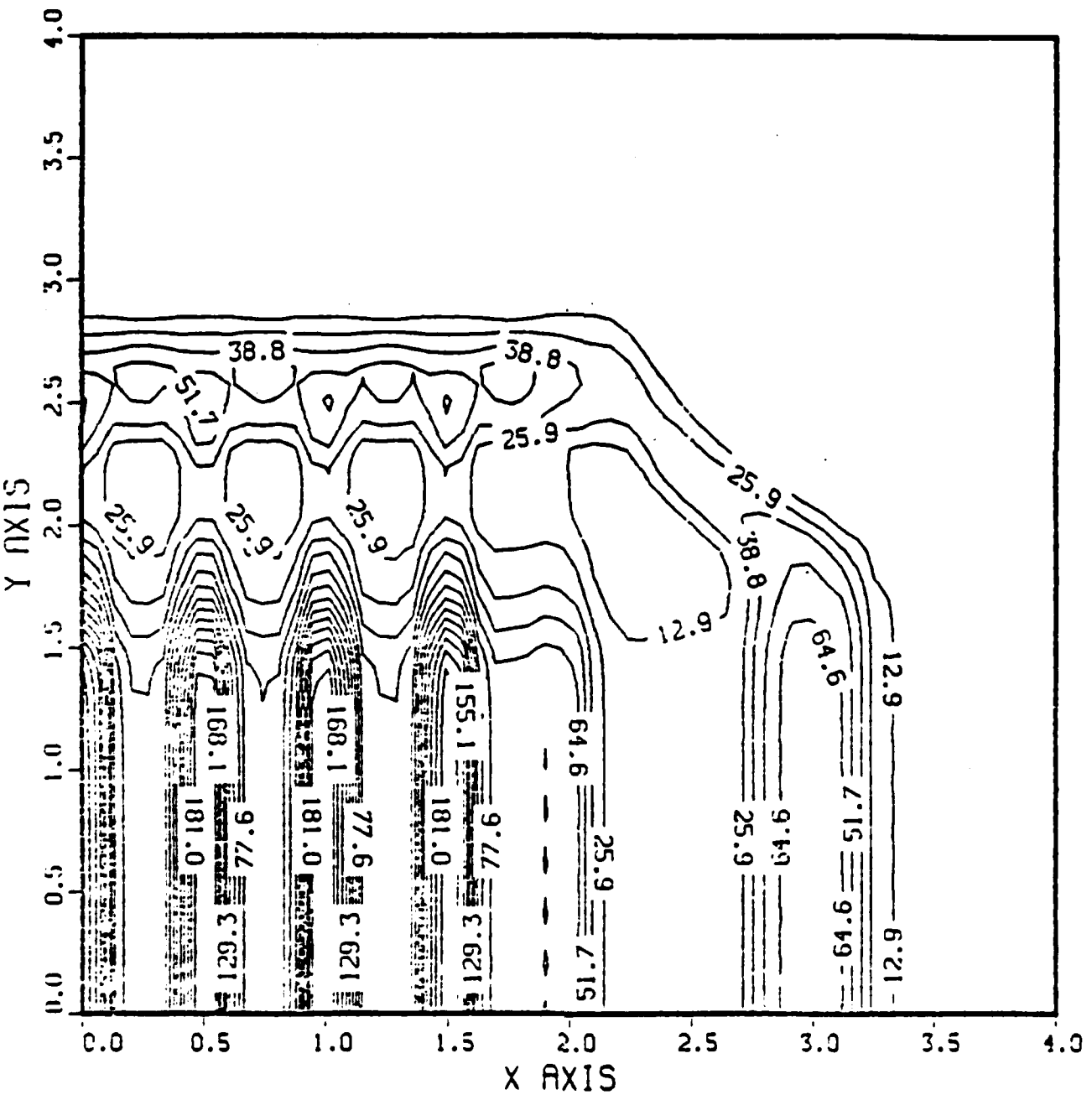


Figure 1.3c Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

NRL
LCP

DECA BLAST ON THE GROUND WEIGHT=100LB

X-Y PLANE, Z=0 CM, TIME = 0.72MSEC

PRESSURE

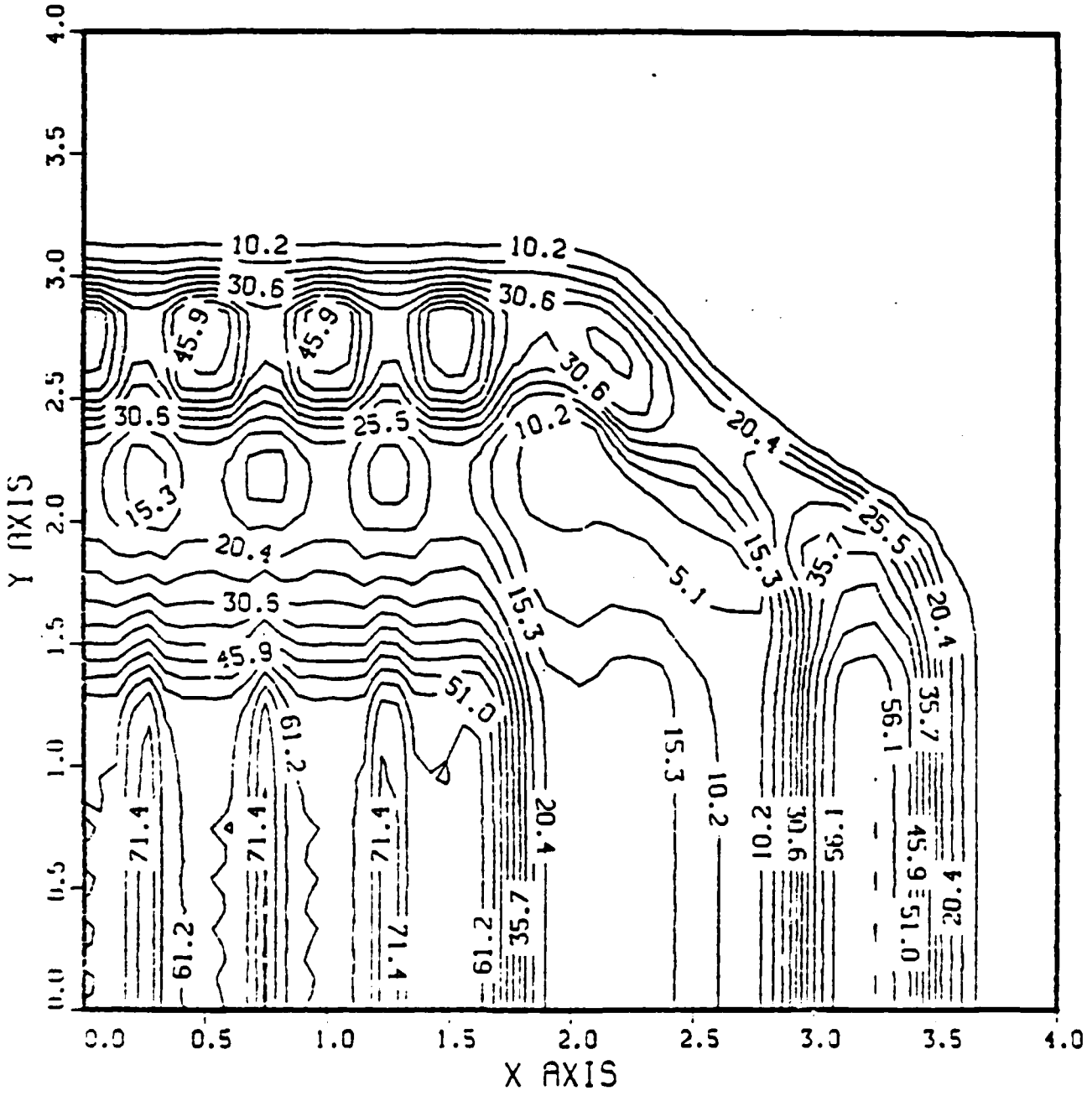


Figure 1.3d Pressure contours for a 9 lines $4m \times 4m$ DECA array test. X-Y cross section.

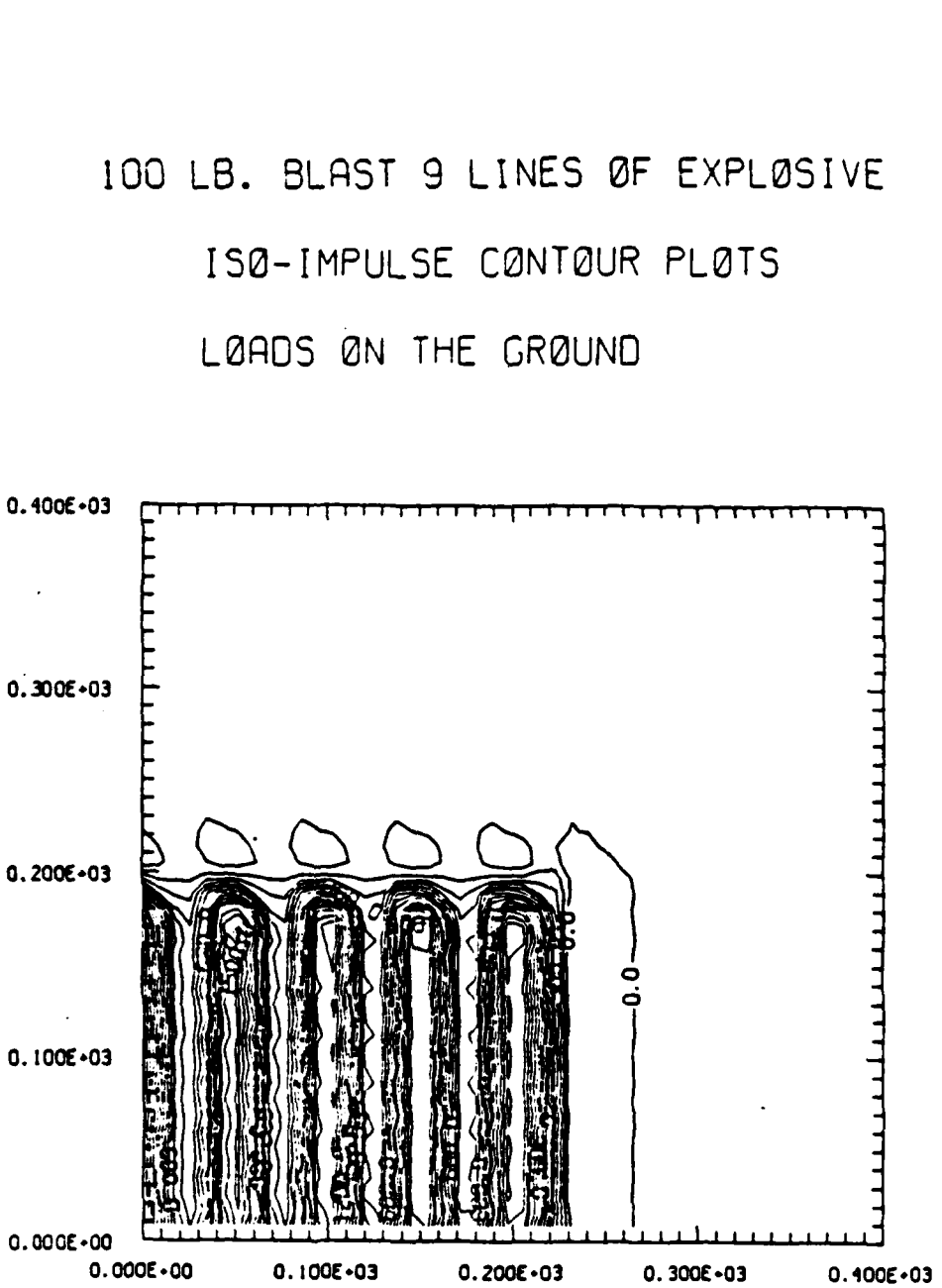


Figure 1.4a Impulse on the ground at $t = 0.25$msec.

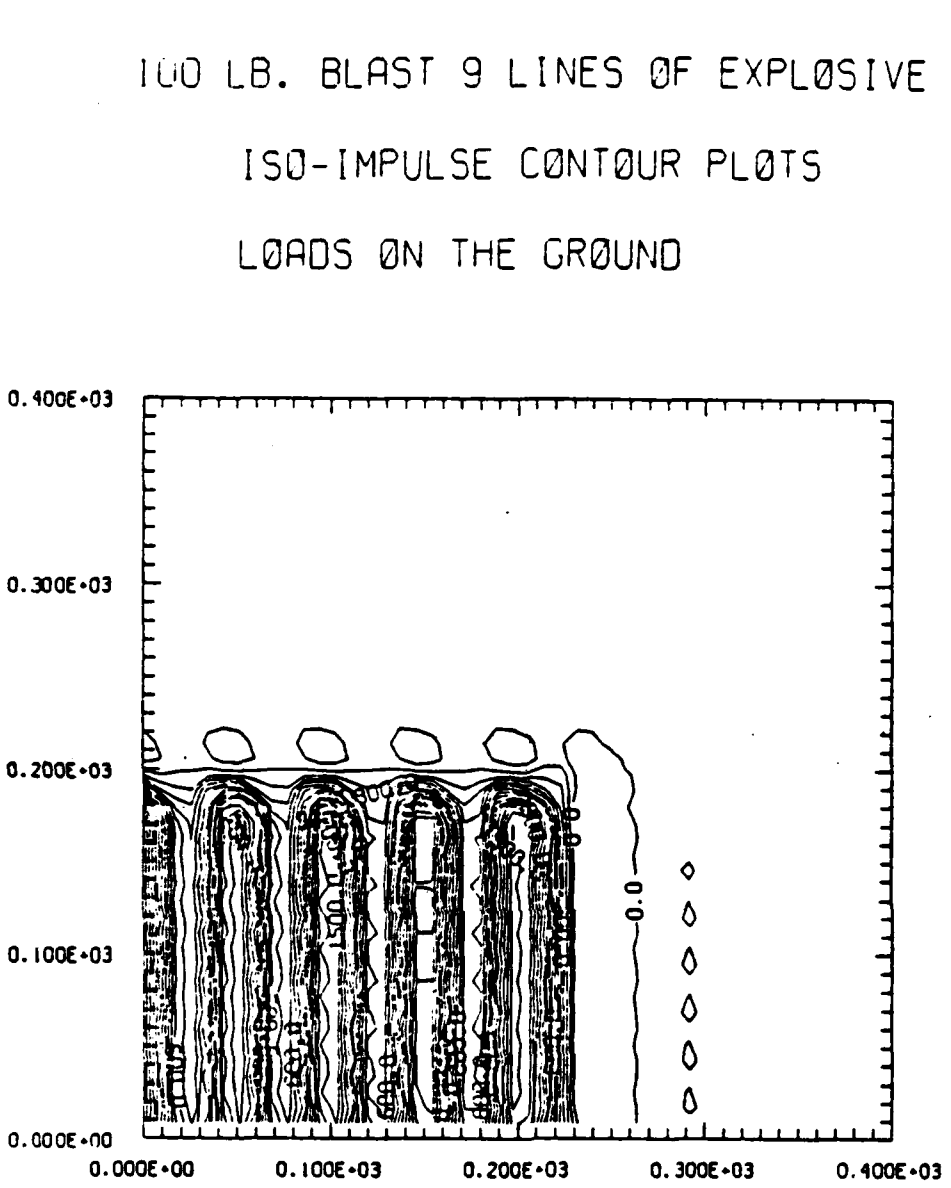


Figure 1.4b Impulse on the ground at $t = 0.50$msec.

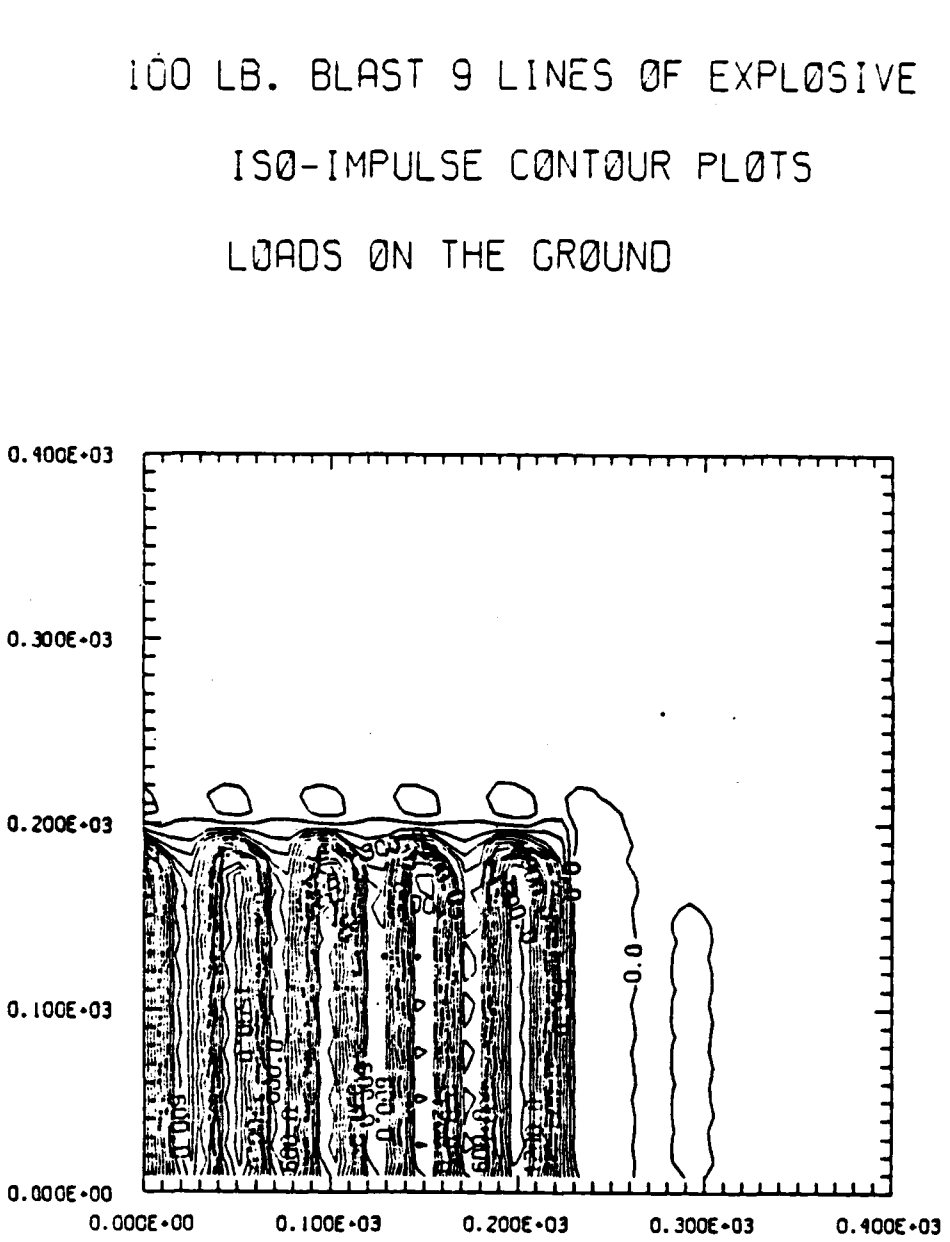


Figure 1.4c Impulse on the ground at $t = 1.00$msec.

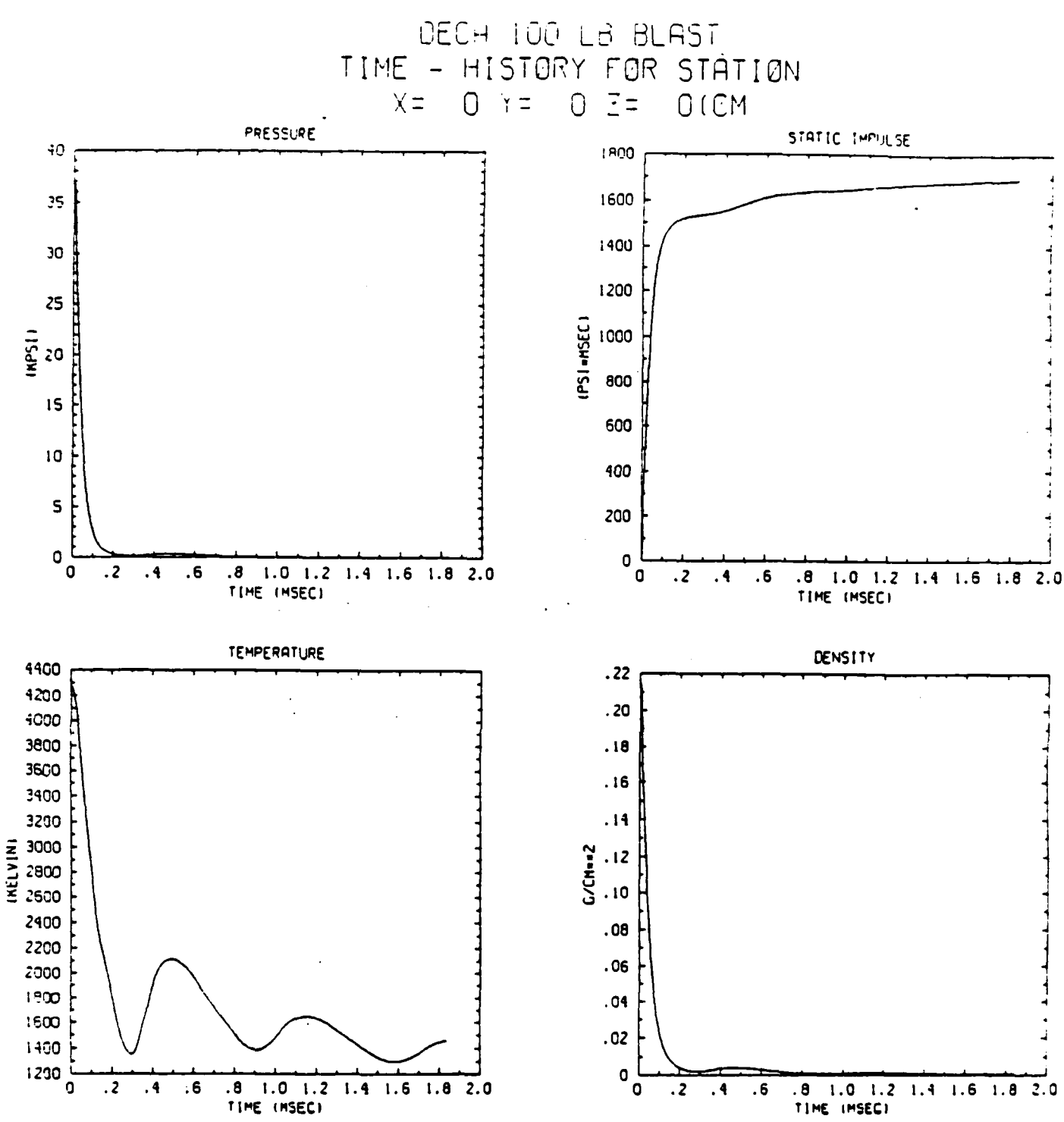


Figure 1.5a Time history of pressure, impulse, temperature and density on the ground under the explosive.

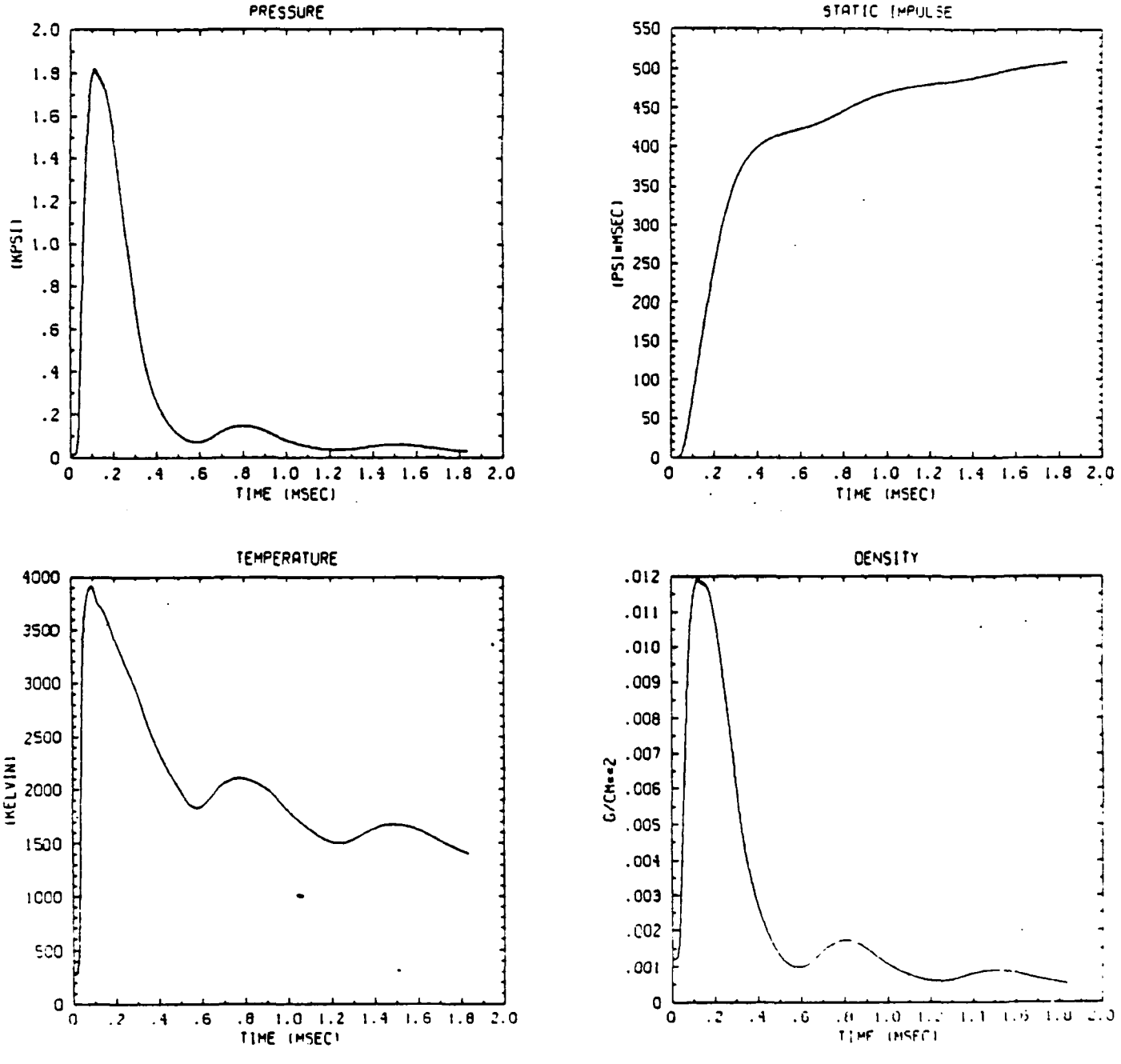


Figure 1.5b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

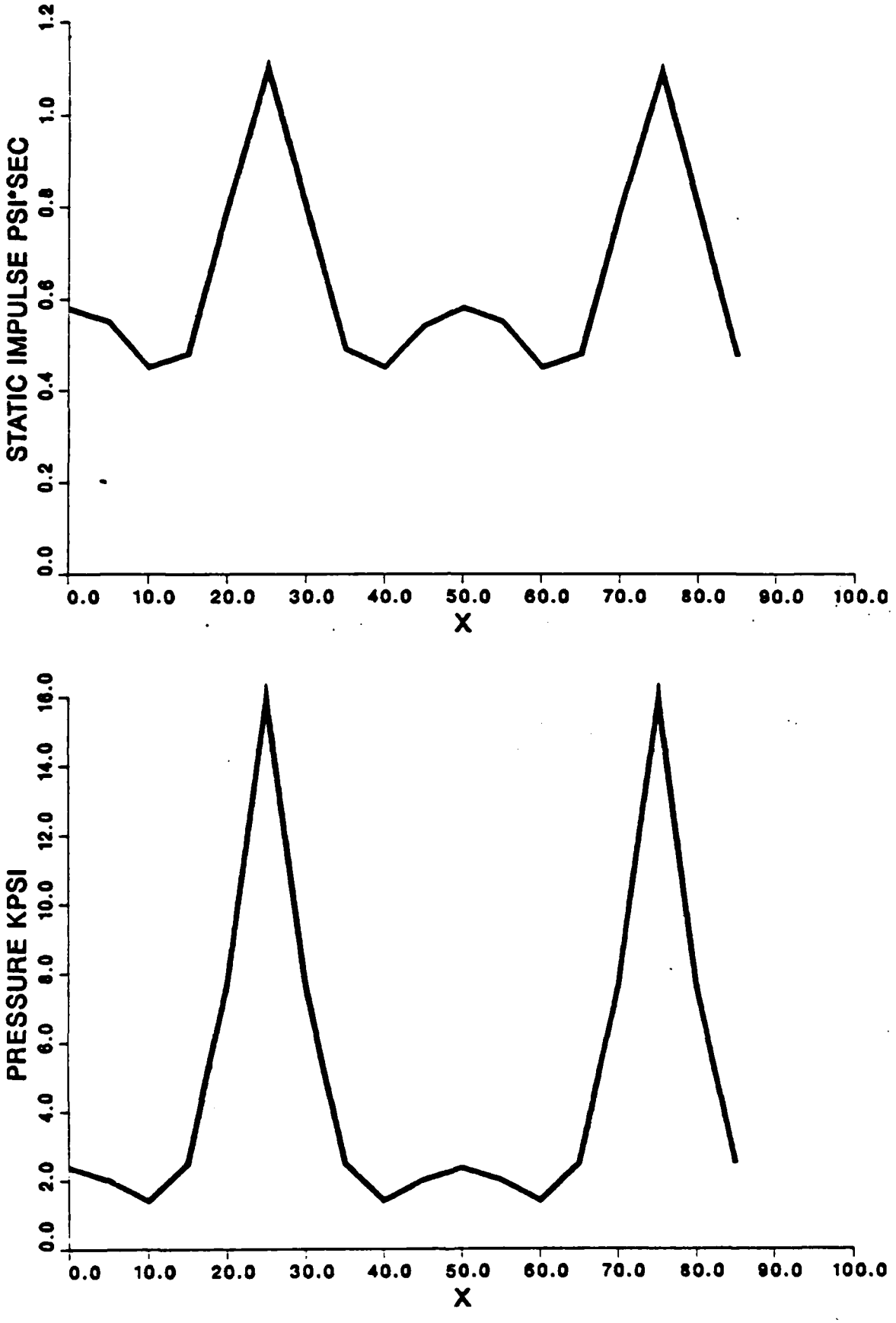


Figure 1.6 Maximum pressure and impulse distribution on the ground for 10 lines (1163) of Iremite.

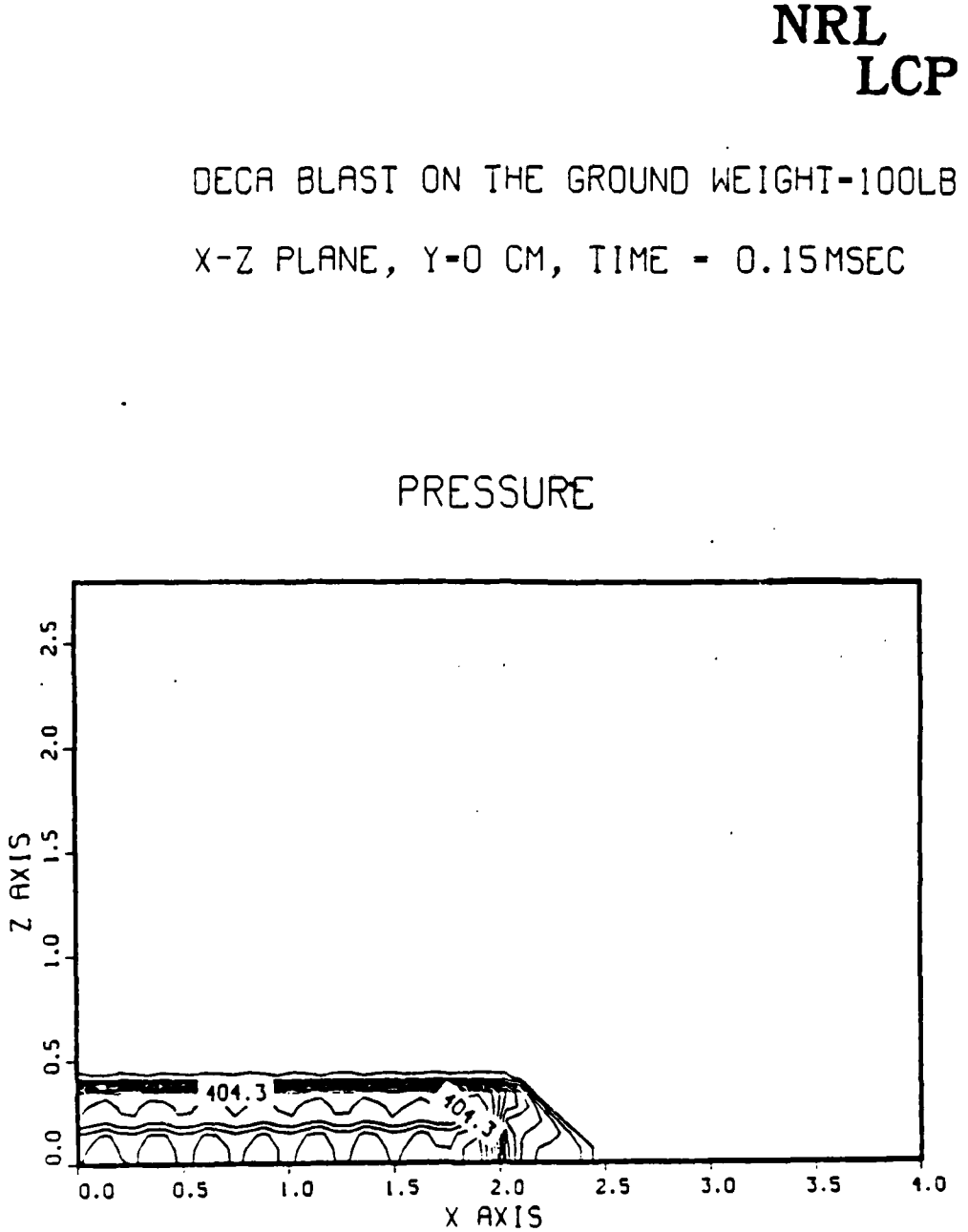


Figure 1.7a Pressure contours for 17 lines $4m \times 4m$ DECA array. X–Z cross section.

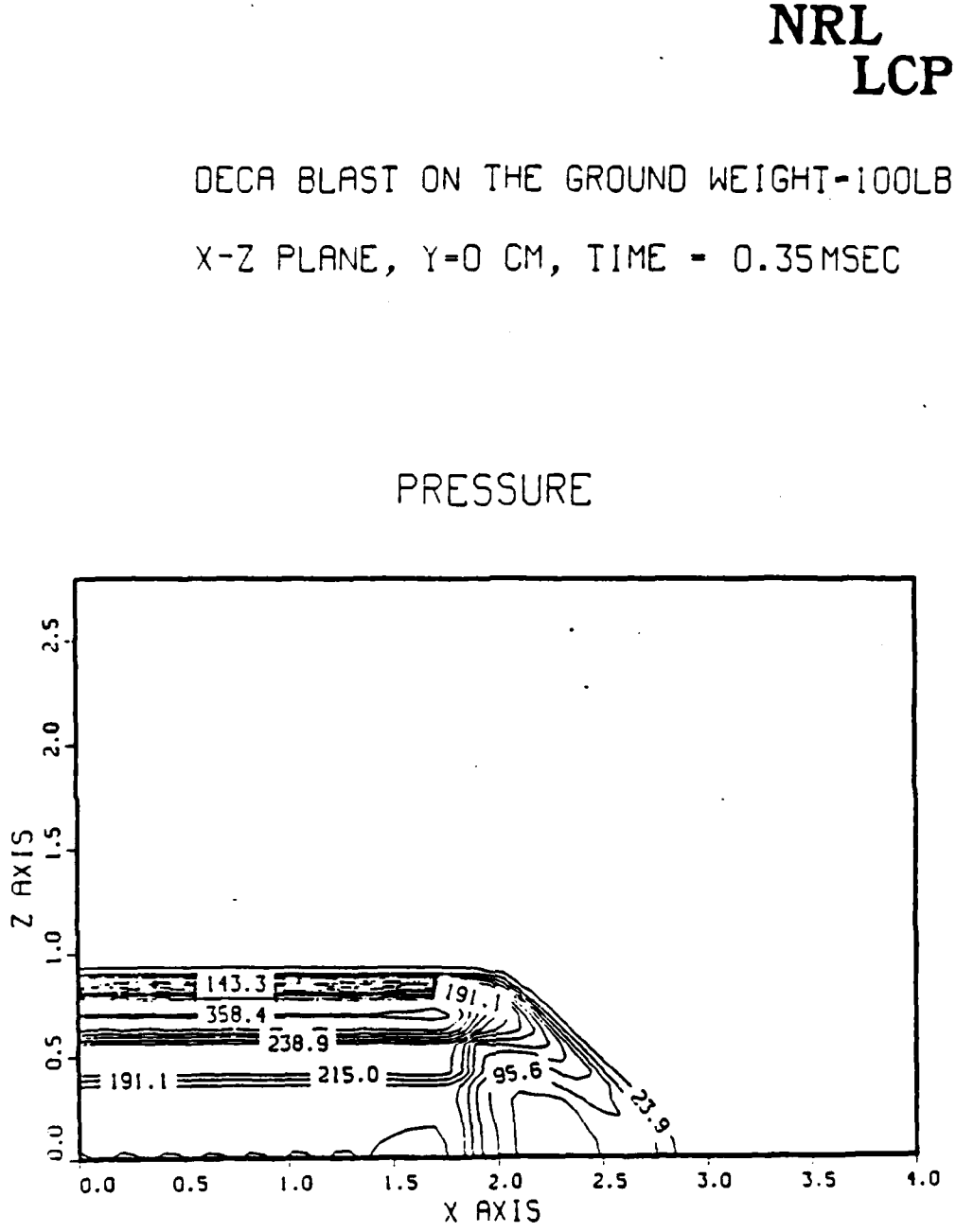


Figure 1.7b Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Z cross section.

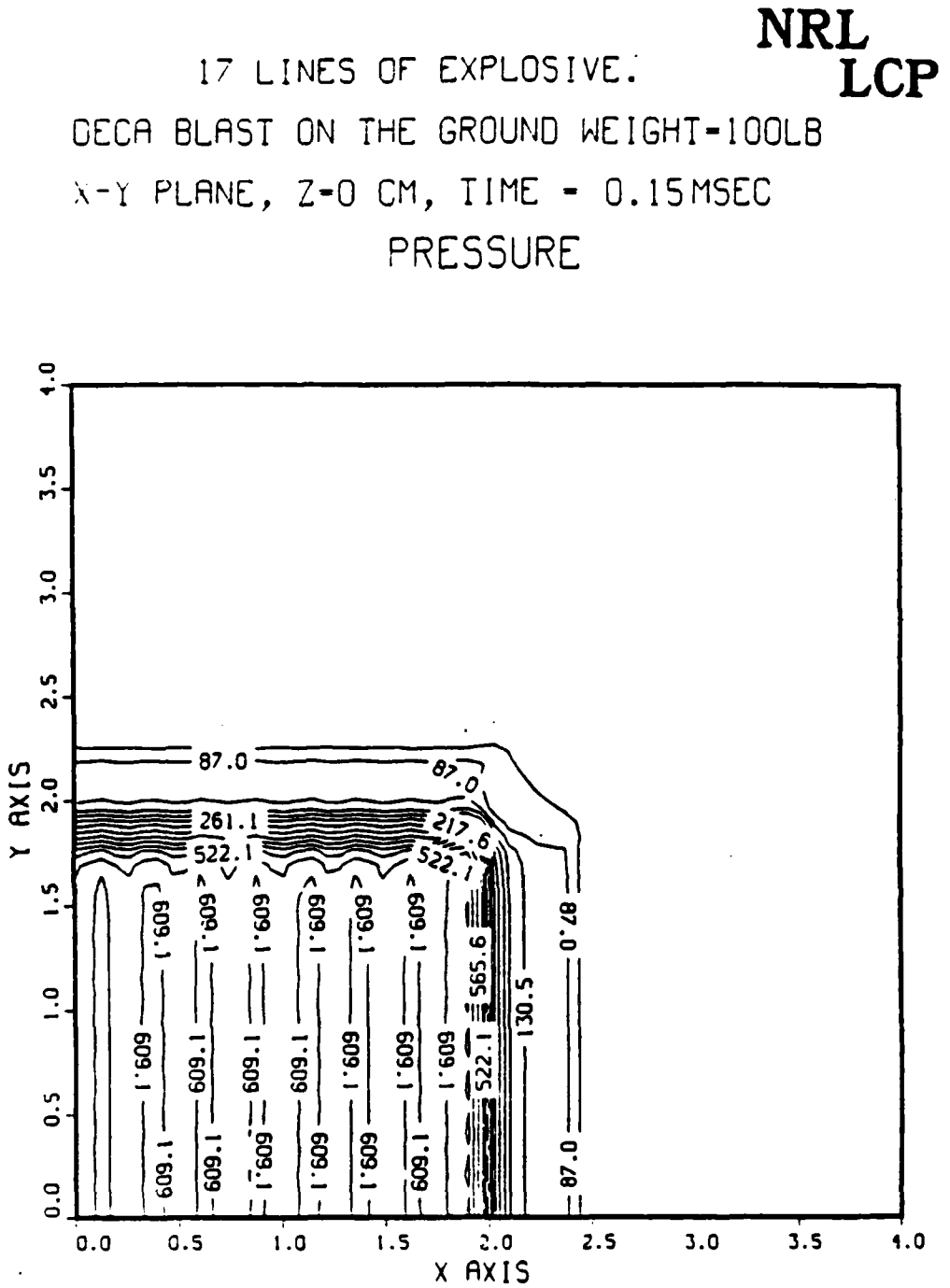


Figure 1.8a Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Y cross section.

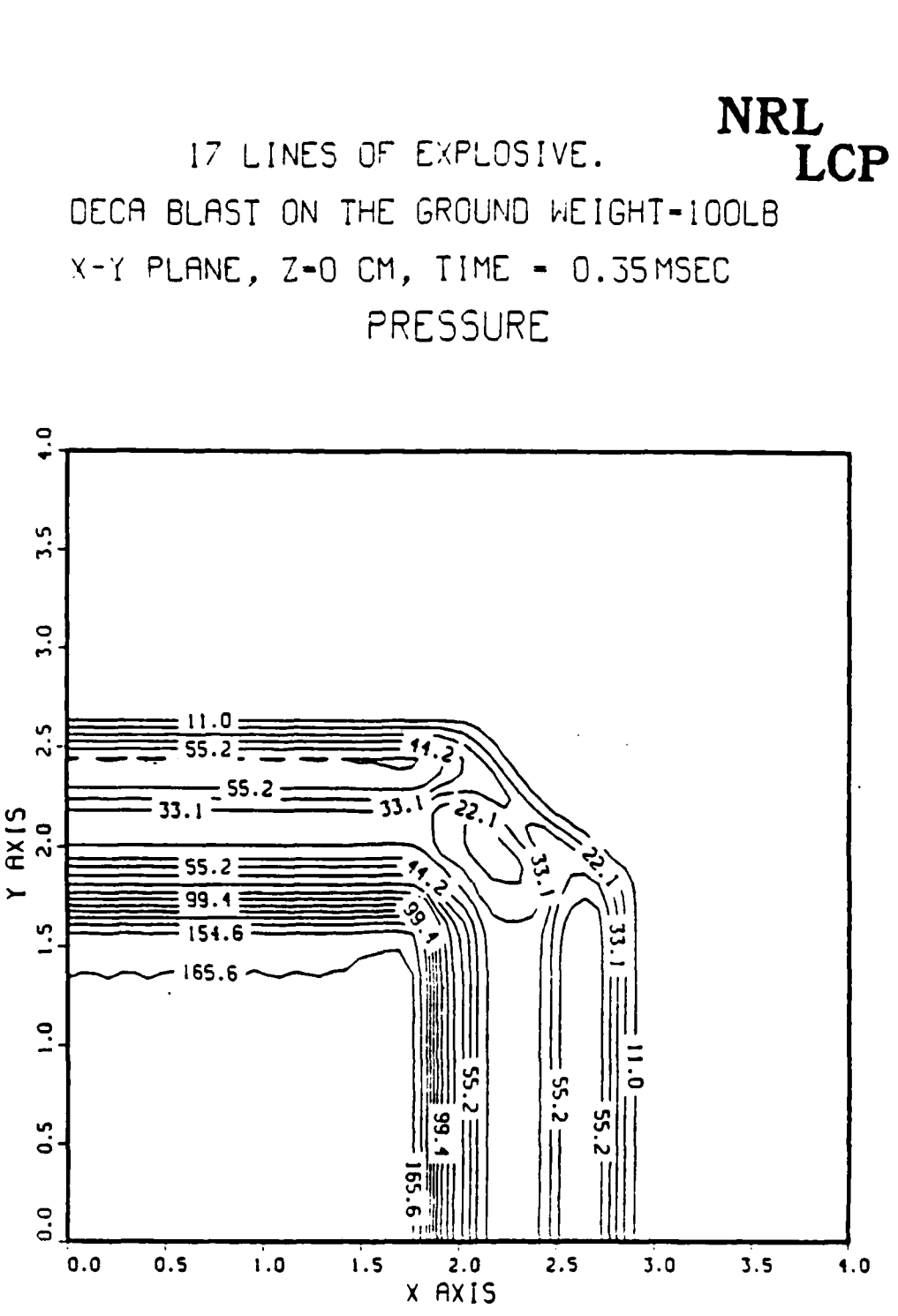


Figure 1.8b Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Y cross section.

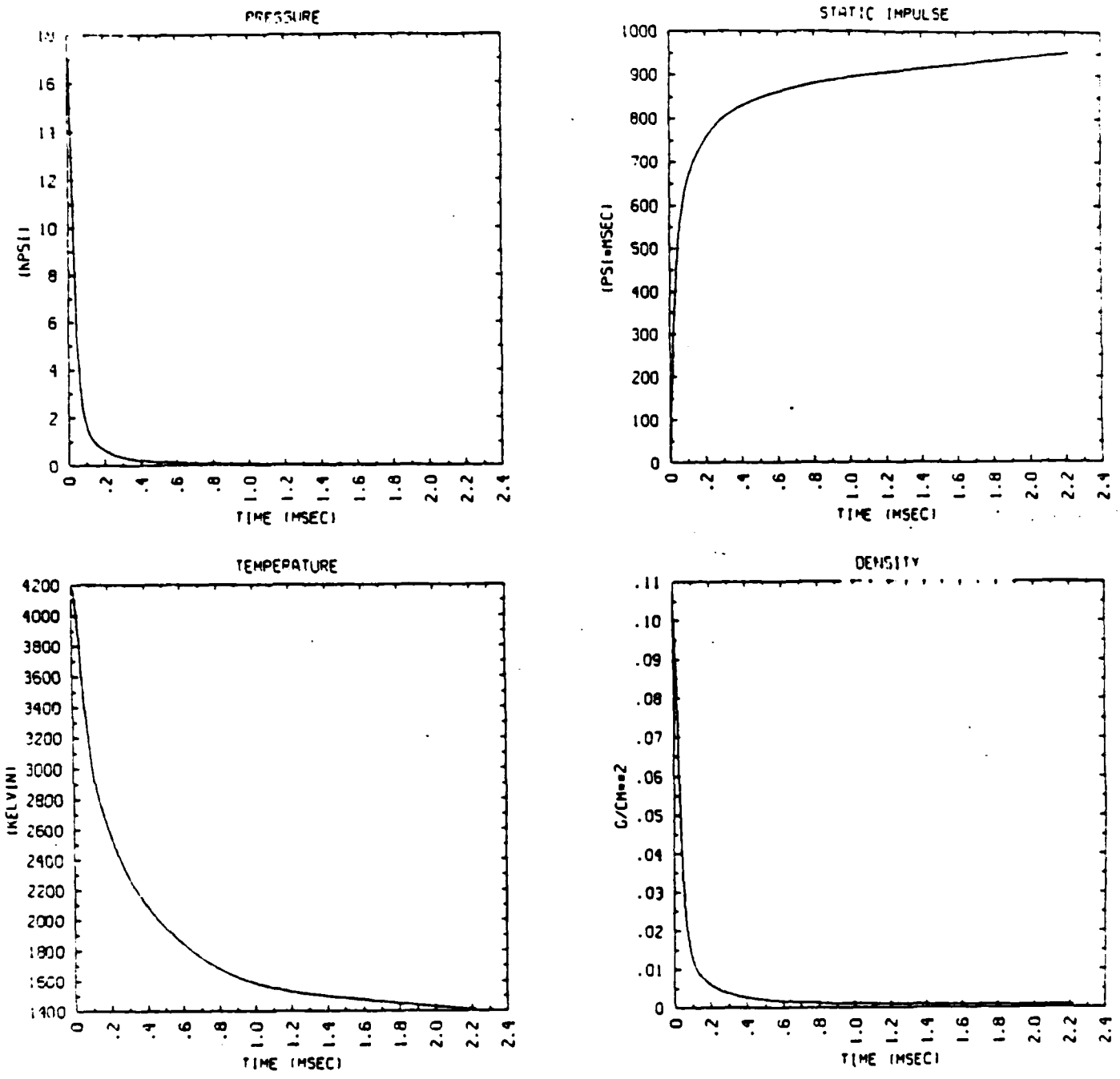


Figure 1.9a Time history of pressure, impulse, temperature and density on the ground under a line of explosive.

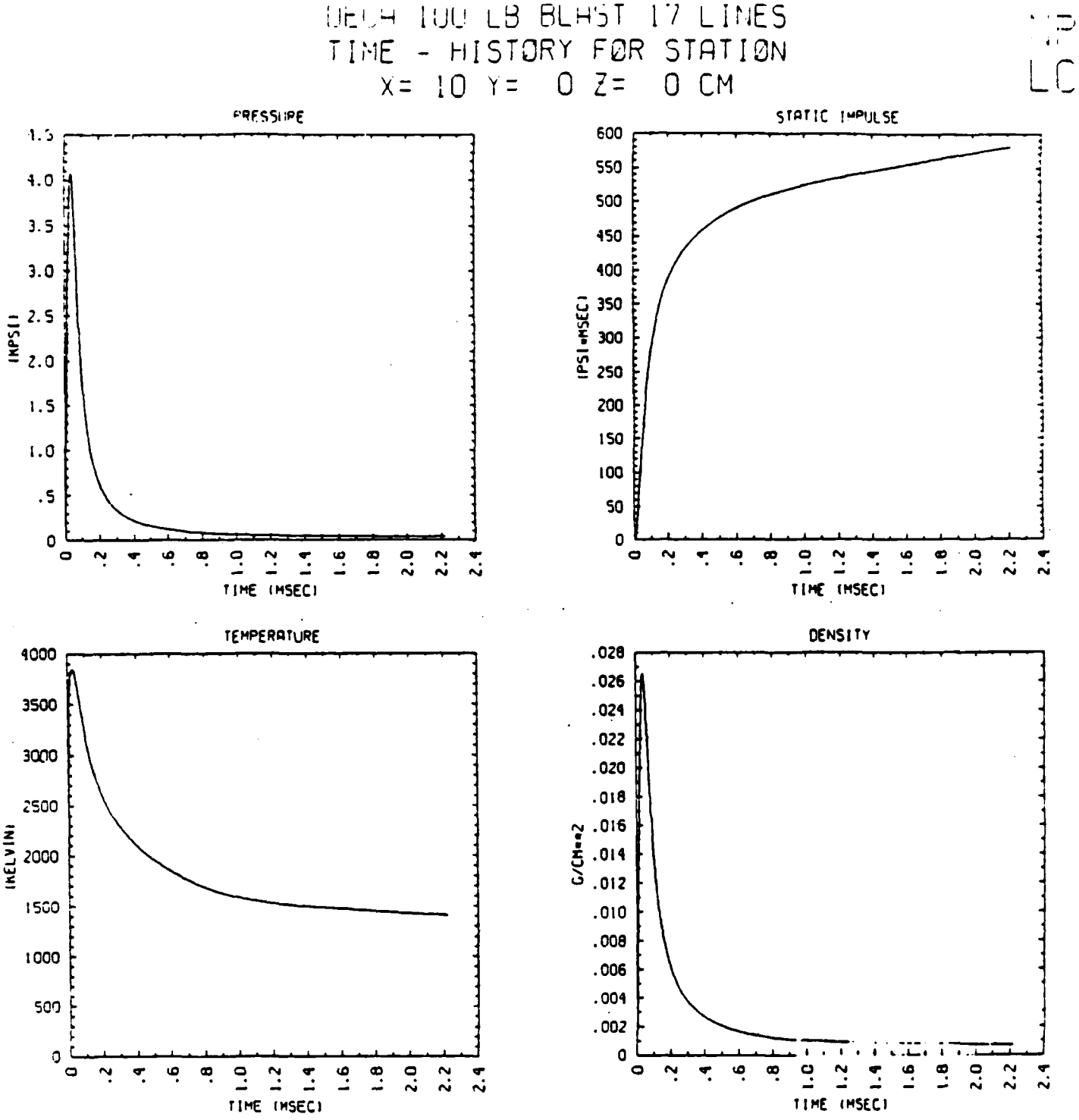


Figure 1.9b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

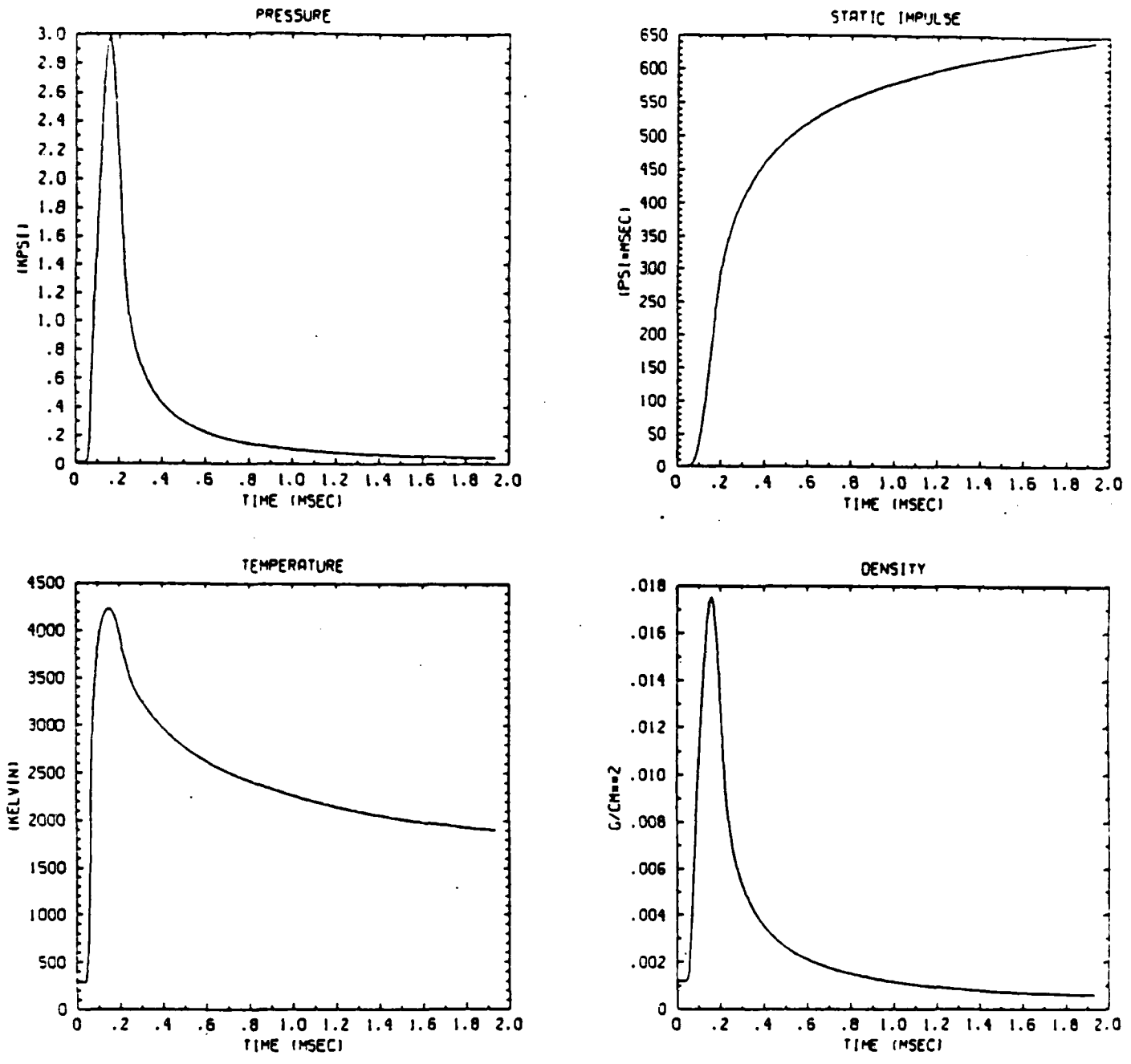


Figure 1.10a Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

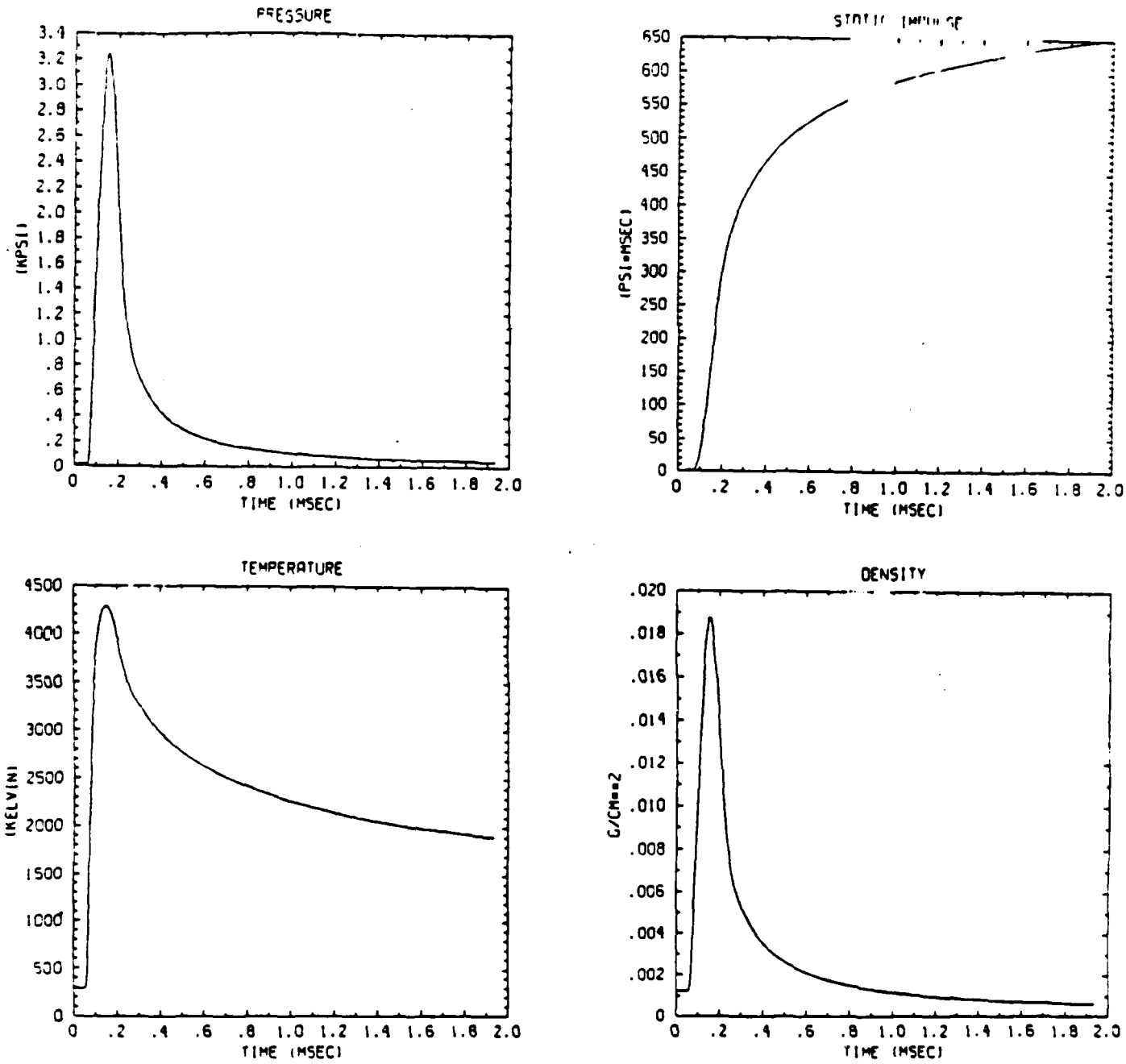


Figure 1.10b Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

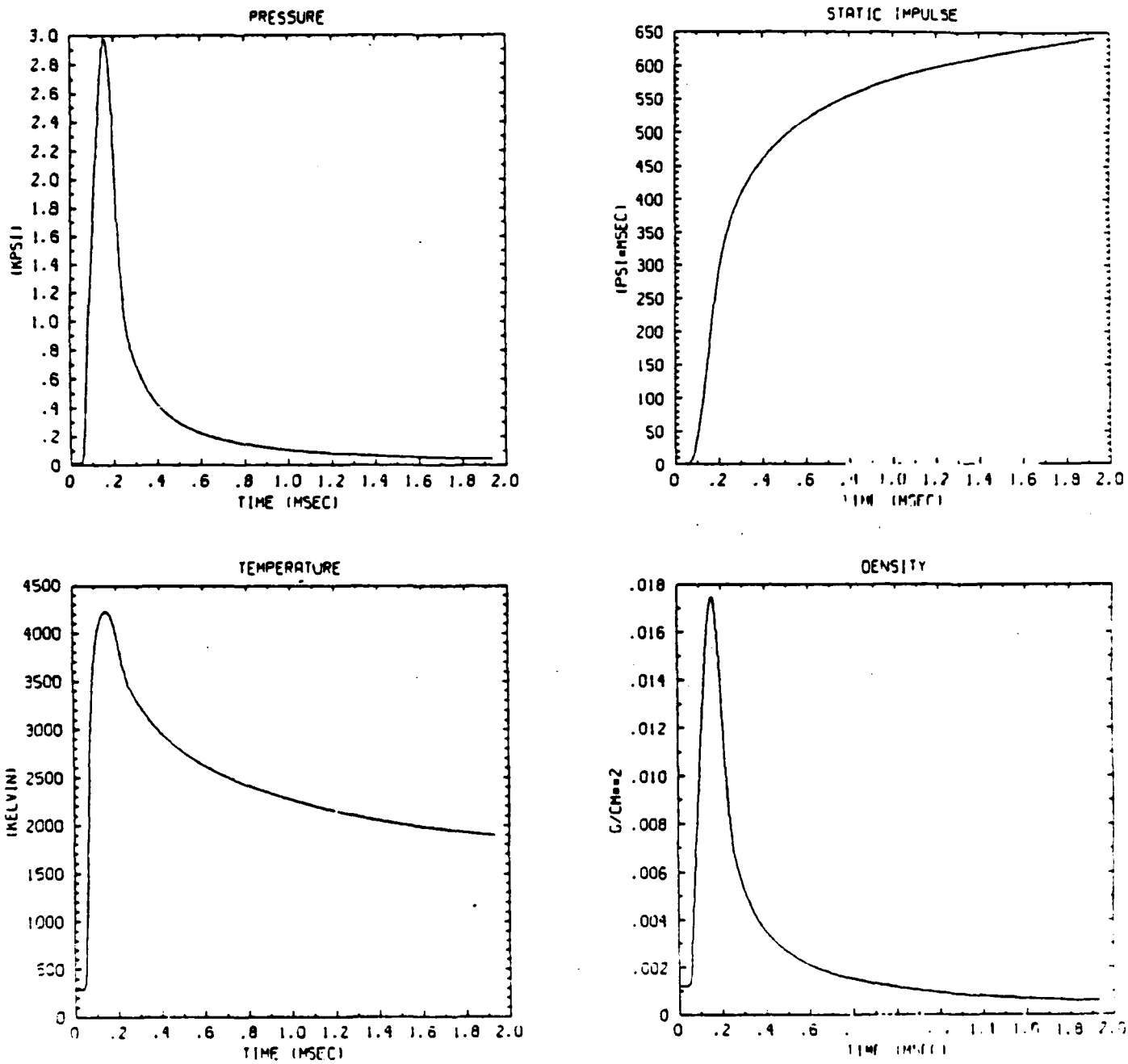


Figure 1.10c Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

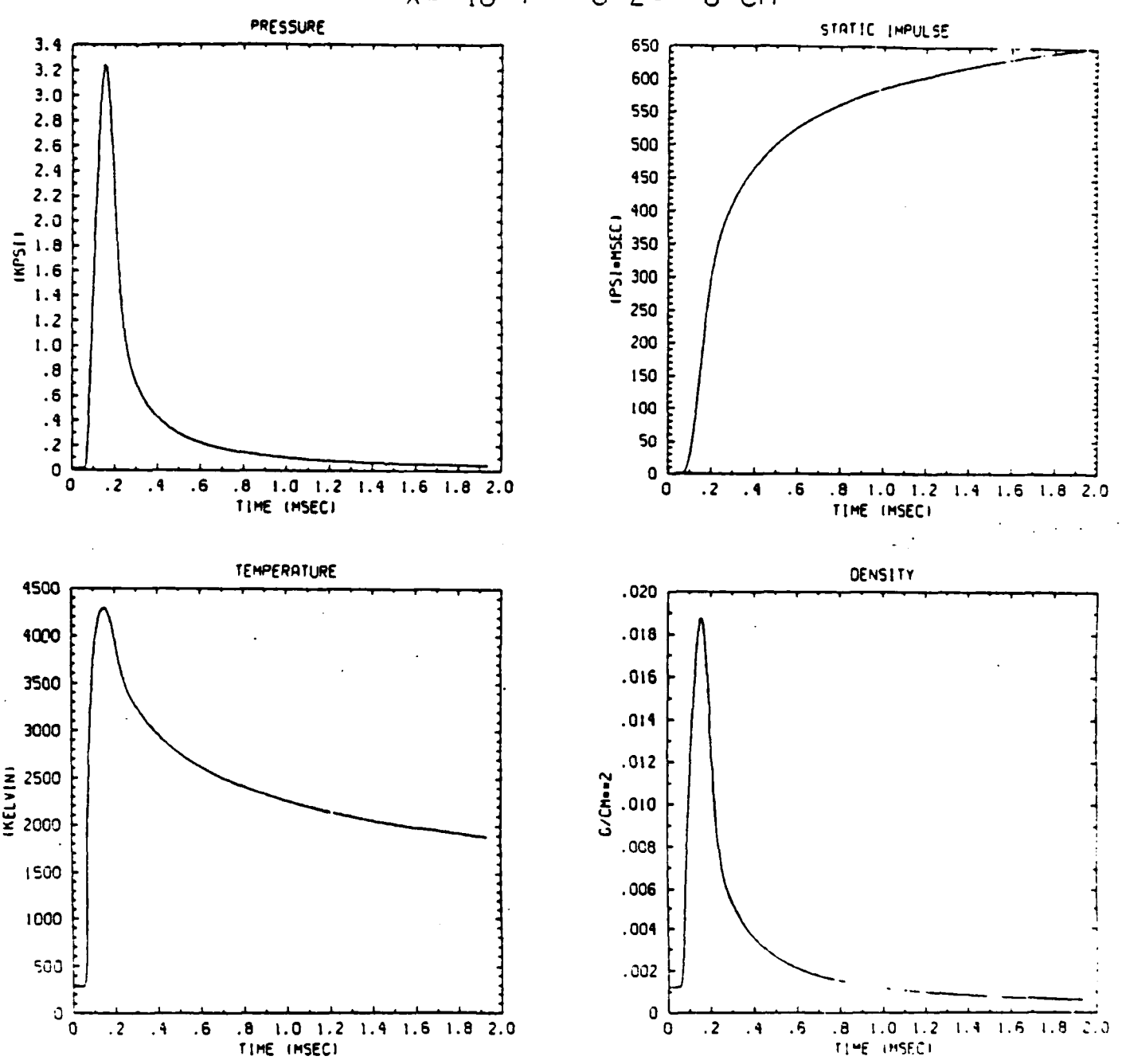


Figure 1.10d Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

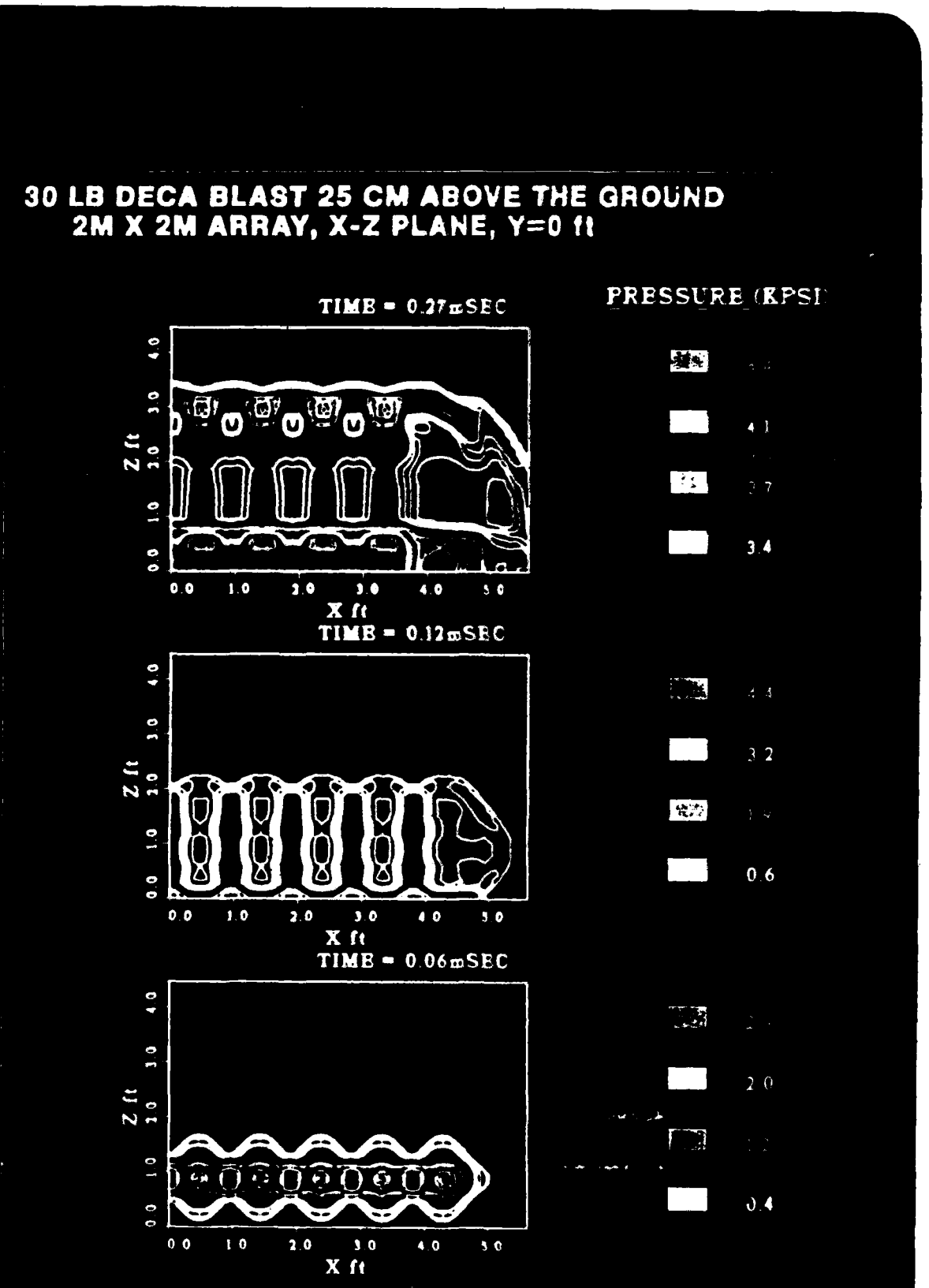


Figure 1.11 Pressure contours for 10 lines (2.25m x 2.0m) Primacord detonation.

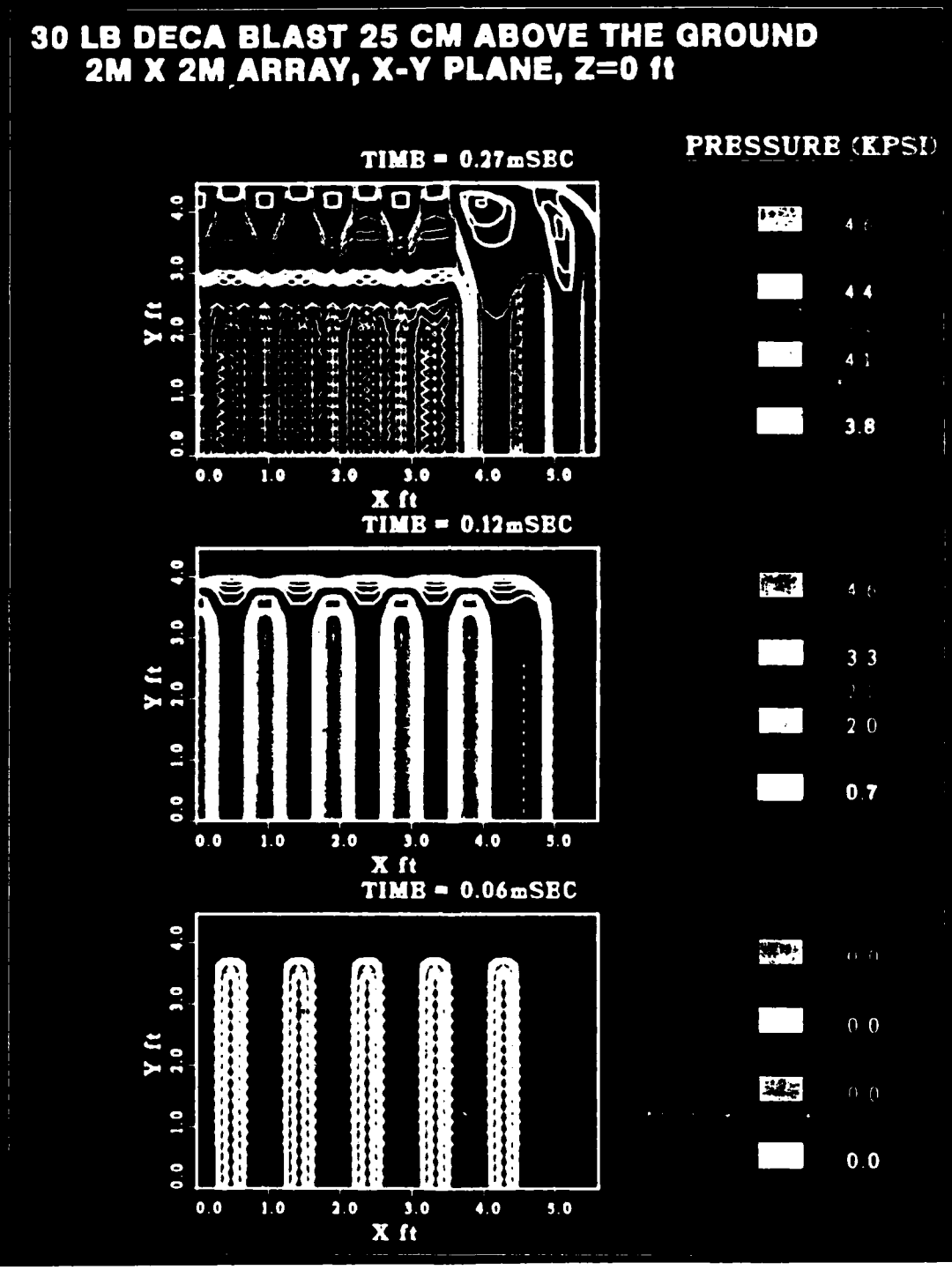


Figure 1.12 Pressure contours for 10 lines ($2.25m \times 2.0m$) Primacord detonation.

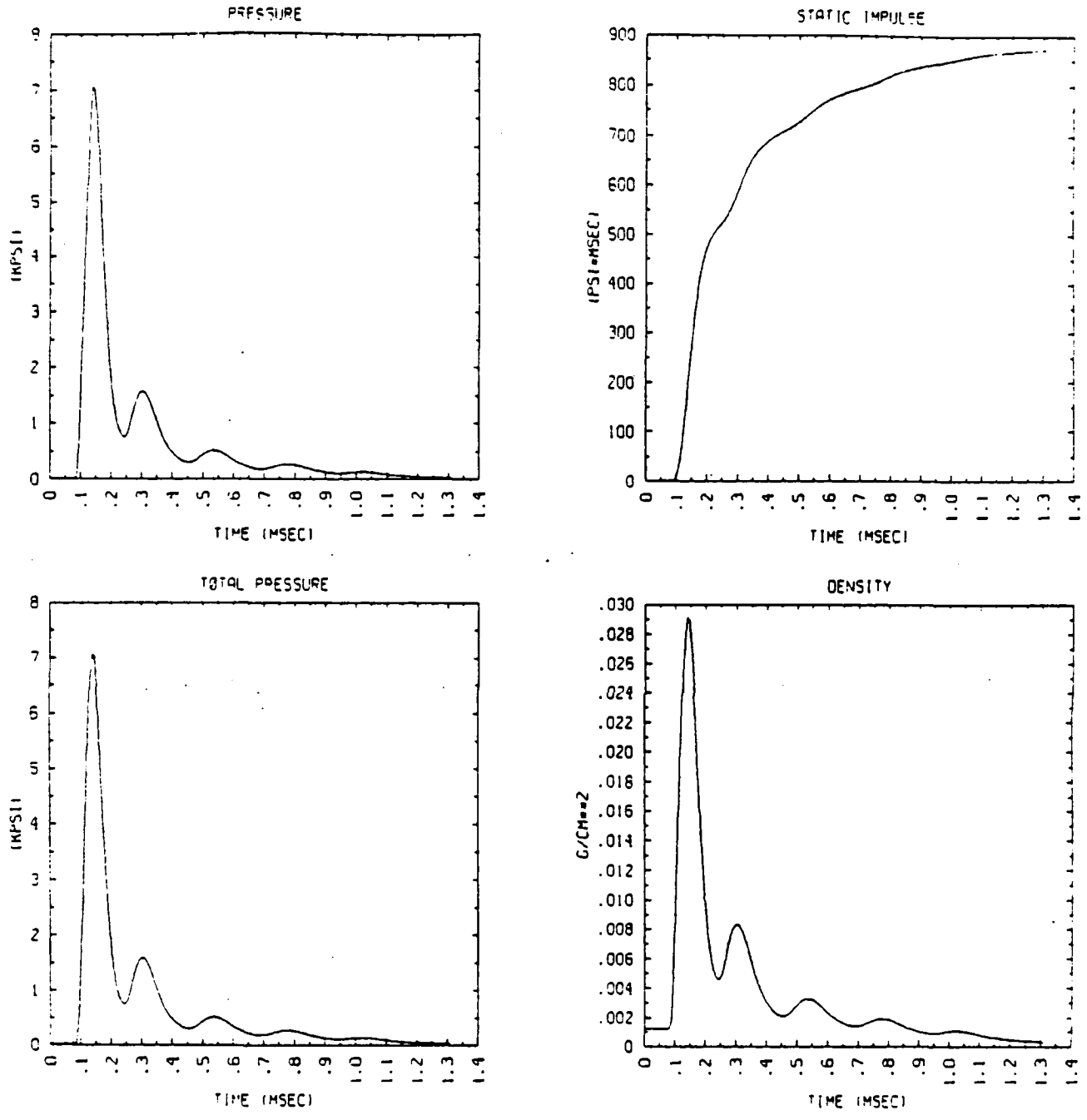


Figure 1.13a Time history for pressure, impulse, total pressure and density on the ground. X=0, Y=0, Z=0

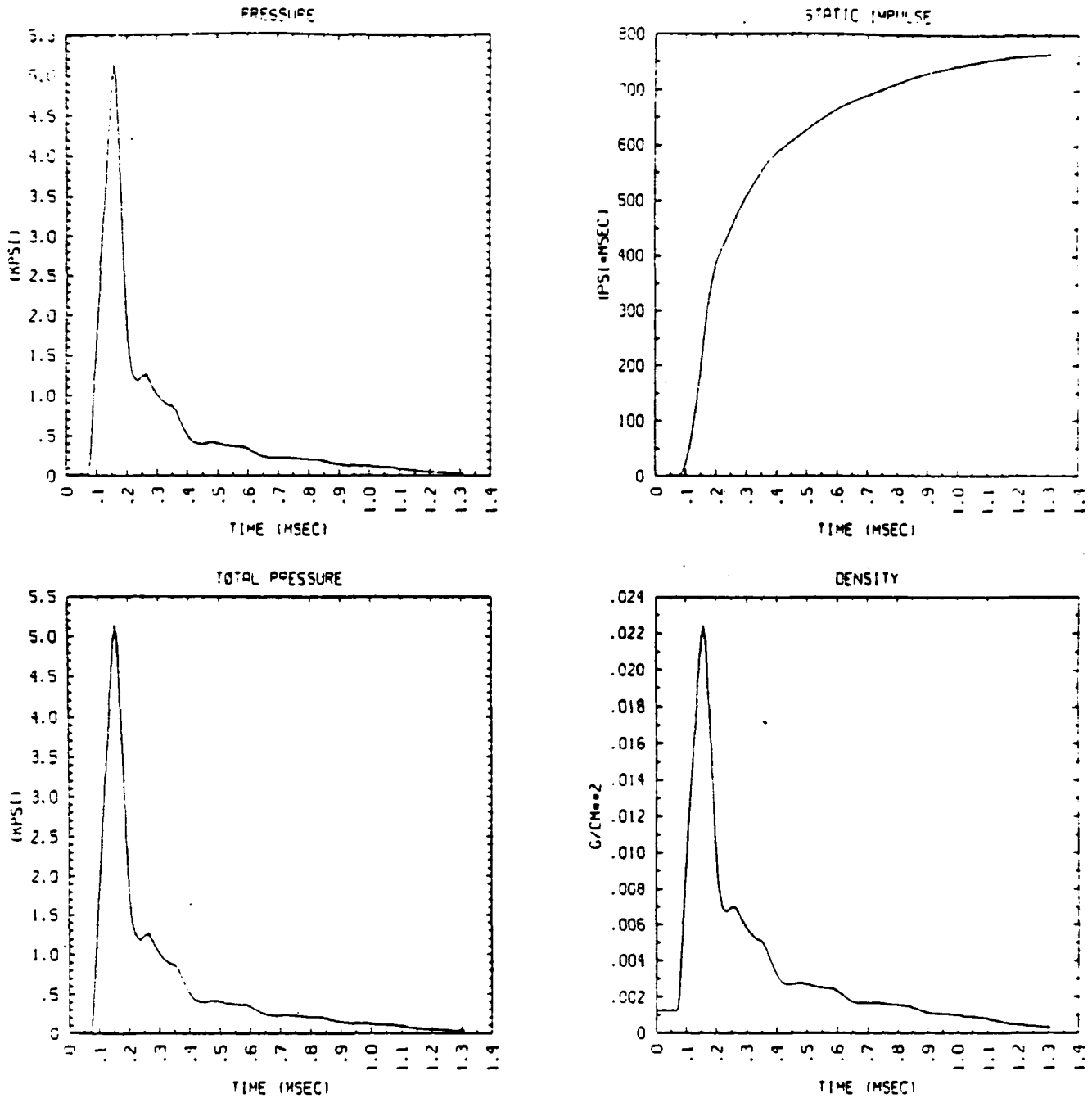


Figure 1.13b Time history for pressure, impulse, total pressure and density on the ground. X=5, Y=0, Z=0

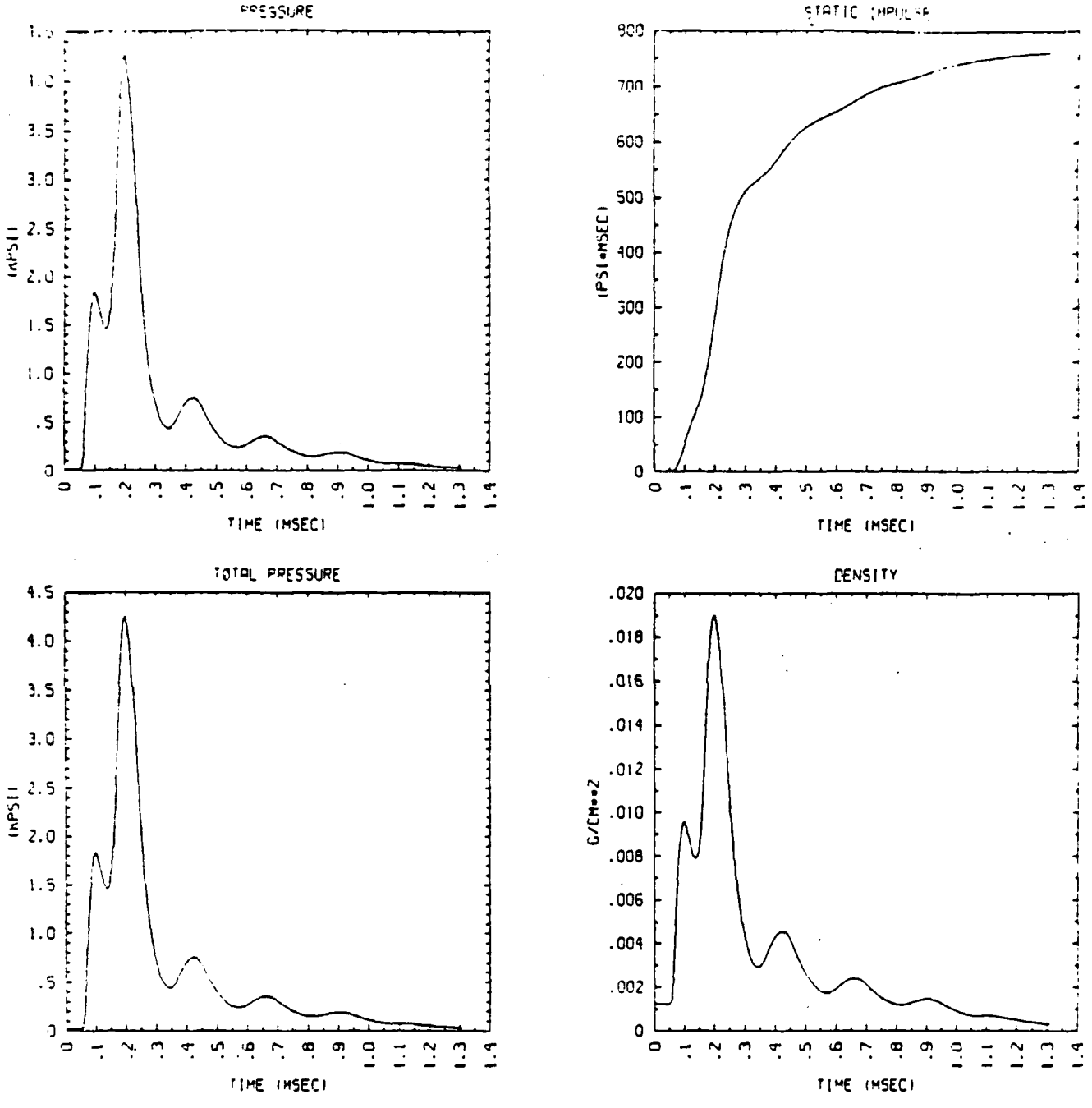


Figure 1.13c Time history for pressure, impulse, total pressure and density on the ground. X=10, Y=0, Z=0

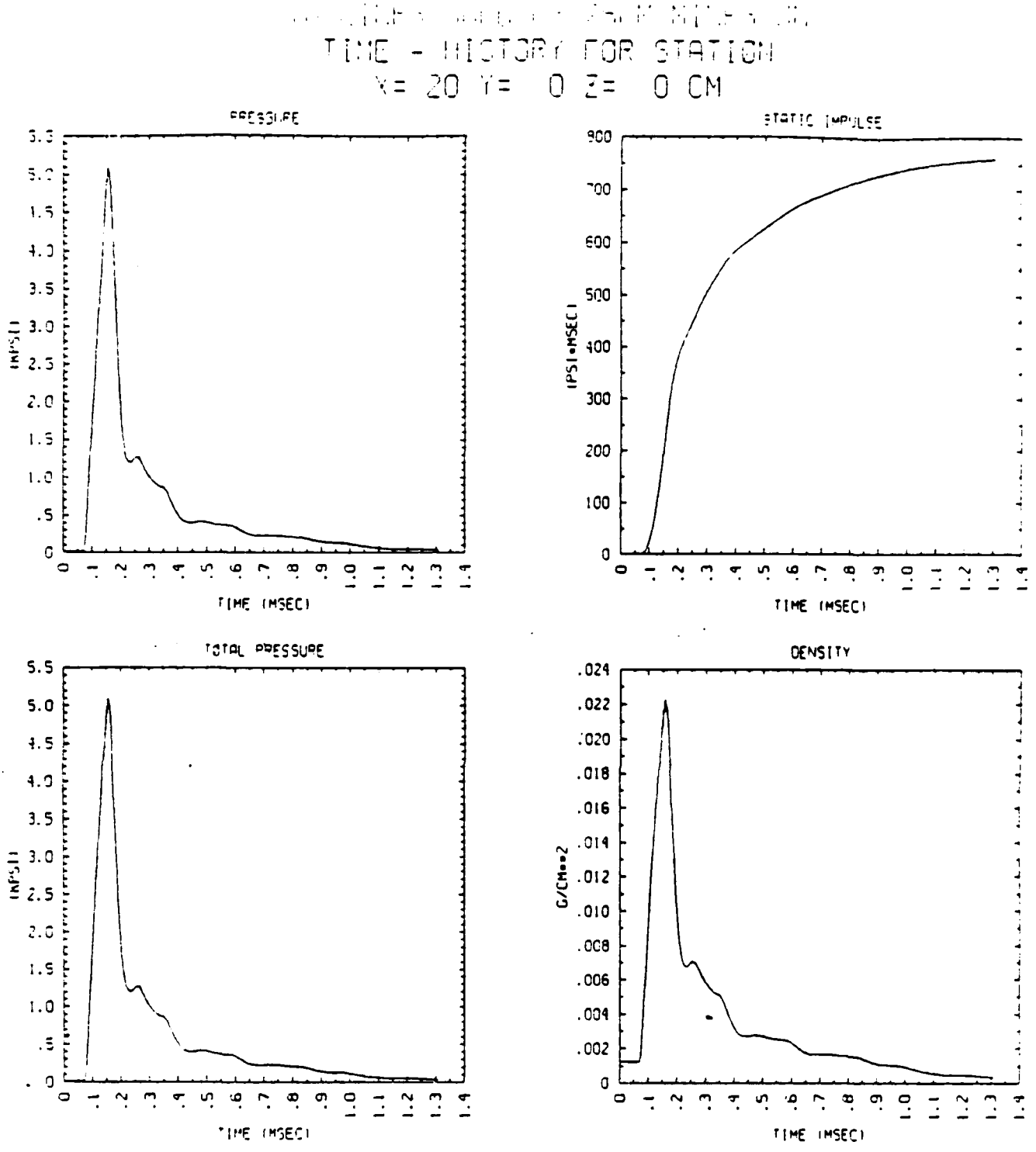


Figure 1.13d Time history for pressure, impulse, total pressure and density on the ground. X=20, Y=0, Z=0

COMPARISON OF M58, FAE AND DECA

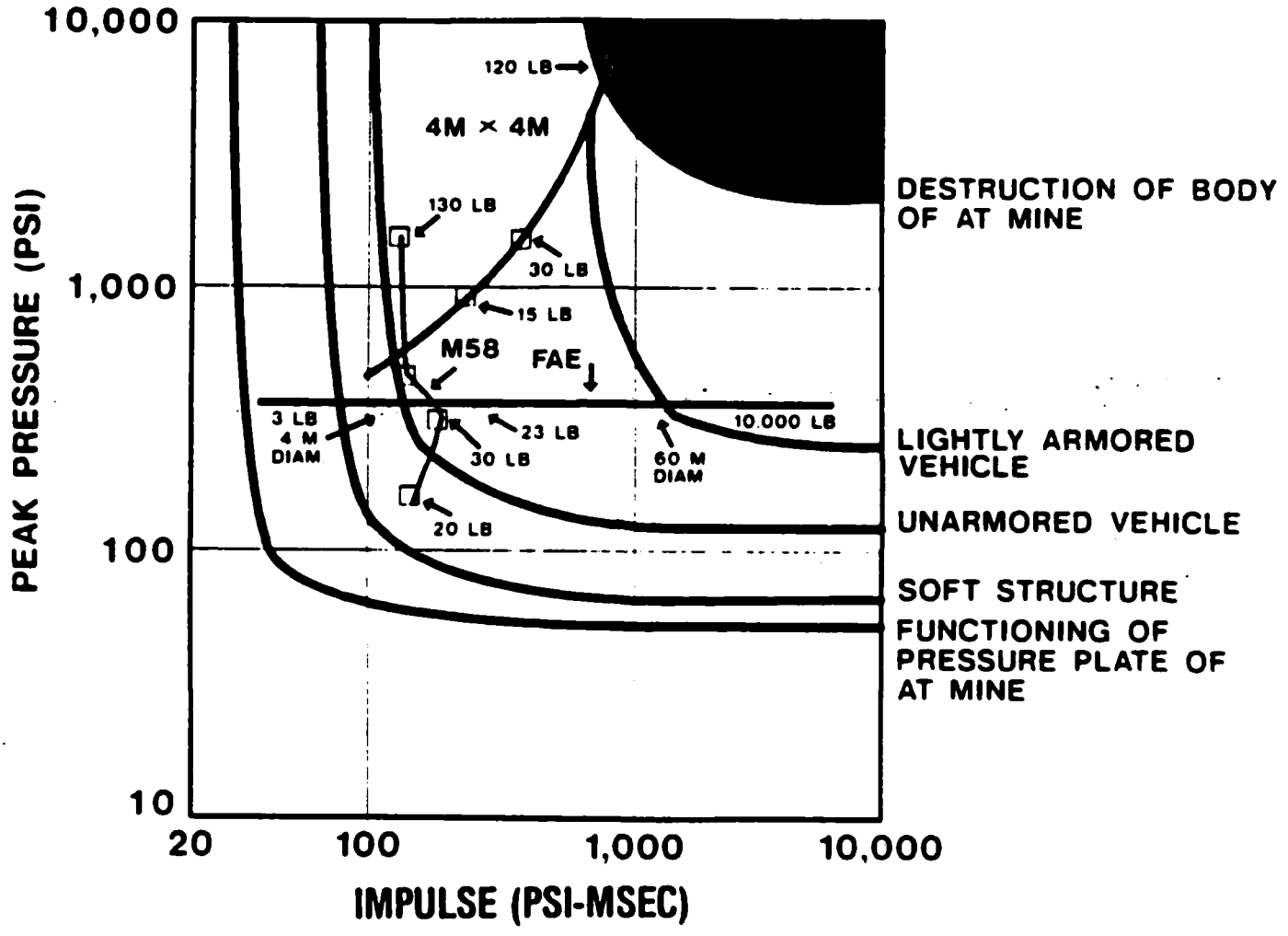


Figure 1.14 Maximum pressure and impulse as function of surface (volume for FAE) density of explosive.

Table 1. SUMMARY OF TEST RESULTS

Test No.	Explosive	No. of Lines	Spacing (m)	Height Above Ground (m)	Area: Width Length (m x m)	Areal Density (kg/m^2)	Peak Pressure (kPa) 1 m Above Ground	Peak Pressure (kPa) 2 m Above Ground	Condition of Inert Mine
1	Iremite	6	0.5	0	2.5 x 4.5	2.6	-[a]	-[a]	
2	Iremite	10	0.5	0	4.5 x 4.5	2.6	-[a]	-[a]	
3	Iremite	10	0.5	0	4.5 x 4.5	2.6	-[a]	-[a]	
4	Iremite	10	0.5	0	4.5 x 4.5	2.6	445	2517	
5	Iremite	10	0.5	0	4.5 x 4.5	2.6	425	2172	
6	Iremite	10	0.5	0	4.5 x 4.5	3.9	3964	2241	
7	Iremite	10	0.5	0.25	4.5 x 4.5	2.6	2861	1517	
8	Primacord	10	0.25	0.25	2.25 x 2.5	2.4	-[b]	-[b]	Lightly damaged
9	Primacord	10	0.25	0.25	2.25 x 2.5	2.4	4826	2054	Pressure plate destroyed, mine case crushed
10	Primacord	12	0.25	0.50	2.75 x 2.5	2.4	5722	2606	Pressure plate severely damaged, mine slightly crushed

[a]No useful data were obtained because of severe vibrations in instrumentation poles.

[b]Explosive did not fully detonate (about 50% detonation).

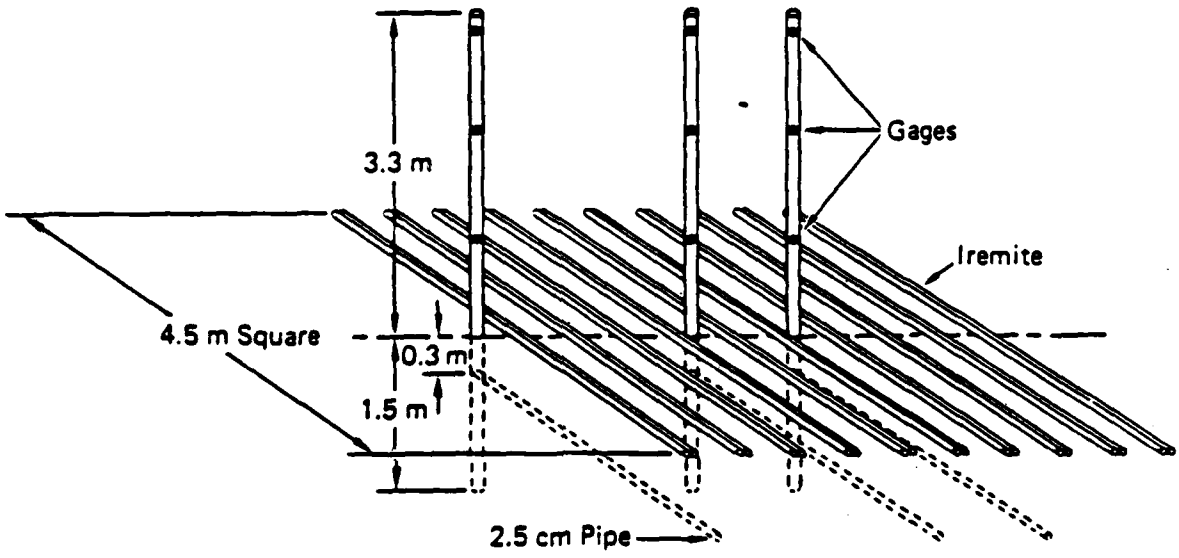


Figure 2.1 Prespective of experimental configuration.

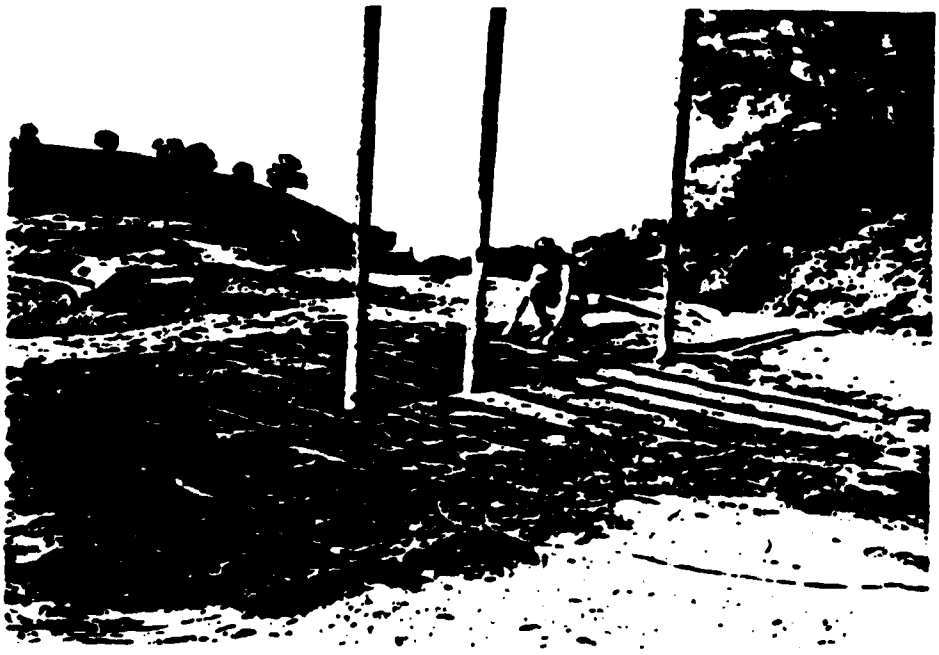


Figure 2.2 Explosive array and insrumentation poles.

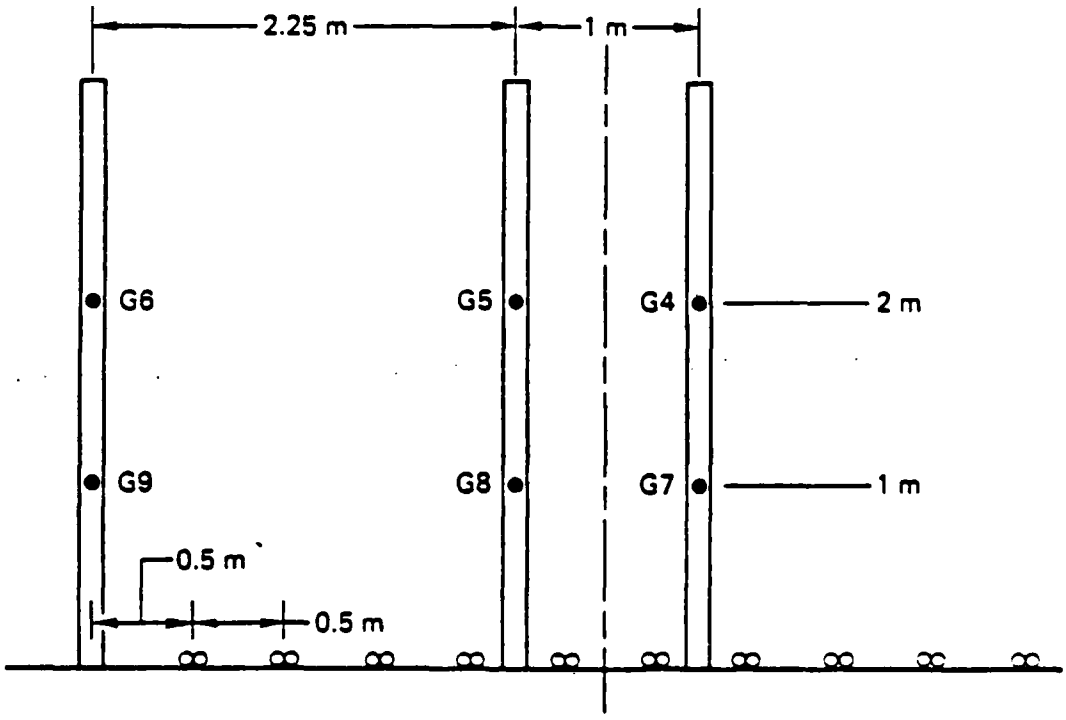
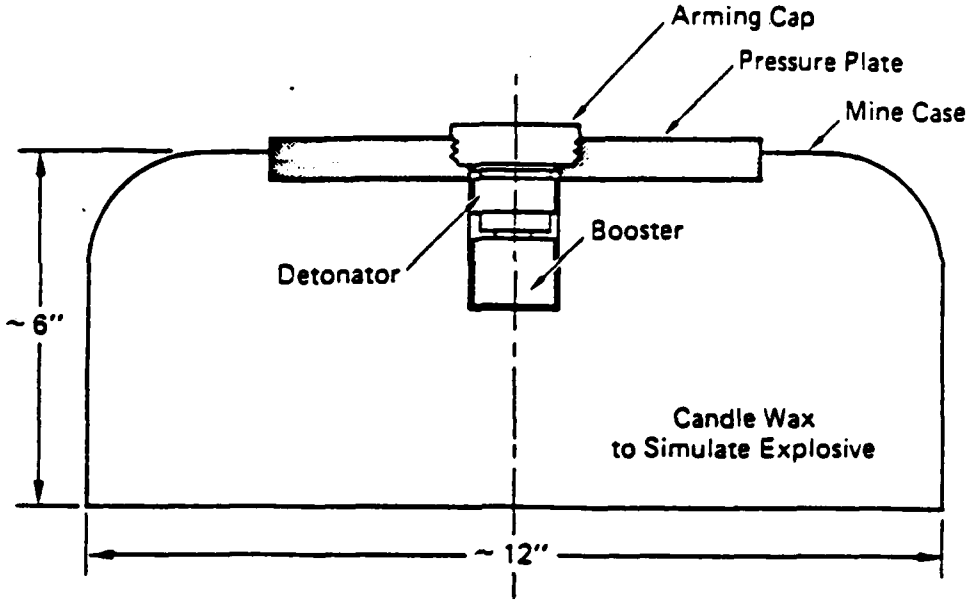
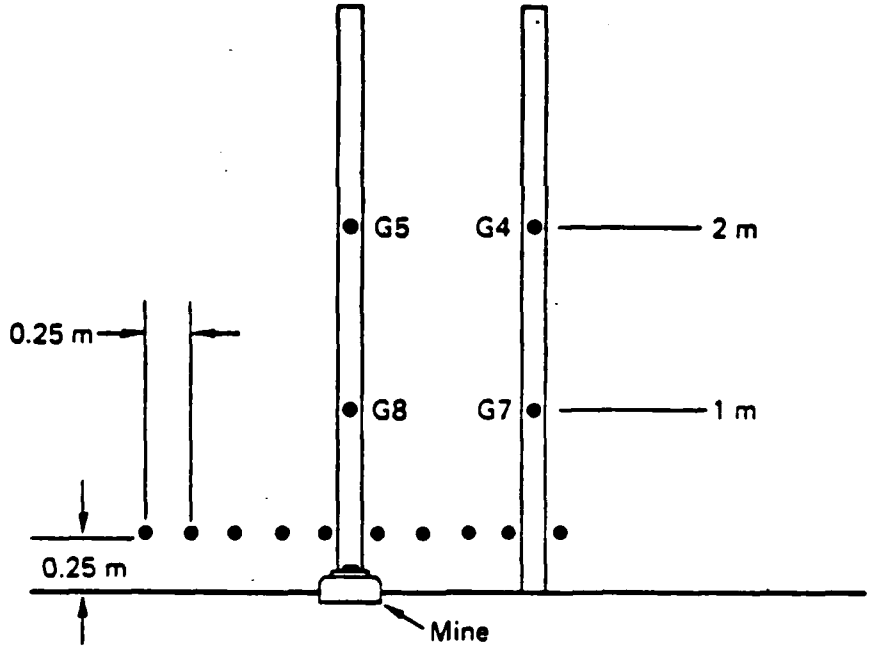


Figure 2.3 Experimental setup showing instrumentation poles.

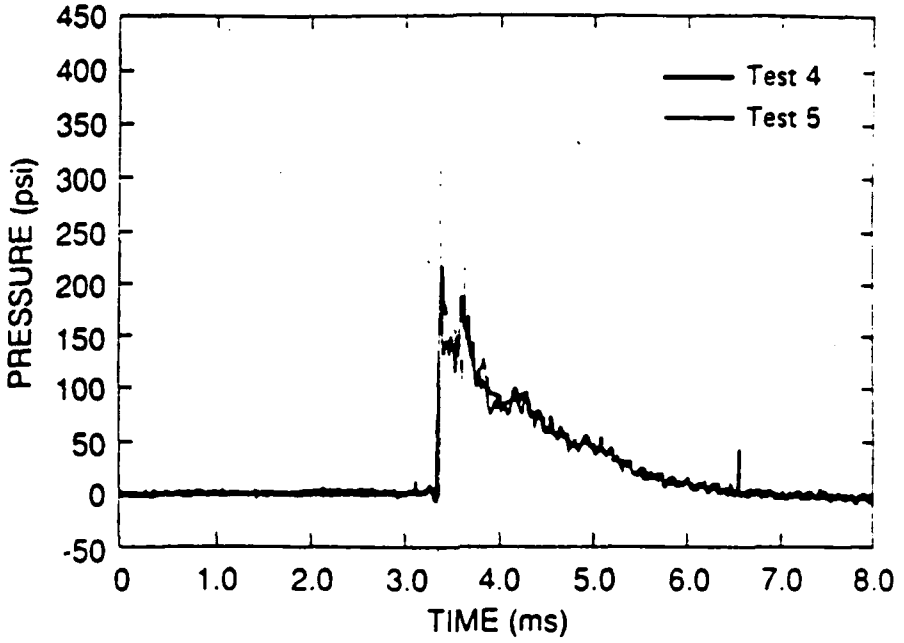


(a) Cross Section of Antitank Mine

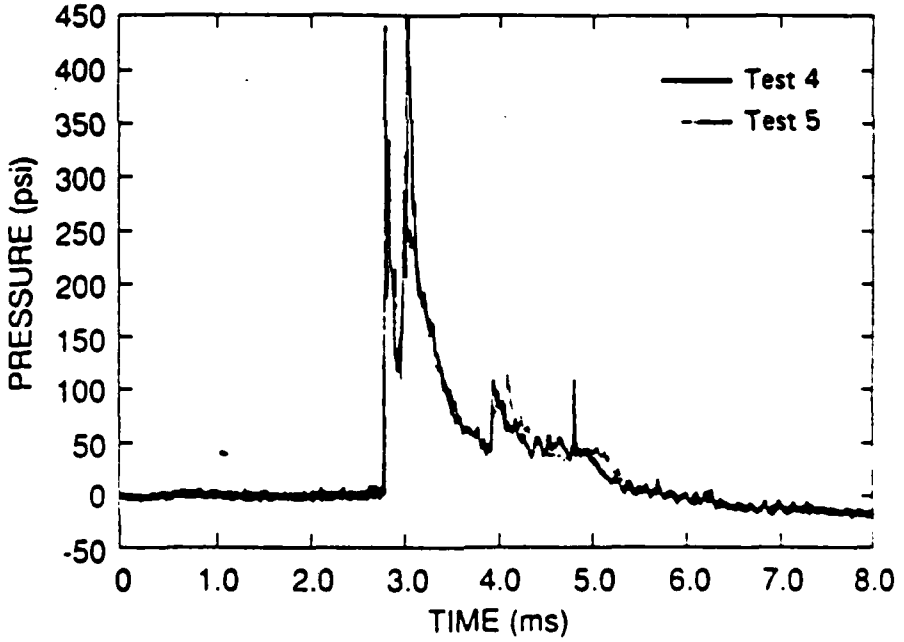


(b) Explosive Array and Instrumentation

Figure 2.4 Experimental configuration for antitank mine tests.



(a) Gage G4 (2 m above soil)



(b) Gage G7 (1 m above soil)

Figure 2.5 Comparison of tests 4 and 5.

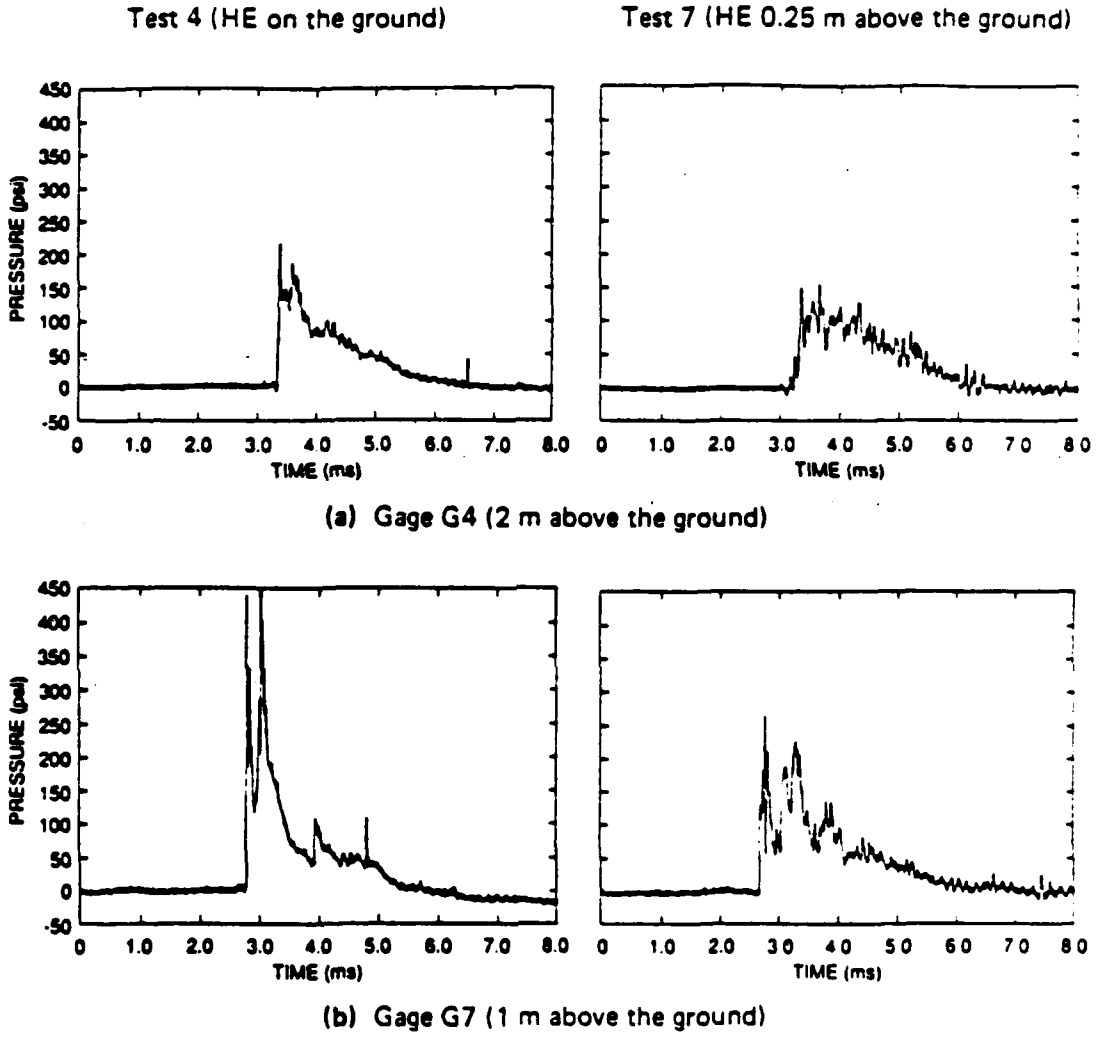
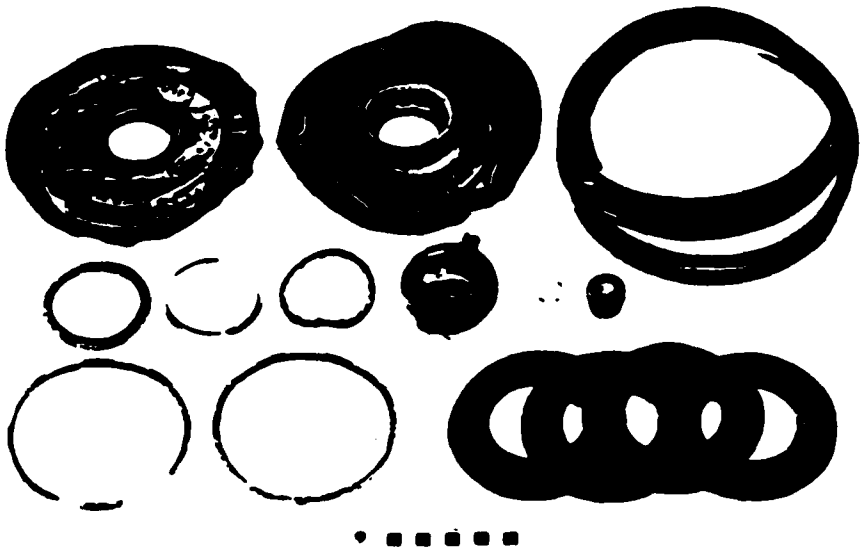


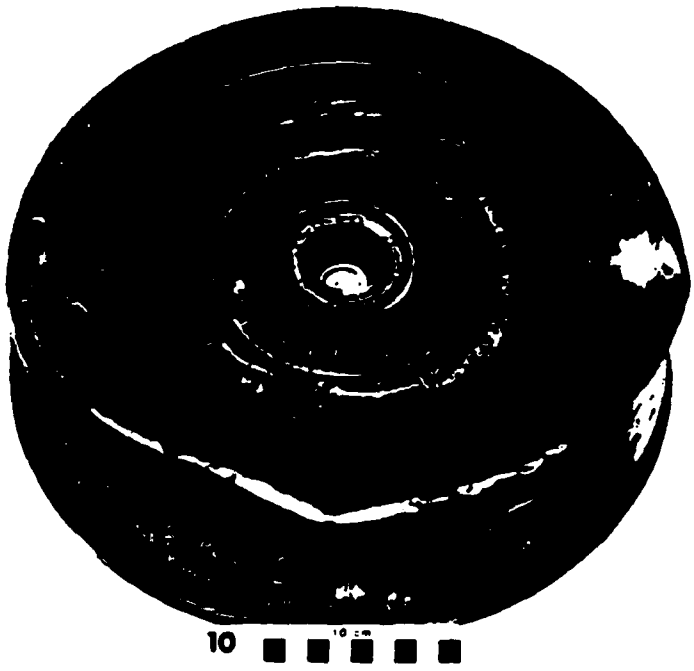
Figure 2.6 Comparison of data to show effects of suspending explosive above the ground.

(a) Mine Casing

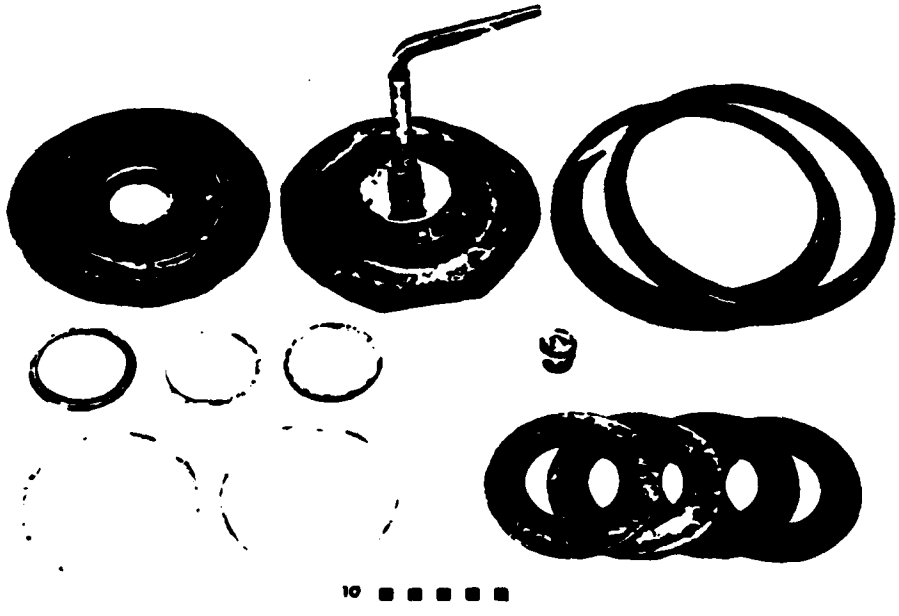


(b) Pressure Plate and Detonator Components

Figure 2.7 Damage to antitank mine in test 9.

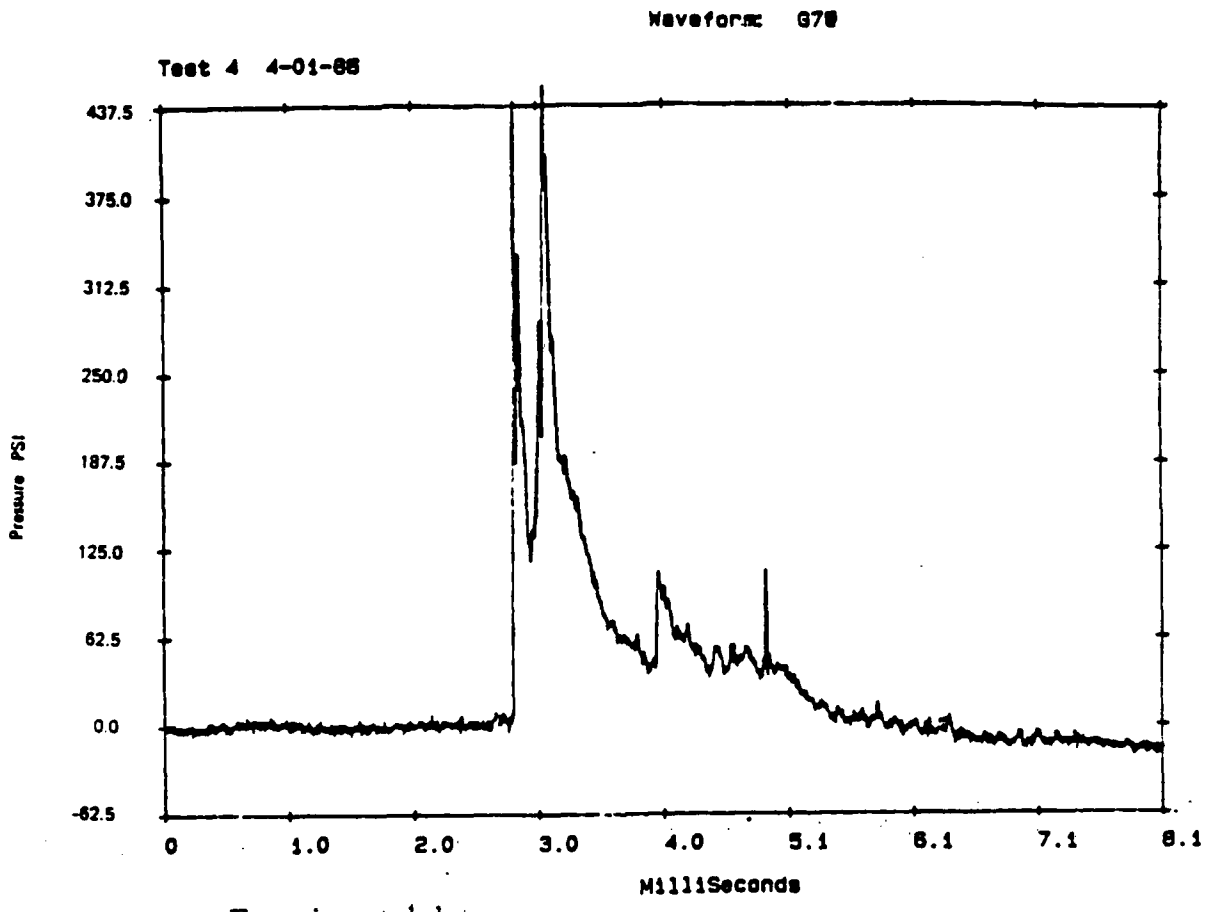


(a) Mine Casing

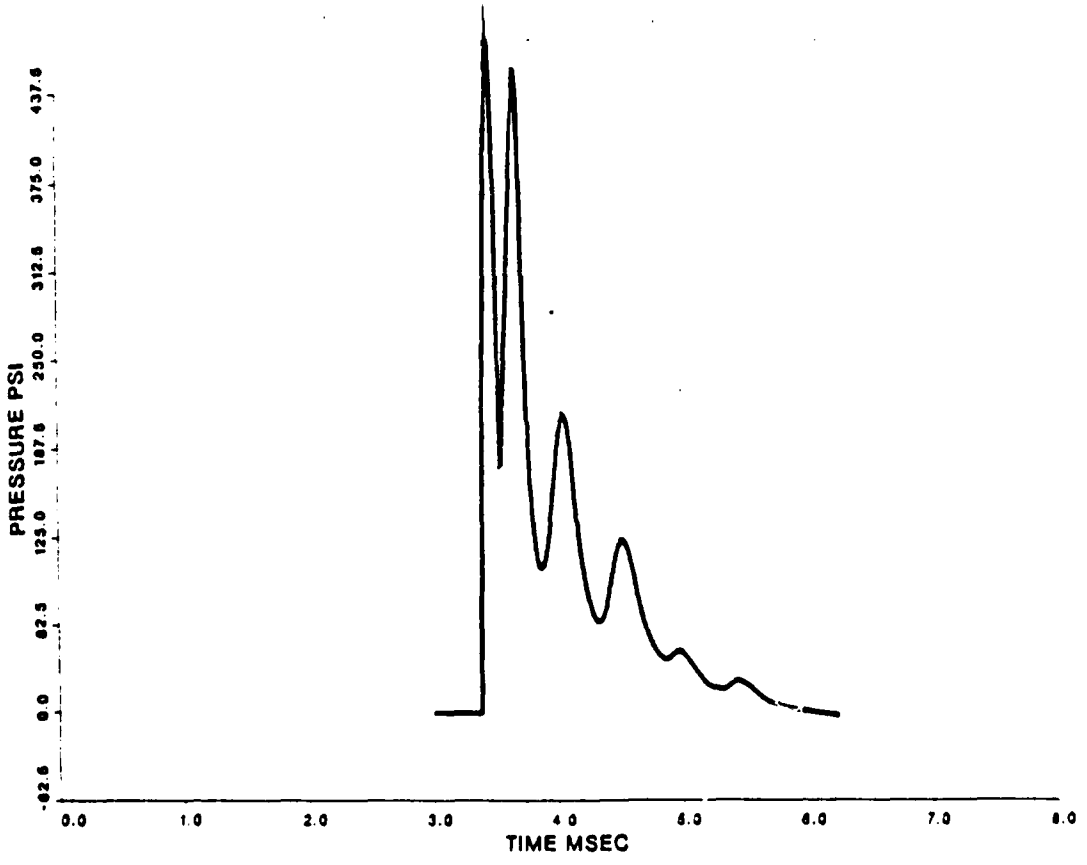


(b) Pressure Plate and Detonator Components

Figure 2.8 Damage to antitank mine in test 10.

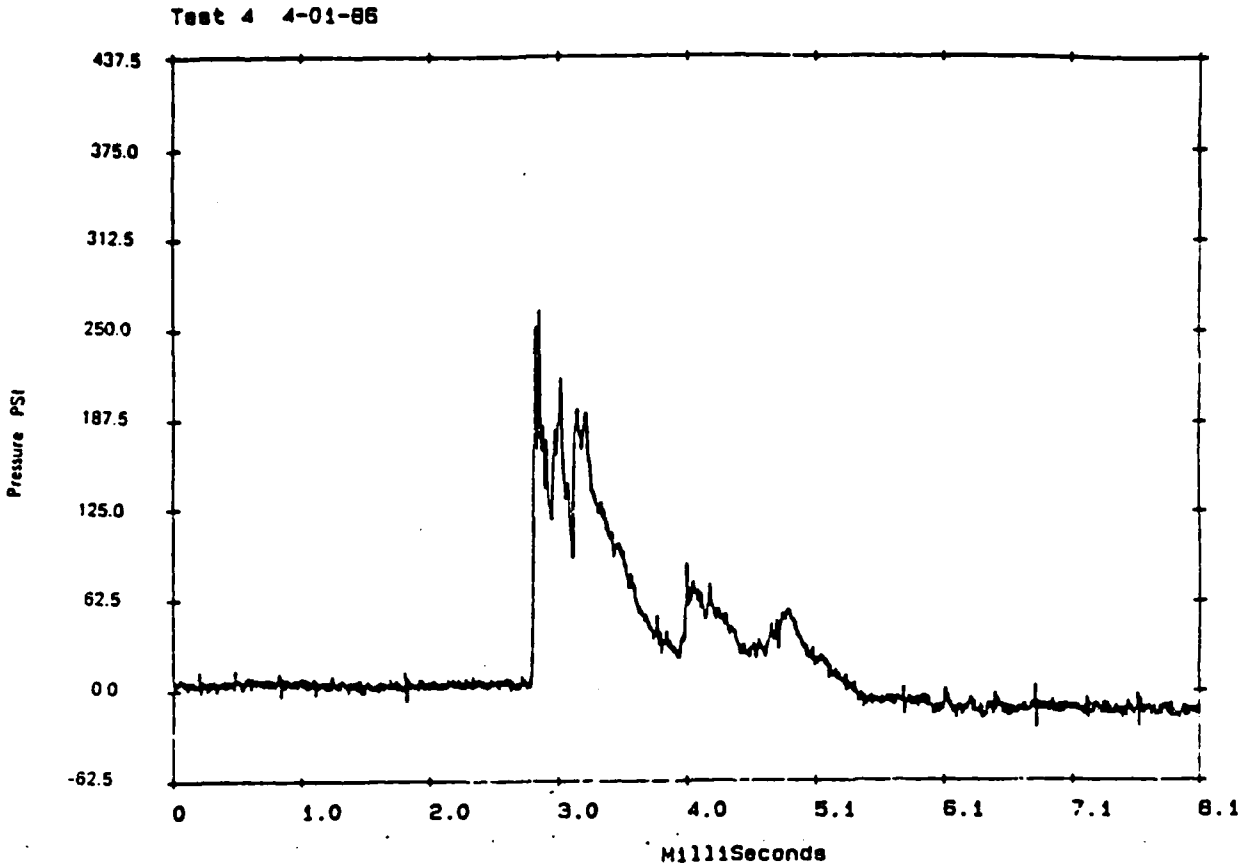


a . Experimental data

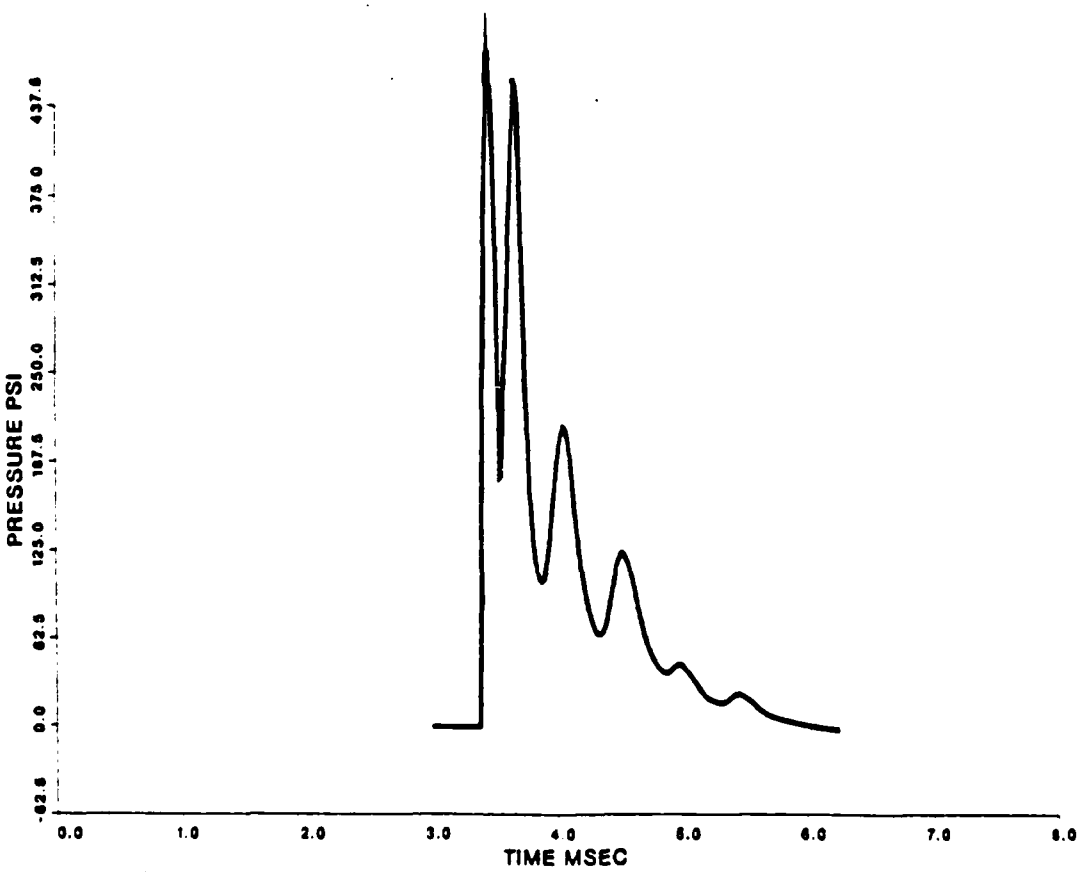


b . Computer simulation.

Figure 3.1 Comparison of data from gauge G7 (test 4) and simulation.

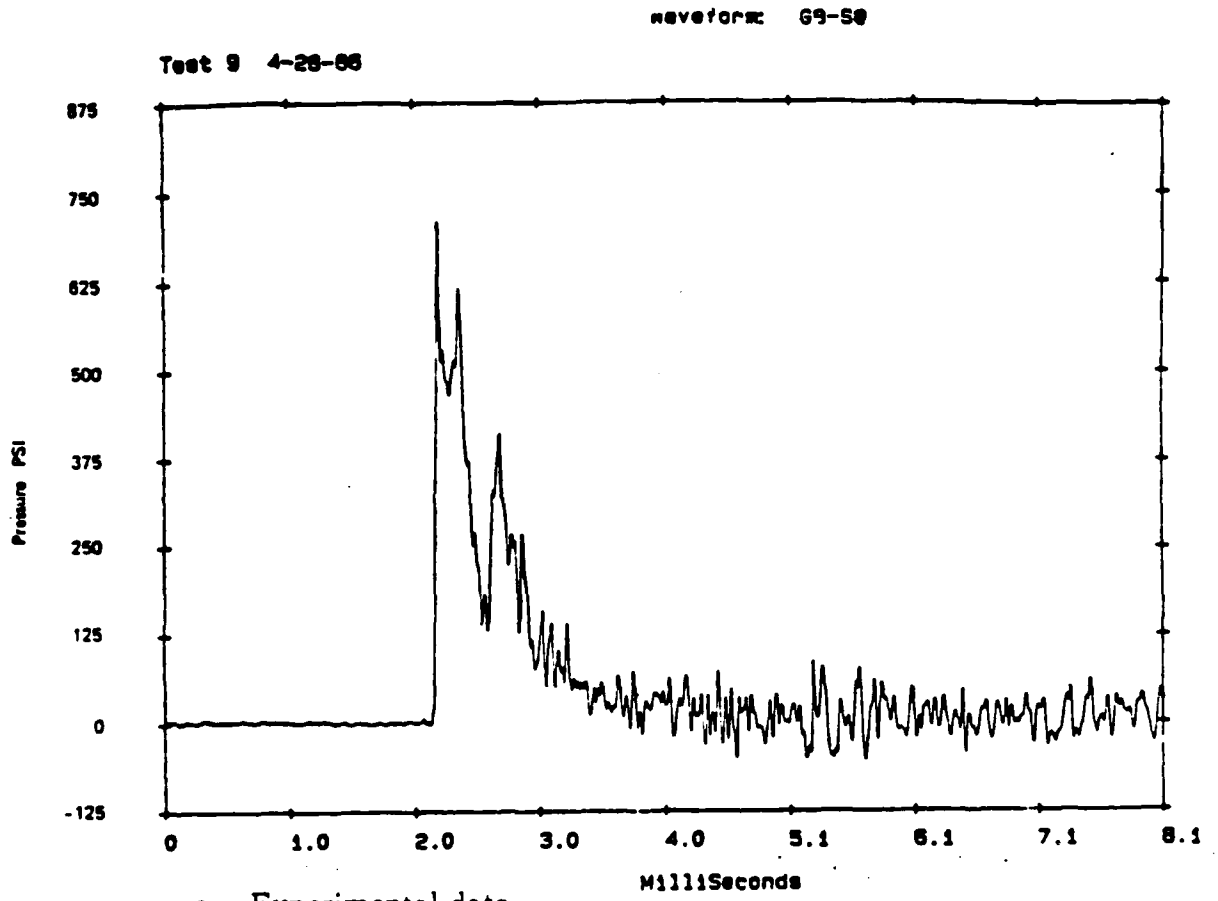


a . Experimental data

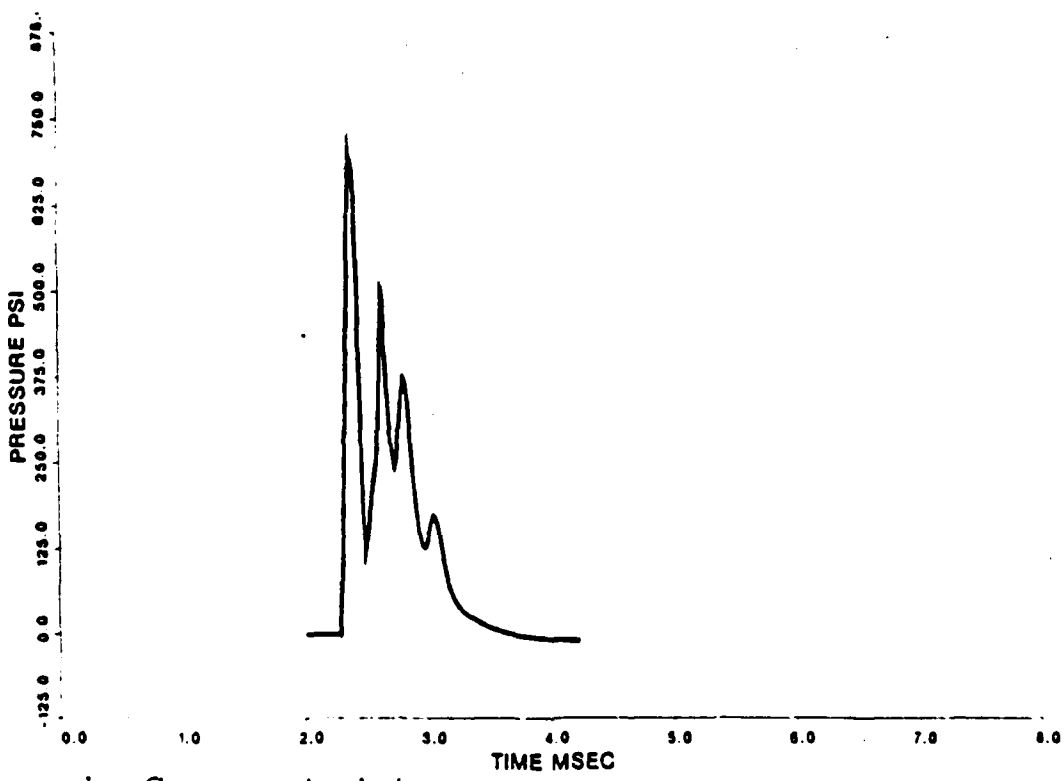


b . Computer simulation.

Figure 3.2 Comparison of data from gauge G8 (test 4) and simulation.

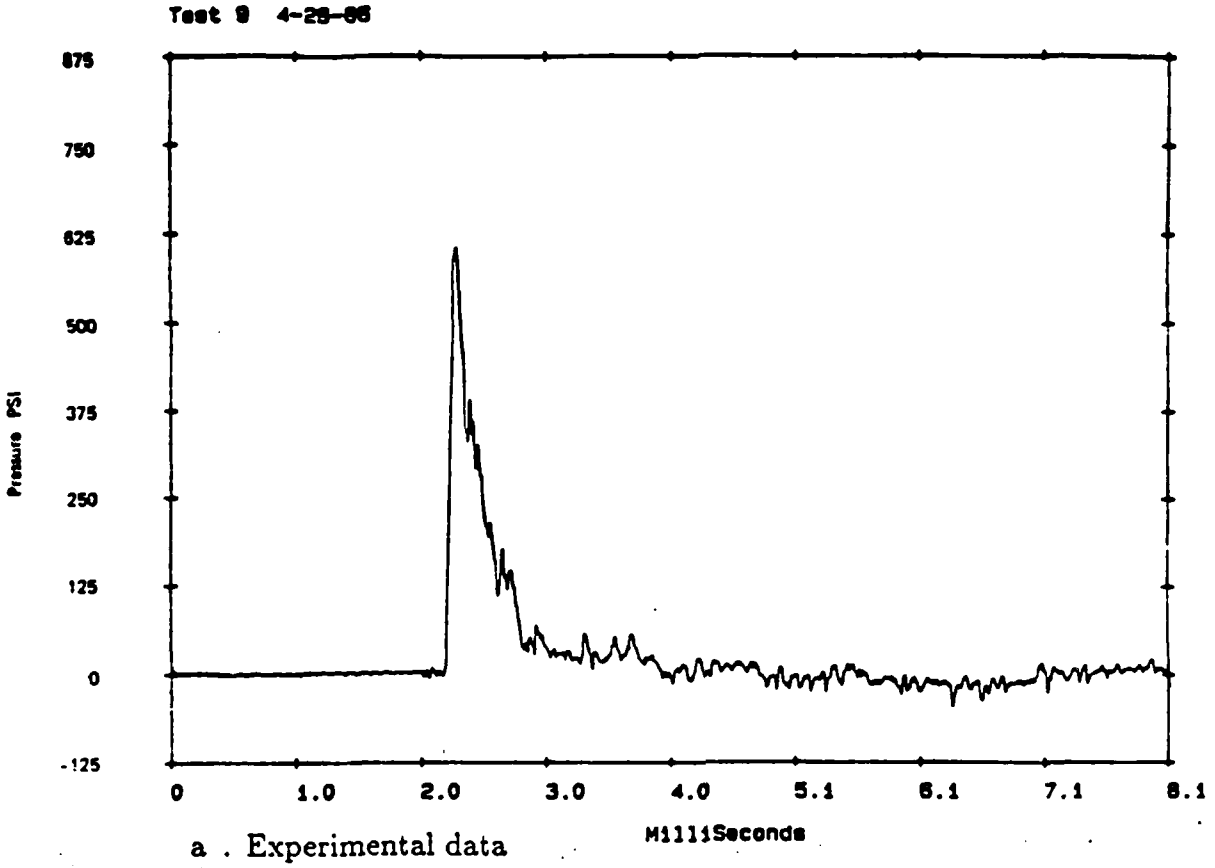


a . Experimental data

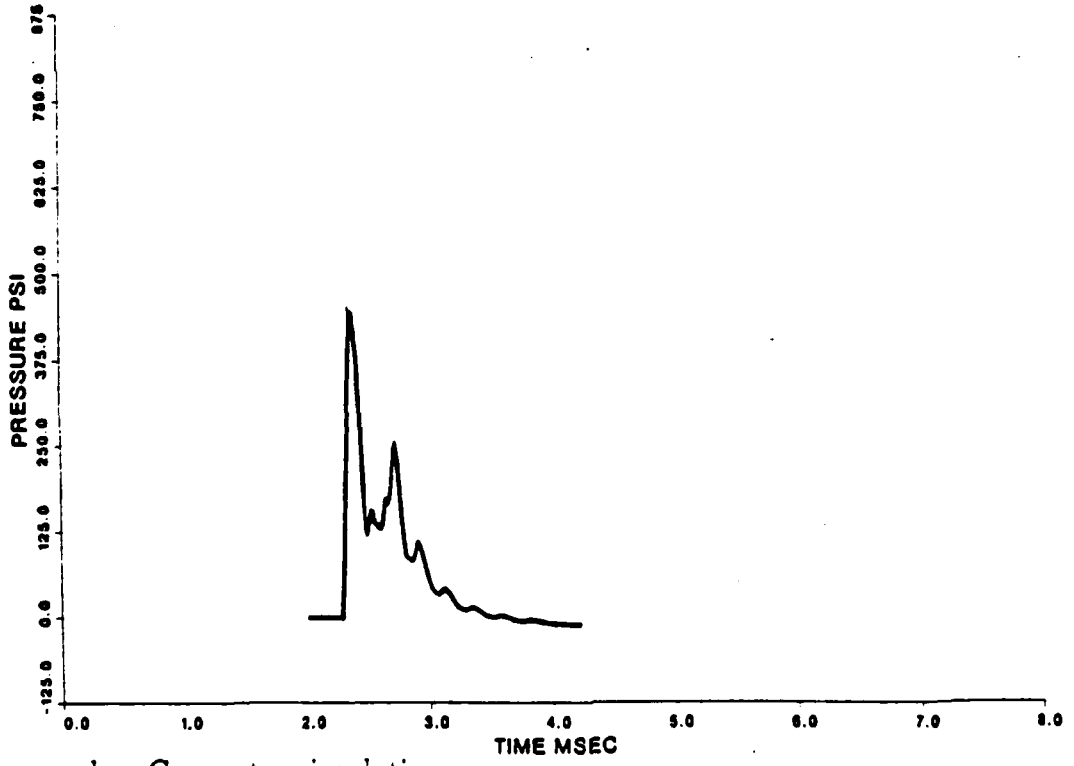


b . Computer simulation.

Figure 3.3 Comparison of data from gauge G8 (test 9) and simulation.

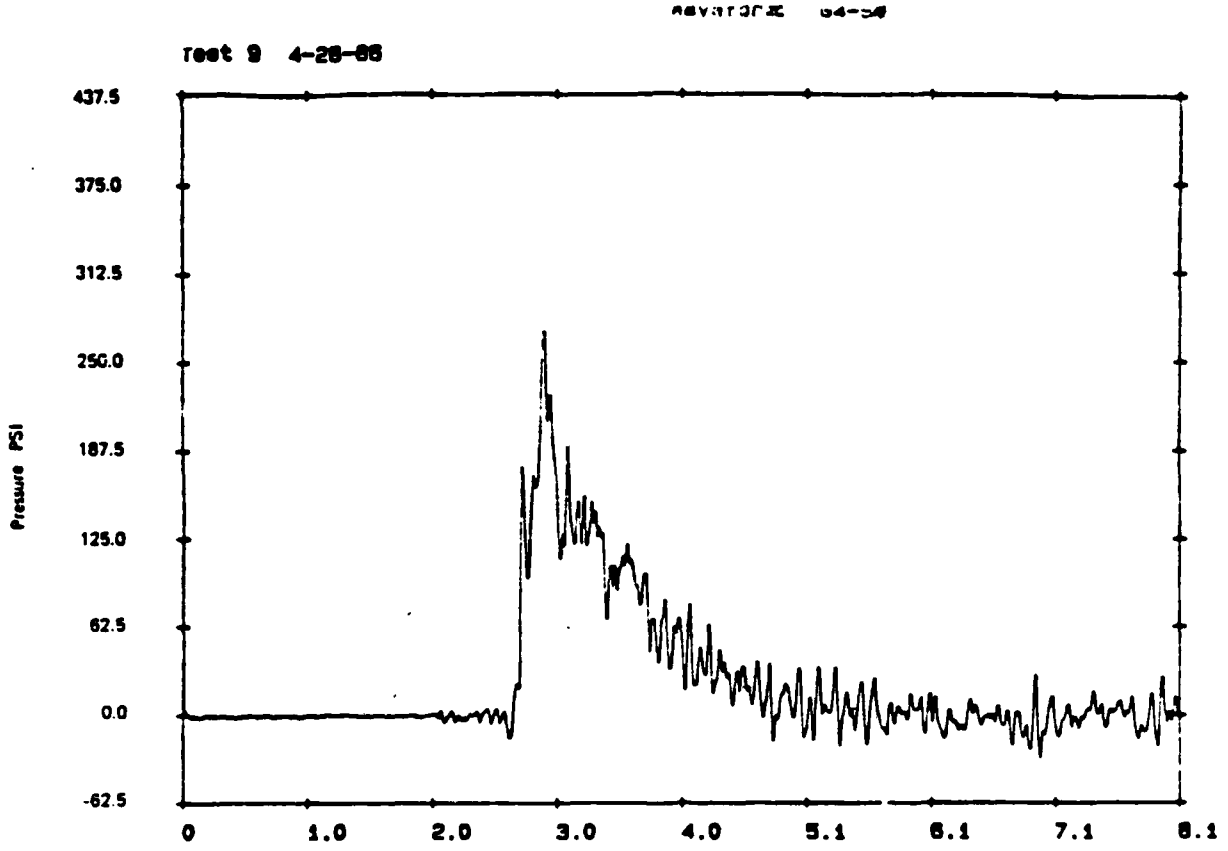


a . Experimental data

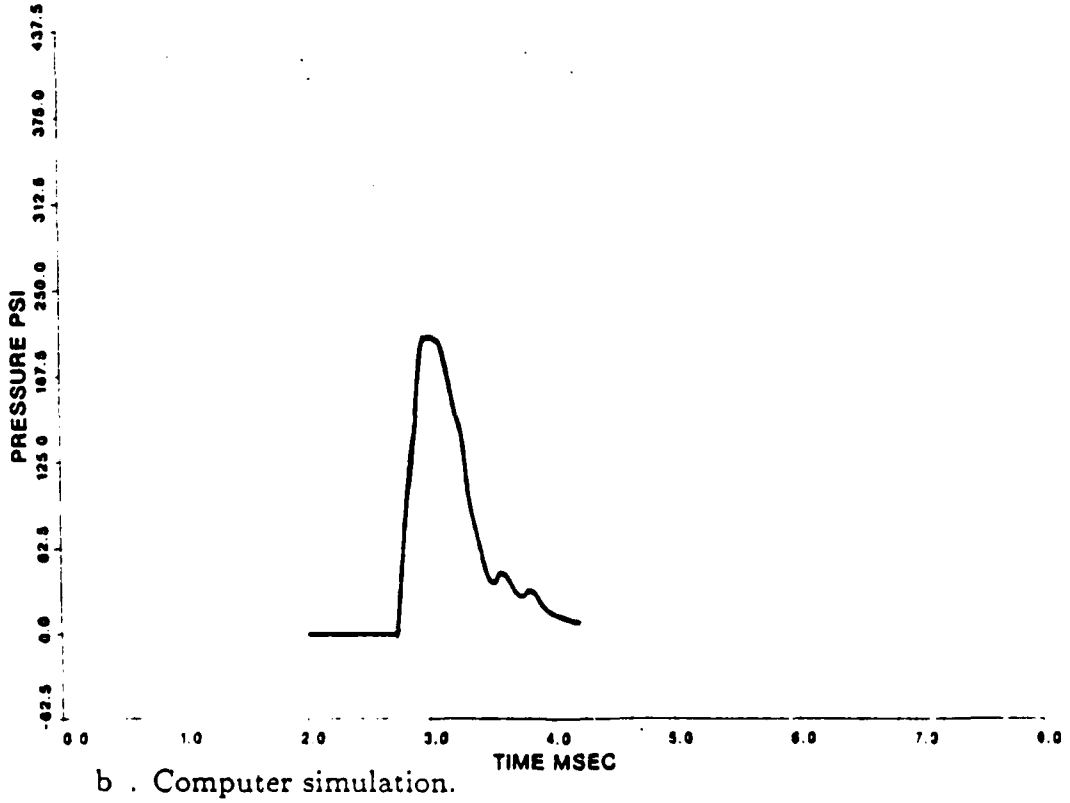


b . Computer simulation.

Figure 3.4 Comparison of data from gauge G7 (test 9) and simulation.

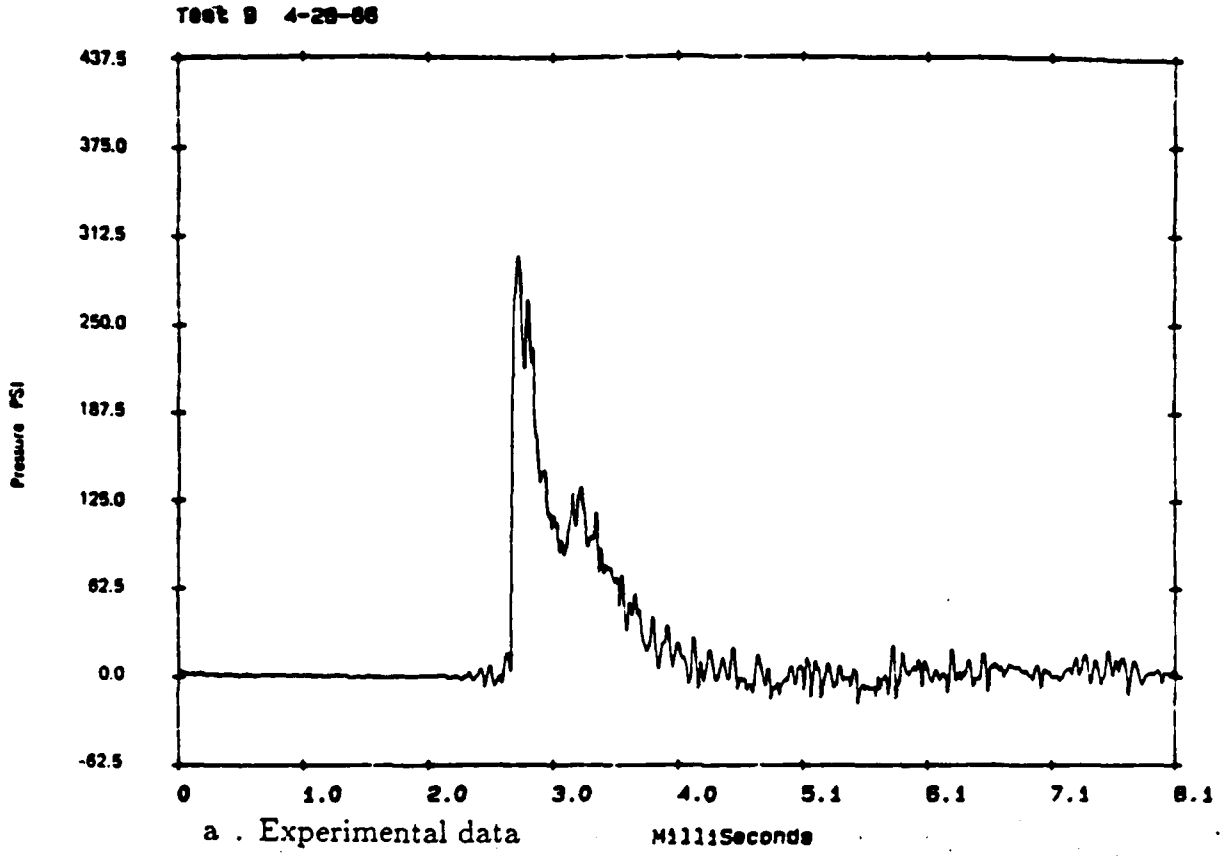


a . Experimental data

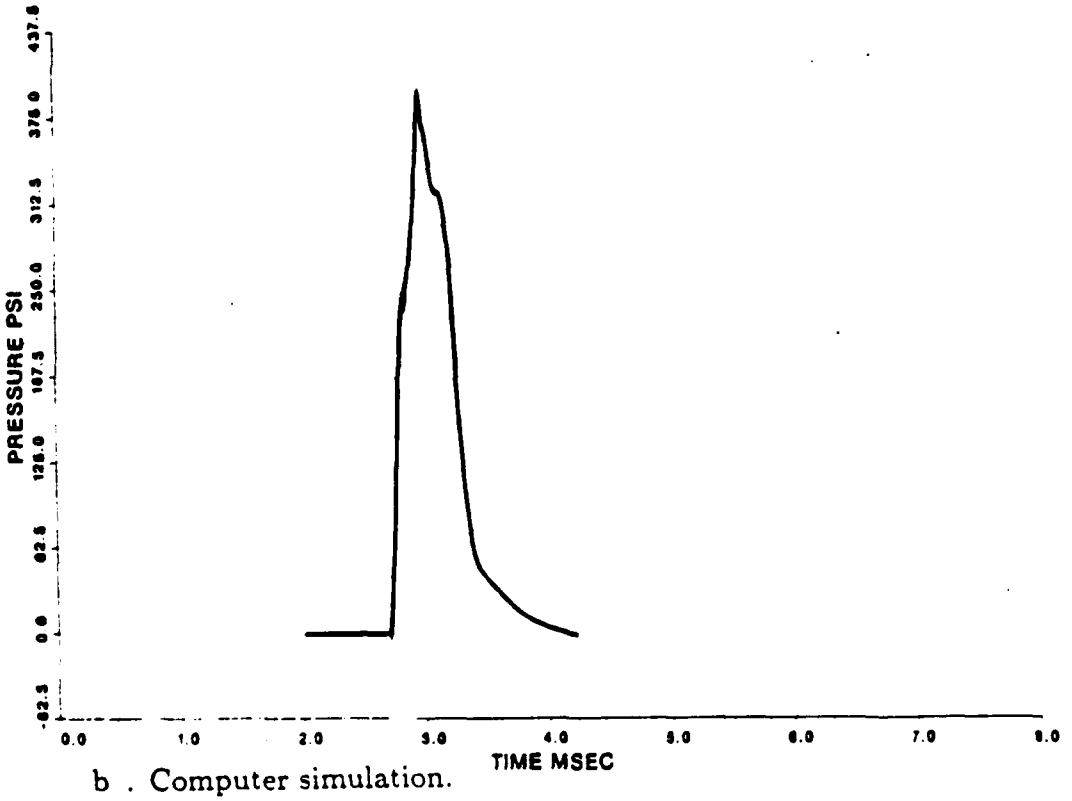


b . Computer simulation.

Figure 3.5 Comparison of data from gauge G4 (test 9) and simulation.



a . Experimental data



b . Computer simulation.

Figure 3.6 Comparison of data from gauge G5 (test 9) and simulation.

APPENDIX A

THREE-DIMENSIONAL ALGORITHM DESIGN FOR HYDRODYNAMIC CALCULATIONS

Mark A. Fry and Pradeep Kamath

Science Applications, Inc.

1710 Goodridge Drive

McLean, VA 22102

and

David L. Book

Laboratory for Computational Physics

Naval Research Laboratory

Washington, DC 20375

ABSTRACT

A new three-dimensional hydrodynamic algorithm has been deveoped utilizing a virtual system for data storage. The code called FAST3D solves partial differential equations aproximated by finite differences. Data are stored on y - z planes, so that only three planes are in memory at any given time. The choice of the algorithm together with the data handling system provide a highly accurate and efficient simulation tool for use on vector computers. Sample calculations are provided to illustrate the versatility of the computer code.

1. Introduction

Three-dimensional hydrodynamic algorithms must be accurate and highly efficient. Large amounts of memory coupled with large amounts of central processer time (CPU) are required for even moderately well resolved problems. FAST3D addresses these points as well as obtaining the required storage by implementing a virtual system. Details of the algorithm and sample calculations are presented.

2. Design And Implementation of FAST3D

We describe FAST3D, a code which solves the three-dimensional hydrodynamic equations for the conservation of mass, momentum and energy of an ideal fluid using a new leapfrog FCT algorithm. The equations advancing the dependent variables are

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho v) = 0; \tag{1}$$

$$\frac{\partial(\rho v)}{\partial t} + \nabla\cdot(\rho vv) = -\nabla(P - P_a) - (\rho - \rho_a)\, G(z); \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla\cdot(Ev) = \nabla\cdot(Pv) + S - L. \tag{3}$$

To calculate the pressure we employ an equation of state in the form

$$P = P(e,\rho), \tag{4}$$

where $e = E - \frac{1}{2}\rho v^2$. In Eq. (2), the ambient pressure P_a and density ρ_a are subtracted from corresponding time-dependent quantities in the source terms to avoid the "falling sky" problem; G(z), the acceleration due to gravity, is generally taken to be constant at its nominal sea-level value. S and L denote energy sources and losses.

The solution is found by numerically advancing the equations in Cartesian geometry on a rectangular variably-spaced mesh. The basic finite-difference technique is the fully three-dimensional leapfrog scheme with no time splitting. This scheme has very low dissipation, with a linear amplification matrix of unity on a uniform grid, but is subject to a weak (grid separation) instability and large dispersive errors at short wavelengths. These difficulties are overcome by using the Flux-Corrected-Transport (FCT) technique, in which the final solution is made up of the weighted average of a high-order scheme (leapfrog) and a low-order scheme, in this case upwind differencing, designed to maintain positivity (monotocity) where it is required physically. The FCT algorithm is implemented in the code by adding to the leapfrog convective terms a diffusive flux pro-

portional to the absolute velocity, plus a small velocity-independent diffusive term. If F represents any of the dependent fluid variables, the finite-difference scheme used to approximate the left-hand side of Eqs. (1)-(3) is given by

$$F^{td}_{ijk} = F_{ijk} + \frac{\Delta t}{\Delta x}\left[(Fu)_{i-1} - (Fu)_{i+1}\right] + \frac{\Delta t}{\Delta X}\, u_{i+1}\,(F_{i+1} - F_i)$$

$$- \frac{\Delta t}{\Delta x}\, u_{i-1}\,(F_i - F_{i-1}) + C(F_{i+1} - 2F_i + F_{i-1}) + \text{other two directions} \quad (5)$$

for the half of the zones which are active at any given time step, and

$$F^{td}_{ijk} = F_{ijk} + \left(\frac{\Delta t}{\Delta s}\, u_i + C\right)(F_{i+1} - 2F_i + F_{i-1}) + \text{other two directions} \quad (6)$$

for the half which are inactive. Here C is a constant used to apply extra diffusion (beyond that needed to ensure positivity) in strong-shock problems.

An antidiffusive step is added to the algorithm in timesplit fashion, removing most of the added diffusion. To do this we begin by using the transported diffused momenta and density to calculate an interim velocity in each direction:

$$\bar{u}_i = (\rho u)^{td}_i / \rho^{td}_i. \quad (7)$$

Then we calculate "raw" antidiffusive fluxes for the active zones,

$$\phi_{i-1/2} = \left(\frac{\Delta t}{\Delta x}\,\bar{u}_{i-1} + C\right)(F^{td}_{i+1} - F^{td}_i), \quad (8a)$$

$$\phi_{i-1/2} = \left[\frac{\Delta t}{\Delta x}\,\bar{u}_{i-1} + C\right](F^{td}_i - F^{td}_{i-1}), \quad (8b)$$

and the inactive zones,

$$\phi_{i+1/2} = \left(\frac{\Delta t}{\Delta x}\, u_i + C\right)(F^{td}_{i+1} - F^{td}_i), \quad (9a)$$

$$\phi_{i-1/2} = \left(\frac{\Delta t}{\Delta x}\,\bar{u}_i + C\right)(F^{td}_i - F^{td}_{i-1}). \quad (9b)$$

Using the difference between neighboring values of the transported diffused

quanitities,

$$\Delta_{i+1/2} = F^{td}_{i+1} - F^{td}_{i}, \tag{10}$$

and the signs of the raw fluxes,

$$S_{i+1/2} = \text{sign}(\phi_{i+1/2}), \tag{11}$$

we calculate the "corrected" fluxes according to the Boris-Book strong flux-limiting formula:

$$\phi^{c}_{i+1/2} = S_{i+1/2} \max[0, \min(\phi_{i+1/2}, \sigma_{i+1/2}\Delta_{i+3/2}, \sigma_{i+1/2}\Delta_{i-1/2}]. \tag{12}$$

These fluxes are then used in the antiduffusion stage to get the new values of the dependent variable:

$$F^{new}_{ijk} = F^{td}_{ijk} + \phi^{c}_{i-1/2} - \phi^{c}_{i+1/2} + \text{other two directions.} \tag{13}$$

The flux-correction part of the solution leaves just enough of the low-order scheme to guarantee monotonicity of the solutions. The driving terms [right-hand sides of Eqs. (1)-(3)] can be added in at almost any point in the calculation. Currently this is done following the transport and diffusion in Eq. (5).

The main region of the calculation employs equally spaced zones to maintain high acuracy. The grid, which does not vary in time, may be stretched in regions of little activity or near the edges to reduce the influence of boundary approximations. Either reflecting or outflow boundaries may be applied in the transverse (y and z) directions and a choice of reflecting, outflow, or periodic boundary conditions is availabe in the longitudinal direction. A variable time step is used based on the minimum CFL limit in each direction over the whole mesh, $\Delta t < \min[\Delta x/(v_x + c), \Delta y/(v_y - c), \Delta z/(v_z + c)]$. The momentum equation contains a gravity term. A real-gas equation of state is also available for strong shock calculations in air.

For our applications, the shocks and other unsteady phenomena being simulated are initially localized near one end of the system, conventionaly taken as the origin of the x axis. At early times it is unnecessary to update variables far from this region, so the number of active zones in this direction is $(NX)' < (NX)'_{max}$. At late times (NX)' increases until the entire mesh is active.

Three-dimensional hydrodynamic calculations require large amounts of memory, typically more than is physically available on a given computer. This necessitates the use of auxiliary storage such as disk. The FAST3D algorithm uses only the neighboring values of the fluid variables to time-advance a given value. The data are stored on y-z planes, so that only three planes need be in memory at any given time during the computation (Fig. 1). Furthermore, by allowing I/O buffers for transferring planes to and from the disk, computation can be overlapped with I/O to minimize overall computer time. This implementatin of the algorithm requires two passes through the auxiliary disk file per time step: one for the leapfrog time advancement and one for the FCT correction. The parallel architecture of the Cray XMP series can be utilized here. Other passes through the disk file which do not occur at every time step may be made as needed, e.g., to produce diagnostics or restart/dump files.

The leapfrog FCt algorithm involves enough arttithmetic to keep the transport part of the FAST3D code compute-bound. However, due to the explicit nature of the algorithm, the computations on the planes can be written to take advantage of vecotr hardware that exists on machines such as the Cray-1. When fully vectorized, FAST3D runs on the Cray-1 at about [illegible] (zone-step). The code has also been run an the Cray-XMP/12 at 15μs/zone step).

3. Problems and Results

The code was tested on a variety of simple idealized calculations. The Riemann (exploding diaphragm) problem was used since it embodies all the gas dynamic discontinuities one finds in shock physics. Figure 2 shows the density as a function of distance some time after diaphragm rupture. Initially the density and pressure increased across the diaphragm from left to right by factors of 10 and 100, respectively. An ideal gas equation with a gamma of 1.4 was used to relate pressure to internal energy. Examination of Fig. 2 reveals an accurate simulation of the shock front moving to the left, the contact discontinuity with only a small amount of smearing, and the smoothly varying rarefaction region propagating to the right.

A more difficult problem is presented to indicate the versatility of the FAST3D code. Shock tubes with moderate shock strengths corresponding to over pressures of approximately three atmospheres have been used to investigate the physics of shock diffraction in the pressence of heated layers and obstacles. Helium gas layers are generally used to simulated the higher temperature gas along the wall of the tube. Rigid obstacles are constructed and fixed downstream from the diaphragm. When the shock wave encounters the heated layer (helium) it begins to propagate faster and creates an additional contact surface (discontinuity in density but not pressure). Furthermore, the shock is inclined, moving upward, as one moves away from the shock tube wall. Figure 3 shows the density and pressure contours in the plane of symmetry (x - z axis). Note the "toeing out" of the shock front into the heated layer. When an additional discontinuity such as an abstacle is placed in this flow field, a more complex flow is produced. Three-dimensional graphics provide a unique way to view what is taking place. Graphics programs have been created with the same virtual data handling system as found in FAST3D. The results are show in Fig. 4. A series of four different density level contour plots, all at the same

time, provide a look into the features of interest in the calculation. Starting from left to right, one first sees a density level of $\rho = 5.0 \times 10^{-}$ which represents the helium layer. The intersection of the three axis is the origin for the cartesian frame of reference. The shock wave is moving from back to the front. The axis parallel to horizontal is the y axis and the vertical axis is the z axis. Second from left is the density level, $\rho = 2.0 \times 10^{-3}$, which shows the shock impinging on the rigid structure and the diffraction of the shock over the structure, respectively.

Current large-scale scientific computers provide insufficent central memory for even moderate simulations. The FAST3D code combines a state-of-the-art numerical algorithm with storage designed for vector computers. Additional speedups are possible by taking a account of the parallel features in the code. The result is the ability to model gas dynamic systmes which previously were too complex and to achieve accuracies not otherwise available.

Fig. 1. Representation of the data storage system used in FAST3D.

Fig. 2. Density and Pressure vs. distance for the Riemann problem. The vertical line in the middle indicates the position of the diaphragm.

Fig. 3. Density and Pressure contours in the x - z plane of symmetry. The shock is advancing from left to right. Note the thermal layer along the bottom in the density figure.

Fig. 4. Three-dimensinal representation of the shock on structure in a heated layer. Z axis is vertical to page, y axis is horizontal, and x axis moves out of page. The shock is moving from back to front. The figure at far left is contoured with density = 5×10^{4} g/cm^3. The middle is contoured with density = 2×10^{-3}. Lower right is density = 20 and upper right is density = 1.0×10^{-3}.

THREE--DIMENSIONAL OUT-OF-CORE CALCULATIONS

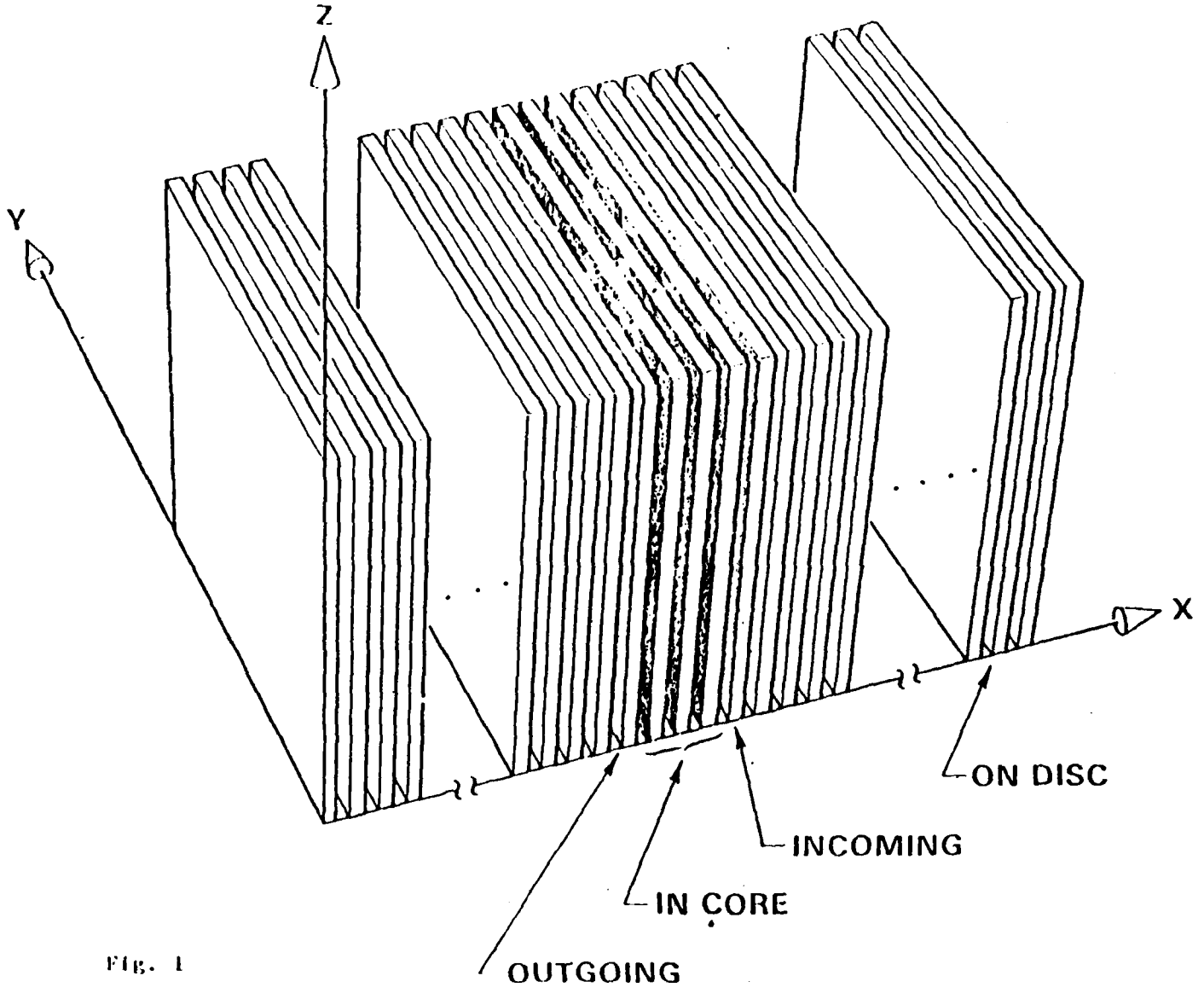
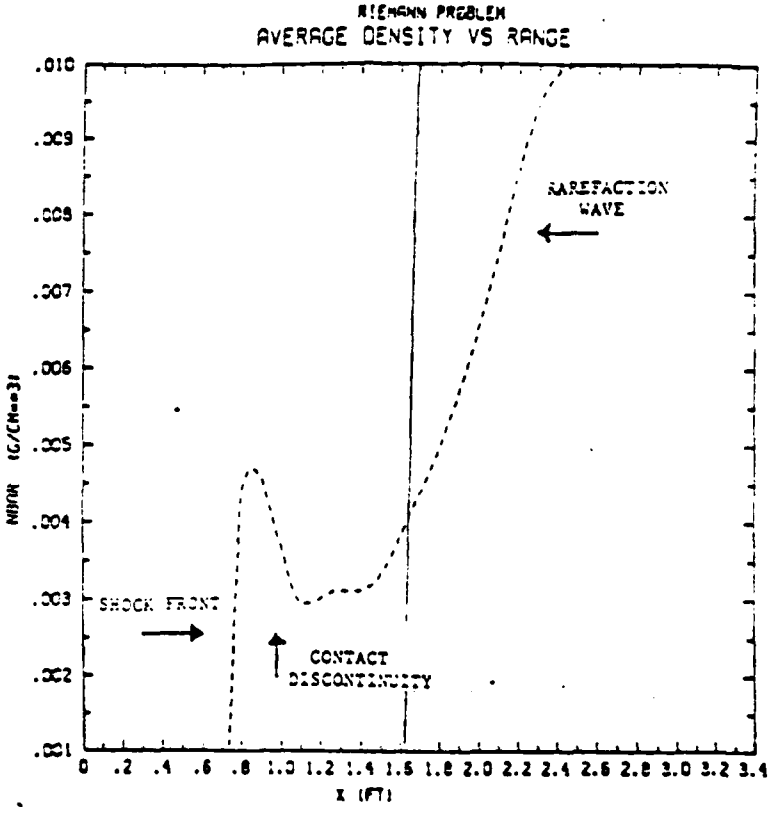


FIG. 1


RIEMANN PROBLEM
AVERAGE DENSITY VS RANGE
.010
.009
.008
.007
.006
.005
.004
.003
.002
.001
RAREFACTION WAVE
SHOCK FRONT
CONTACT DISCONTINUITY
0 .2 .4 .6 .8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0 3.2 3.4
X (FT)

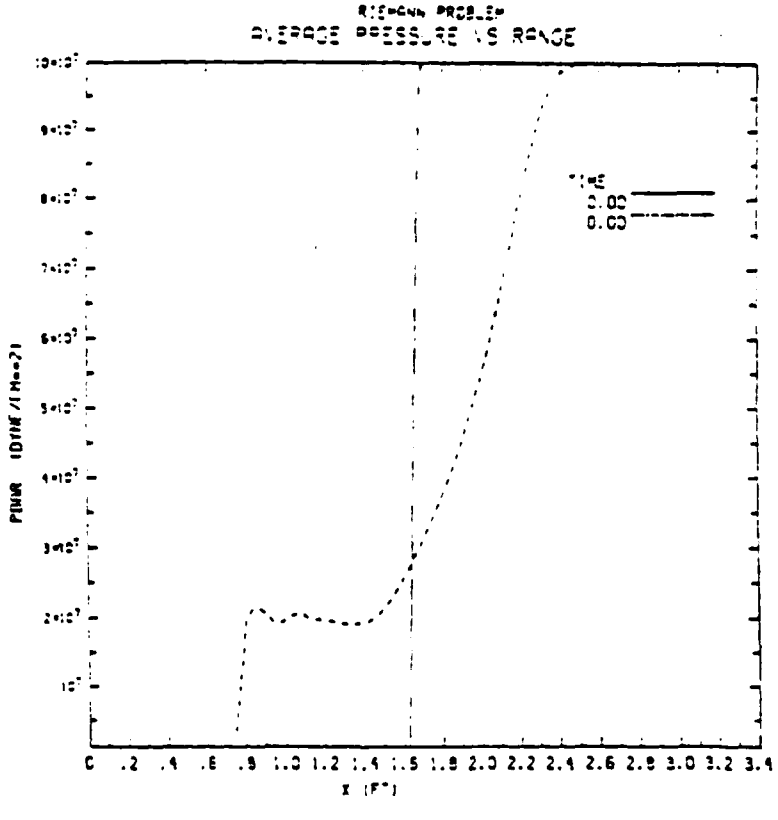

RIEMANN PROBLEM
AVERAGE PRESSURE VS RANGE
TIME
0.00
0.03
0 .2 .4 .6 .8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0 3.2 3.4
X (FT)

Fig. 2

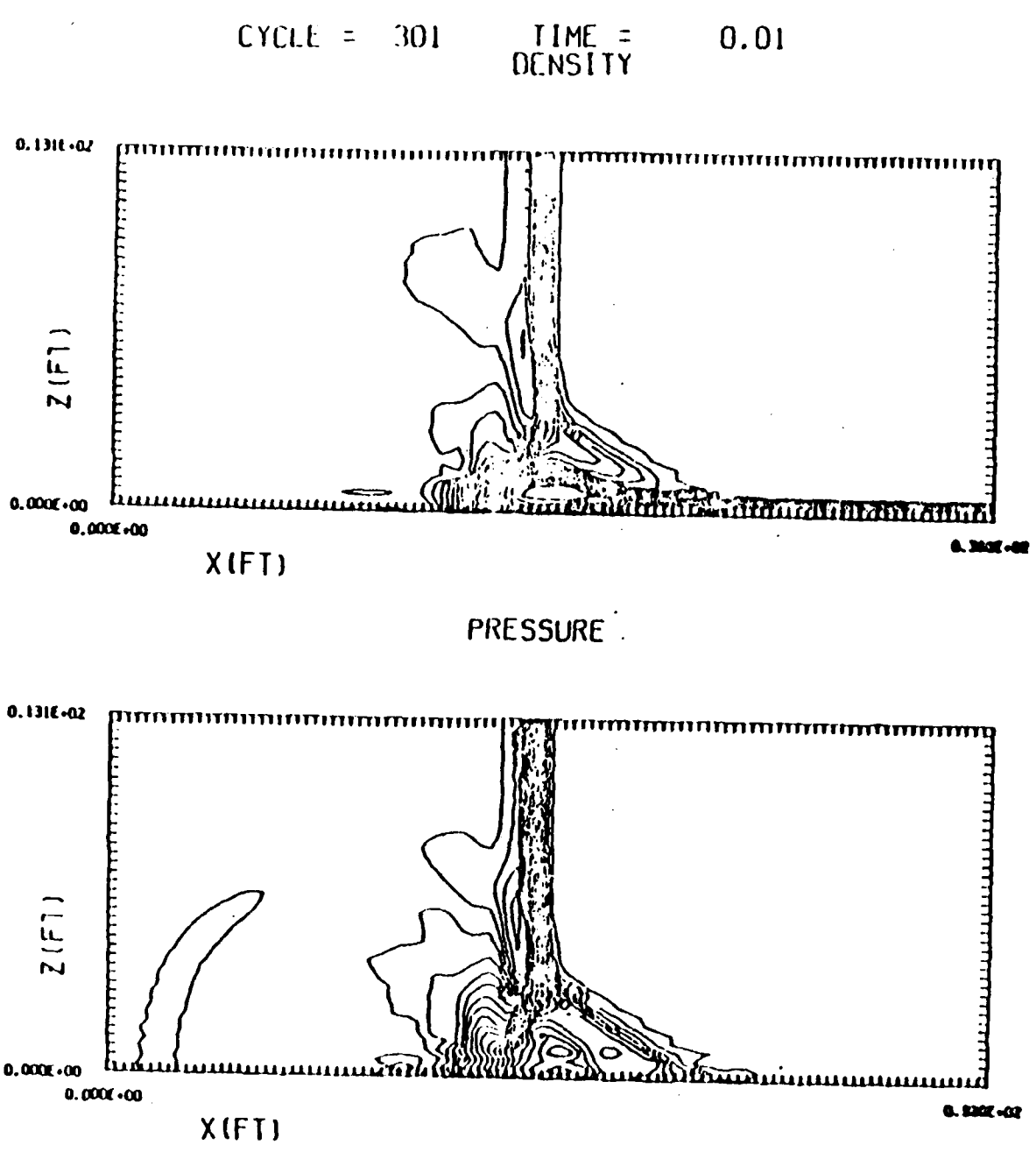

CYCLE = 301 TIME = 0.01
DENSITY
0.131E+02
Z(FT)
0.000E+00
0.000E+00
X(FT)
PRESSURE
0.131E+02
Z(FT)
0.000E+00
0.000E+00
X(FT)

Fig. 3

THREE - DIMENSIONAL GRAPHICS

Shock on Structure in Heated Layer

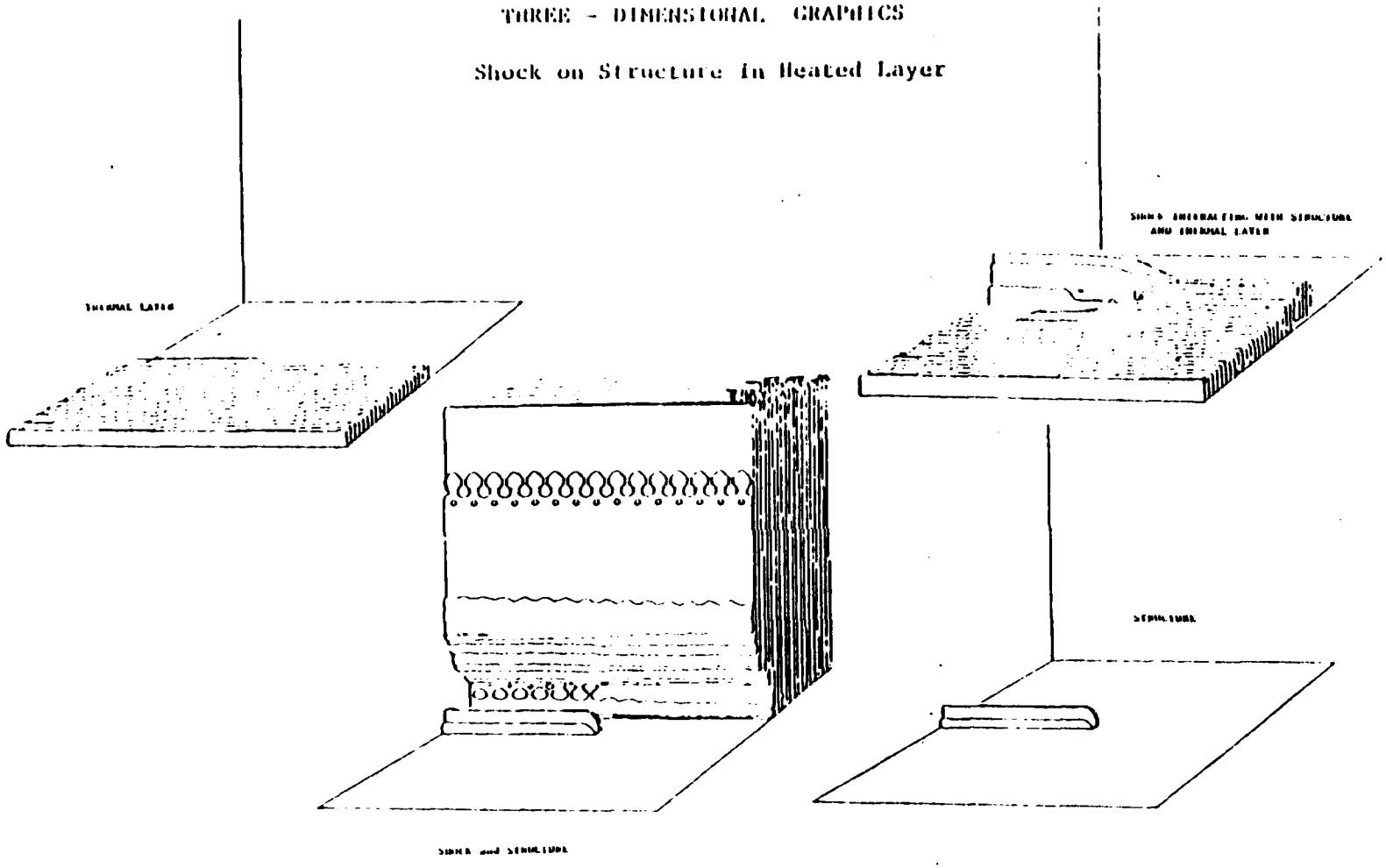


FIG. 4

APPENDIX D

COMPILATION LISTING OF

FLIK 3D

GRAPHICS CODE

THE KORAN OF FLIK3D

Flik3d, or Nplot3d as its current incarnation is named, is a graphics package created to plot the results of airblast simulations in 3 dimensions. The program is written in FORTRAN, and makes use of two different graphics libraries: NCAR (from the National Center of Atmospheric Studies) and DISSPLA (put out by ISSCO graphics software).

In its current form, nplot3d is capable of plotting fourteen different graphs, both contour and profile, plus history graphs showing the changes in various quantities over time. It can make use of calculations on both regular and irregular grids, showing all or part of a viewing window, in color or black and white.

The information specifying the plots to be drawn is read in from the file ndat, using namelists qdata and histry. The raw data to be plotted is read in in a series of dump files. The names of the dump files are listed in the file *ndump* in the order that they will be plotted. That is, ndump might contain the list

```
1 n20.d0001
2 n20.d0002
3 n20.d0003
  .
  .
  .
12 n20.d0012
```

Most of the information on the plot window, slices to be plotted, and so forth is in the namelist qdata. This namelist contains the following values:

nslice - this is the number of "slices" through the calculated region which are to be plotted. Usually up to three planes: x-y, y-z, and x-z plane.

msect - this is an array (presently from 1 to 3) of integers, holding the planes which are to be plotted. That is, if msect(1)=1, then the x-y plane will be the first to be plotted. Nplot3d will plot the first nslice slices in msect.

mswich - this is an array (14 by 3) of integers. Each value in mswich represents one plot. If the value of mswich(m,n) is zero, graph m will NOT be plotted for slice n. The graphs are, in order:

```
1 density profile (species 1, 2, and 3)
2 energy profile
3 dynamic pressure profile
4 pressure profile
5 renormalized density profile (scaled to ambient)
6 renormalized pressure profile (scaled to ambient)
7 z velocity (at right angles to current plane)
8 density contour (renormalized, scaled to ambient)
9 energy contour
10 pressure contour
11 mach number contour
12 velocity vector plot
13 trace particles plot (used to track small particles)
14 grid (the actual irregular grid used in the calculations)
```

For the profile plots, mswich's value specifies the profile to be drawn. If mswich is positive, the profile is plotted versus distance in the horizontal direction; if negative, in the vertical direction. The absolute value of mswich gives the value of the vertical or horizontal distance for

example, if mswich(1)=36, that means "plot density vs. distance in the z direction at x 36."

ilef -- ilef is an array (dimension 3) of integers, each giving the left most boundary of the region to be plotted in the x direction for a particular slice.

iriq -- iriq, similarly to ilef, gives the rightmost boundary in the x direction for each slice.

jfnt -- jfnt gives the "frontmost" boundary of the region to be plotted in the y direction for a particular slice.

jbak -- jbak gives the "backmost" boundary of the region in the y direction.

kbot -- kbot gives the lower boundary of the region to be plotted in the z direction for each slice.

ktop -- ktop gives the upper boundary of the region to be plotted in the z direction for each slice

nfiles -- gives the number of dump files to be plotted.

incore -- incore is a logical which specifies whether the calculations will be made entirely in core or not.

color -- this logical determines whether the contour plots will be plotted in color or black and white

lstat -- a logical, stating whether or not there are sensor stations present.

interp -- this logical should be true if the data was calculated on an uneven grid. The values will be interpolated onto an even grid for plotting.

dmin -- this is the minimum contour level for the contour plot of density. If dmin=dmax=0 then the program will choose its own, rounded minimum.

dmax -- this is the maximum contour level for the contour plot of density. See dmin.

displa -- displa is a logical which determines which graphics library will be used. If displa=.true. then the graphs will be plotted using DISSPLA. Otherwise, NCAR will be used.

nbu -- this is the number of bursts for which the particles should be plotted.

darkhd -- this logical is used in conjunction with the displa logical. If displa and darkhd are both true, then headings will be printed in a dark, bold type (swissbold). If darkhd is false, the headings will be in a simple line font. All NCAR headings are simple.

ilines -- this is used with the grid plot. Every ILINESth vertical line will be drawn.

jlines -- this is used with the grid plot. Every JLINESth horizontal line will be drawn.

There are a number of particular problems to look out for in modifying this code, as well as in running it. One particularly subtled problem is that, in compiling and loading the code, the DISSPLA library discft MUST be loaded first. The proper command would then be

```
rcft f nplot3d lib=(discft,ncarcft,cgscft,cftmath)

                    or

cft f nplot3d
ldr lib=(discft,ncarcft,cgscft,cftmath)
```

If this is not done, then conflicts arise between NCAR and DISSPLA commands with the same names.

Also, in running this program, only use darkhd=.true. if your object is to produce presentation quality plots. These plots are high in quality, but consume almost ten times as much cpu time as plots with darkhd=.false. If one is concerned with working plots, simply for your own reference, one should use NCAR, or DISSPLA with darkhd=.false. And, if you run off plots with dark headings, print them out at the electrostatic printer or plotter

utility at Los Alamos. Otherwise, producing plots is too time consuming. Old age comes fast, sitting in front of a tektronix terminal.

Within the code itself are some peculiarities that must be watched out for. If you, for any reason, wish to add or take away plots, extensive changes must be made. First, change the declared dimension of mplot and mswitch EVERYWHERE in the program. Secondly, look for the do loops in conv3d and prof3d. They are of the form

```
do xxx ipm,n
```

These do loops, and the assigned gotos that go with them, must be altered for the programs to function properly. Also, after the section of code which actually draws the plots, there is a line which checks to see if other species of density are to be drawn. This line, too, makes reference to ip.

```
if (ispec .le. nspec .and. ip-7 .eq. 1) go to 50
```

The particle plot is another potential trouble source. At present, it is hard-wired to trace the particles from up to 2 bursts. In order to increase the number of bursts, one must alter several declarations at the beginning of Conv3d. The data statements assigning ichar and imark must be changed; the parameter nburst must be increased.

Another frequent change will have to be made in the parameter statements. Every time the grid is changed, you must change the parameters nnnx, nnny, nnnz, ndm1, ndm2, and nhv. Nnnx, nnny, and nnnz are the dimensions of the 3D grid used in calculations, and ndm1 and ndm2 are the dimensions of the 2D slices.

Several general points to mention:

1. Parameters are declared in many places in this code. Make sure to change them everywhere.
2. Line numbering is erratic at best in conv3d and prof3d. If you must add a line number, check to make sure that it is not already used.
3. Similarly, those routines make use of many small variables, in small do loops and such. All such variables are listed in the following listing, after the routines in which they occur. Before you add a variable, check to make sure that it has not been used for something else.
4. Two different space arrays have been provided for use. If you need to modify some variable (pressure, density, or whatever), USE THE SPACE ARRAYS! DO NOT OVERWRITE THE MAIN ARRAYS! These arrays are used in other calculations, and changes will cause the other plots to be invalid.
5. This code is long, tangled, and obscure. The comments are sometimes inadequate. Be careful, it is easy to miss some needed changes and cause errors that are very difficult to find.

The output of this program will be a cgs metafile titled "plot." This metafile can be viewed on a tektronix terminal using pscan or ptekt, or printed out on the electrostatic printer using pesp, or printed to film using pfilm. Good luck with this program... you may well need it!

Todd Brun
19 August 1986

Errata

At the end of every plot file will be a blank page, present for reasons unknown. Ignore it, it's harmless.

Every velocity vector plot will be followed by a blank page. This is a

small bug (a feature, now that I've documented it) which is caused by the use of the DISSPLA and NCAR calls in combination, and cannot be easily corrected. As it is also harmless, I have not tried to correct it. Thus, if one requests all fourteen plots for two dump files, one slice each, the plot file will contain 31 pages: 28 plots, 2 blank pages after the velocity vector plots, and one blank page at the end. If you try it and get 3 pages, something is wrong.

```
 1          program flik3d (tty,output tty)
       c- -----------------------------------------------------------
       c     main program for FAST3D graphics
       c- -----------------------------------------------------------
       c        Flik3d is a graphics package designed to print several different
       c  plots from information stored in a dumpfile, produced by an airblast
       c  simulation program.  These plots are produced by two major routines,
       c  Conv3d (which produces contour plots, velocity vector plots, particle
       c  plots, and a template grid), and Prof3d (which produces various profile
       c  plots), supported by a number of other routines.
       c        The graphics are drawn using two different graphics libraries:
       c  DISSPLA (by ISSCO) and NCAR.  The library used is controlled by a
       c  logical switch 'displa', which is read in in the namelist gdata and
       c  passed in the common block grfcom.
       c        The specifications of the plots are input from the file ndat.  Most
       c  of these are passed in the namelist gdata.  The array mswich specifies
       c  which plots are to be drawn, while the switch displa determines which
       c  library is to be used.  Nslice holds the plane which is to be plotted:
       c  x y, y z, or x z.  For full explanation of the ndat file, see the
       c  Flik3d Koran.
       c  To run correctly, the source code must be compiled and loaded correctly.
       c  The proper format is
       c     rcft i=nplot3d lib=(discft,ncarcft,cgscft,cftmath)
       c  The discft library must be loaded first, in order for the program to run
       c  correctly.
       c        Many parts of this code are hardwired; they cannot be changed
       c  easily without altering the code in several places.  Several examples
       c  are listed in the Koran.  Several important ones:  the number and order
       c  of the plots is controlled by the array mplot.  If more plots are added,
       c  be sure to alter mplot's dimension EVERYWHERE.  Also, one must alter
       c  a number of do statements (of the form do xxx ip=yyy,zzz) which control
       c  the order of plot calls.
       c        Furthermore, the size of the page and the subplot area is set, in
       c  the routine Conv3d.  Great care should be exercised in changing them, as
       c  this will require that one re-scale a number of, other calls, such as
       c  the SET calls which precede the velocity vector plot.
       c        Also, the line numbering is irregular, at best.  Check to make sure
       c  that you have not duplicated any existing line numbers.
       c        Good luck with these graphics!!
 2          character *40 label
 3          character *8 readf, printf, namef, restf
 4          data readf /'ndat'/, printf /'nout'/, namef /'ndump'/
       c- ------------------- station arrays ---------------------------
 5          parameter (mxx=1, nsta=1)
 6          parameter (mdmp=1, mdxx=mdmp*mxx)
 7          logical lstat
 8          real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta)
 9          real prs(mxx,nsta), tme(mxx), xs(nsta), ys(nsta), zs(nsta)
 10         common /stats/ lstat, rhs, vls, gms, prs, tme, xs, ys, zs, nxx
 11         real tim(mdxx), psta(mdxx,nsta), dsta(mdxx,nsta)
 12         real star(mdxx,nsta)
 13         pointer (ipstar,star)
       c- ------------------- local declarations -----------------------
 14         parameter (npar=3)
 15         logical interp(npar), incore, lintp, colour, ladj, displa
 16         logical darkbd
```

```
      17.            integer mswich(14,npar), msect(npar), ilef(npar)
      18.            integer irig(npar), jfnt(npar), jbak(npar), kbot(npar)
      19             integer ktop(npar), mplot(14), ilines(npar), jlines(npar)
      20             namelist /qdata/ nslice,msect,mswich,ilef,irig,jfnt,jbak,
     1     kbot,ktop,nfiles,incore,colour,lstat,interp,dmin,dmax,
     2     displa,nbu,ilines,jlines,darkhd
      21.            namelist /history/ ista,jsta,ichose,ladj
      22.            common /qfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
     1     mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
      23             common /nbur/ nbu
      24.            data ilines, jlines /npar*1, npar*1/
c
c     open files.
      25.            open (5,file=readf,status='old')
      26             open (6,file=prntf,status='new')
      27.            open (7,file namef,status 'old')
c
c     read in namelist data.
c     this data specifies the types of plots to be drawn.
      28             read (5,qdata)
c
c     initialize graphics and NCAR or DISSPLA.
      29.            call gplot (ibu,4hplot,12)
      30             if (displa) then
      31               call libdisp
      32             else
      33.              call libncar
      34.            endif
c
      35             if (lstat) go to 120
c
c     plot contours and velocity vectors.
c     ***********************************
      36             do 110 if 1,nfiles
      37.            read (7,270,end=110) restf
      38             call restrt (istep,incore,restf,label)
c
c     count number of characters in label.
      39             nlabel=0
      40             do 10 i 1,40
      41             nlabel=nlabel+1
      42.            if (label(i 1) eq '$') go to 20
     43     10      continue
     44.    20      continue
      45             if (incore) call readin
c
c     process one slice at a time.
c     the data controlling the appearance of the plots is read in from
c     the file ndat in namelist qdata.
      46             do 100 lslice=1,nslice
      47             mslice msect(lslice)
      48             imin=ilef(lslice)
      49             imax=irig(lslice)
      50             jmin=jfnt(lslice)
      51             jmax=jbak(lslice)
      52             kmin=kbot(lslice)
```

```
 53             kmax ktop(islice)
 54.            il=ilines(islice)
 55.            jl jlines(nslice)
 56.            lintp=interp(islice)
 57             do 30 ip=1,14
 58.      30    mplot(ip)=mswich(ip,islice)
       c
       c    fill the plot arrays and plot.
 59.            go to (40,60,80), mslice
       c
       c    X Y slice.
       c    Reads in the data and calls the routines to draw profile
       c    and contour plots.
 60       40    do 50 kk=kmin,kmax
 61             call datain (imin,imax,jmin,kk)
 62.            call prof3d (istep,restf,label,nlabel,kk)
 63.            call conv3d (istep,restf,label,nlabel,kk)
 64.      50    continue
 65             go to 100
       c
       c    Y Z slice.
       c    Reads in the data and calls the routines to draw profile
       c    and contour plots.
 66       60    do 70 ii=imin,imax
 67.            call datain (ii,ii,jmin,kmin)
 68             call prof3d (istep,restf,label,nlabel,ii)
 69             call conv3d (istep,restf,label,nlabel,ii)
 70       70    continue
 71.            go to 100
       c
       c    X Z slice.
       c    Reads in the data and calls routines to plot profile
       c    and contour plots.
 72       80    do 90 jj=jmin,jmax
 73             call datain (imin,imax,jj,kmin)
 74             call prof3d (istep,restf,label,nlabel,jj)
 75             call conv3d (istep,restf,label,nlabel,jj)
 76.      90    continue
 77.     100    continue
       c
       c    close dump
 78.            call filefc
 79       110   continue
 80             go to 260
       c
       c    plot station time histories
       c    ..........................
 81       120   continue
 82             read (5,histry)
 83             kt 0
 84             kount=0
 85             kfile 0
 86             do 210 if=1,nfiles
       c
       c    read dump file name and read data from dump file.
 87             read (7,270,end=210) restf
```

```
                call restrt (istep,incore,restf,label)
                call filefc
                kfile=kfile+1
      c
      c     determine number of characters in label.
                nlabel=0
                do 130 i=1,40
                nlabel=nlabel+1
                if (label(i:i) .eq.'$') go to 140
       130      continue
       140      continue
      c
      c     time
                do 150 i=1,nxx
       150      tim(i+kt)=tme(i)
                kt=kt+nxx
      c
      c     patch time histories.
                k=kount
                do 200 ksta=ista,jsta
                istat=ksta-ista+1
                go to (160,180,160), ichose
      c
      c     static pressure.
       160      do 170 i=1,nxx
       170      psta(i+k,istat)=prs(i,ksta)
                if (ichose.eq.1) go to 210
      c
      c     dynamic pressure.
       180      do 190 i=1,nxx
       190      dsta(i+k,istat)=0.5*rhs(i,ksta)*vls(i,ksta)*vls(i,ksta)
       200      continue
       210      kount=k+nxx
      c
      c     plot time histories.
                do 250 ksta=ista,jsta
                istat=ksta-ista+1
                xstat=xs(ksta)
                ystat=ys(ksta)
                zstat=zs(ksta)
                length=kt
                tmax=tim(kt)
                if (ladj) call adjust (tim,psta(1,istat),kt,tmax,length)
                go to (220,230,220), ichose
      c
      c     static pressure.
       220      ipstat=loc(psta)
      c     separate calls are made to label axes with DISSPLA and NCAR
                if (displa) then
                 call xname ('TIME (SEC)',10)
                 call yname ('PRESSURE (DYNES/CM**2)',22)
                else
                 call anotat(11htime, sec $,22hpressure, dynes/cm**2$,1,1,0,0)
                endif
                icall=1
                go to 240
```

```
             c
             c     dynamic pressure.
  129.  230        ipstar=loc(dsta)
  130.             if (displa) then
  131.               call xname ('TIME (SEC)',10)
  132.               call yname ('DYNAMIC PRESSURE (DYNES/CM**2)',30)
  133.             else
  134.               call anotat(11htime, sec $,30hdynamic pressure, dynes/cm**2$,
            1          1,1,0,0)
  135.             endif
  136.             icall=2
             c
             c     generate plots.
             c     Set up the titles and axes for the plots using DISSPLA or NCAR.
  137.  240        if (displa) then
  138.               imx=ismax(length,star(1,istat),1)
  139.               call page (10.,11.)
  140.               call area2d (6.,8.)
  141.               encode (30,280,glab) star(imx,istat),tim(imx)
  142.               call headin (glab,30,1.5,4)
  143.               encode (32,300,glab) kfile,restf
  144.               call headin (glab,32,1.5,4)
  145.               encode (48,310,glab) ksta,xstat,ystat,zstat
  146.               call headin (glab,48,1.5,4)
  147               encode (nlabel,290,glab) label(1:nlabel)
  148.               if (darkbd) then
  149.                 call swissm
  150.                 call shdchr (90.,1,0.003,1)
  151.               endif
  152.               call headin (glab,nlabel,2.,1)
  153               call dsprt (tim,star(1,istat),length)
  154.               if (icall.eq.1.and.ichose.eq.3) go to 230
  155.             else
  156.               imx=ismax(length,star(1,istat),1)
  157               encode (30,280,glab) star(imx,istat),tim(imx)
  158.               call pwrit (700,950,glab,30,1,0,0)
  159.               encode (32,300,glab) kfile,restf
  160               call pwrit (550,30,glab,32,1,0,0)
  161               encode (48,310,glab) ksta,xstat,ystat,zstat
  162               call pwrit (550,50,glab,48,1,0,0)
  163.              encode (nlabel,290,glab) label(1:nlabel)
  164.              call ezxy (tim,star(1,istat),length,glab)
  165.              if (icall.eq.1.and.ichose.eq.3) go to 230
  166             endif
  167   250       continue
             c
             c     terminate graphics.
  168.  260          call gdone
             c
             c     close files.
  169.             close (5)
  170.             close (6)
  171.             close (7)
  172.             stop
             c
  173.  270 format (1x,a)
```

```
    281    174.    280 format ('peak ',e10.3,' at ',e10.3,'$')
    282    175.    290 format (a)
    283    176.    300 format (12,' dumps, last dump is ',a,'$')
    284    177.    310 format ('station',i3,', x,y,z ',3e10.3)
    285    178         end
FLIK3D          SHORT VECTOR LOOP BEGINS AT SEQ. NO.          57, P=     551b
FLIK3D                VECTOR LOOP BEGINS AT SEQ. NO.          97, P=    1013b
FLIK3D                VECTOR LOOP BEGINS AT SEQ. NO.         104, P=    1047a
FLIK3D                VECTOR LOOP BEGINS AT SEQ. NO.         107, P=    1070b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBE					
10 UNDEF**	43L	40E			
10A530b	40L				
10BUNDEF**	40I				
20 544c	44I	42J			
30 UNDEF**	58L	57E			
30AUNDEF**	57L				
30BUNDEF**	57L				
40 601b	60I	59J			
50 UNDEF**	64L	60E			
50A610b	60L				
50B635a	60L				
60 635c	66I	59J			
70 UNDEF**	70L	66F			
70A644c	66L				
70B670d	66I				
80 671b	72I	59J			
90 UNDEF**	76L	72E			
90A700b	72L				
90B725a	72L				
100 725a	77L	71J	65J	46E	
100A551b	46I				
100B727d	46L				
110 730d	79L	37R	36E		
110A500.1	36L				
110B733c	36L				
120 734a	81L	35J			
130 UNDEF**	95I	92E			
130A777a	92L				
130BUNDEF**	92I				
140 1013b	96I	94J			
150 UNDEF**	98I	97E			
150A1023a	97L				
150B1031b	97I				
160 1047a	104I	103J/2			
170 UNDEF**	105L	104F			
170A1057d	104L				
170B1066d	104I				
180 1070b	107L	103J			
190 UNDEF**	108L	107I			
190A1100b	107L				
190B1112a	107L				
200 UNDEF**	109L	101E			
200A1037a	101L				
200B1114d	101I				
210 1114d	110L	106J	87R	86E	
210A744c	86L				
210B1121b	86I				
220 1150c	120I	119J/2			
230 1174c	165J	154J	129I	119J	
240 1220b	137L	128J			

```
   250 UNDEF**    167I     111E
   250A1124b      111L
   250B1527c      111L
   260 1527c      168L      80J
   270 FN         173L      87R      37R
   280 FN         174L     157R     141R
   290 FN         175I     163R     147R
   300 FN         176I     159R     143R
   310 FN         177I     161R     145R
 00002 473b        30I
 00003 474b        32L
 00004 546c        45I
 00005 577d        59W
 00006 1045c      103W
 00007 1142c      118L
 00008 1147a      119W
 00009 1165b      121L
 00010 1173b      124L
 00011 1211b      130I
 00012 1217b      133L
 00013 1367d      137I
 00014 1350c      148L
 00015 1524d      155L
```

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES							
	$CLS		EXTERNAL		174U	170U	169U					
	$EFA		EXTERNAL		163U	159U	147U	143U				
	$EFF		EXTERNAL		163U	161U	159U	157U	147U	145U	143U	141U
	$EFI		EXTERNAL		163U	161U	159U	157U	147U	145U	143U	141U
	$EFV		EXTERNAL		161U/4	159U	157U/2	145U/4	143U	141U/2		
	$OPN		EXTERNAL		27U	26U	25U					
	$RLA		EXTERNAL		87U	37U						
	$RLF		EXTERNAL		87U	37U						
	$RLI		EXTERNAL		87U	37U						
	$RNL		EXTERNAL		82U	28U						
	$STOP		EXTERNAL		172U							
	ADJUST		EXTERNAL		118U							
	ANGLAT		EXTERNAL		134U	125U						
	AREA2D		EXTERNAL		140U							
26	COLOUR	*I	VARIABLE	GRFCOM	22D	20S	15D					
	CONV3D		EXTERNAL		75U	69U	63U					
34	DARKHD	I	VARIABLE	GRFCOM	148U	22D	20S	16D				
	DATAIN		EXTERNAL		73U	67U	61U					
31	DISPLA	I	VARIABLE	GRFCOM	137U	130U	121U	30U	22D	20S	15D	
	DISPRF		EXTERNAL		151U							
30	DMAX	*R	VARIABLE	GRFCOM	22D	20S						
27	DMIN	*R	VARIABLE	GRFCOM	22D	20S						
147	DSTA	R	2DIM ARRAY		129R	108S	11D					
	EZXY		EXTERNAL		164U							

	FILLIC		EXTERNAL		89U	78U							
427	FLIK3D		ENTRY		1D/2								
317	GDATA		NAMELIST		28P	20P							
	GDONE		EXTERNAL		168U								
316	GLAB	R	VARIABLE		164P	163S	162P	161S	160P	159S	158P	157S	152P
					147S	146P	145S	144P	143S	142P	141S		
	GPLOT		EXTERNAL		29U								
	HEADIN		EXTERNAL		152U	146U	144U	142U					
407	HISTRY		NAMELIST		82P	21S							
273	I	I	VARIABLE		108U/4	107I	105U/2	104I	98U/2	97I	94U/2	92I	42U/2
					40I								
314	ICALL	I	VARIABLE		165U	154U	136S	127S					
267	ICROSE	I	VARIABLE		165U	154U	119P	106U	103P	21S			
270	IF	I	VARIABLE		86I	36I							
277	II	I	VARIABLE		69P	68P	67P/2	66I					
32	II	*I	VARIABLE	GRFCOM	54S	22D							
233	IIEF	I	1DIM ARRAY		48U	20S	17D						
255	IILINES	I	1DIM ARRAY		54U	24S	20S	19D					
2	IMAX	I	VARIABLE	GRFCOM	73P	66N	61P	49S	22D				
1	IMIN	I	VARIABLE	GRFCOM	73P	66N	61P	48S	22D				
315	IMX	I	VARIABLE		157U/2	156S	141U/2	138S					
154	INCORE	L	VARIABLE		88P	45U	38P	20S	15D				
151	INTERP	L	1DIM ARRAY		56U	20S	15D						
275	IP	I	VARIABLE		58U/2	57I							
150	IPSTAR	*I	VARIABLE		129S	120S	13P						
236	IRIG	I	1DIM ARRAY		49U	20S	18D						
	ISMAX	I	EXTERNAL		156U	138U							
265	ISTA	I	VARIABLE		112U	111N	102U	101N	21S				
306	ISTAT	I	VARIABLE		164U	157U	156U	153U	141U	138U	118U	112S	108U
					105U	102S							
271	ISTEP	I	VARIABLE		88P	75P	74P	69P	68P	63P	62P	38P	
244	JBAK	I	1DIM ARRAY		51U	20S	18D						
241	JENT	I	1DIM ARRAY		50U	20S	18D						
300	JJ	I	VARIABLE		75P	74P	73P	72I					
33	JJ	*I	VARIABLE	GRFCOM	55S	22D							
260	JJLINES	I	1DIM ARRAY		95U	24S	20S	19D					
4	JMAX	I	VARIABLE	GRFCOM	72N	51S	22D						
3	JMIN	I	VARIABLE	GRFCOM	72N	67P	61P	50S	22D				
266	JSTA	I	VARIABLE		111N	101N	21S						
304	K	I	VARIABLE		110U	108U	105U	100S					
247	KBOT	I	1DIM ARRAY		52U	20S	18D						
303	KFILE	I	VARIABLE		159U	143U	90U	90S	85S				
276	KK	I	VARIABLE		63P	62P	61P	60I					
6	KMAX	I	VARIABLE	GRFCOM	60N	53S	22D						
5	KMIN	I	VARIABLE	GRFCOM	73P	67P	60N	52S	22D				
302	KOUNT	I	VARIABLE		110S	100U	84S						
305	KSTA	I	VARIABLE		161U	145U	115U	114U	113U	112U	111I	108U/3	105U
					102U	101I							
301	KT	I	VARIABLE		118P	117U	116U	99U	99S	98U	83S		
252	KTOP	I	1DIM ARRAY		53U	20S	19D						
134	LABEL	CH	VARIABLE		163U	147U	94U	88P	75P	74P	69P	68P	63P
					62P	42U	38P	2D					
155	LADJ	L	VARIABLE		118U	21S	15D						
312	LENGTH	I	VARIABLE		164P	156P	153P	138P	118P	116S			
	LIBDISP		EXTERNAL		31U								
	LIBNCAR		EXTERNAL		33U								

25	LINTP	*I	VARIABLE	GRFCOM	56S	22D	15D					
	LOC	I	INLINE FUNCT		129U	120U						
274	LSLICE	I	VARIABLE		58U	56U	54U	53U	52U	51U	50U	49U	48U
					47U	46I						
0	LSTAT	I	VARIABLE	STATS	35U	20S	10D	7D				
	MAIN		ENTRY									
	MDMP	I	PARAMETER		6S							
	MDXX	I	PARAMETER		12P	11P/3	6S					
7	MPLOT	*I	1DIM ARRAY	GRFCOM	58S	22D	19D					
230	MSECT	I	1DIM ARRAY		47U	20S	17D					
0	MSLICE	I	VARIABLE	GRFCOM	59P	47S	22D					
156	MSWICH	I	2DIM ARRAY		58U	20S	17D					
	MXX	I	PARAMETER		9P/2	8P/3	5S					
143	NAMEF	CH	VARIABLE		27P	4S	3D					
0	NBU	*I	VARIABLE	NBUR	23D	20S						
264	NFILES	I	VARIABLE		86N	36N	20S					
272	NLABEL	I	VARIABLE		163U	163P	152P	147U	147P	93U	93S	91S	75P
					74P	69P	68P	63P	62P	41U	41S	39S	
	NPAR	I	PARAMETER		24S/2	19P/3	18P/4	17P/3	15P	14S			
263	NSLICE	I	VARIABLE		55U	46N	20S						
	NSTA	I	PARAMETER		12P	11P/2	9P/4	8P/3	5S				
11	NXX	I	VARIABLE	STATS	110U	107N	104N	99U	97N	10D			
	PAGE		EXTERNAL		139U								
142	PRNTF	CH	VARIABLE		26P	4S	3D						
	PROF3D		EXTERNAL		74U	68U	62U						
	PROGRAM		EXTERNAL		1U								
4	PRS	R	2DIM ARRAY	STATS	105U	10D	9D						
146	PSTA	R	2DIM ARRAY		120P	118P	105S	11D					
	PWRIT		EXTERNAL		162U	160U	158U						
141	READF	CH	VARIABLE		25P	4S	3D						
	READIN		EXTERNAL		45U								
144	RESTF	CH	VARIABLE		159U	143U	88P	87S	75P	74P	69P	68P	63P
					62P	38P	37S	3D					
	RESTRT		EXTERNAL		88U	38U							
1	RHS	R	2DIM ARRAY	STATS	108U	10D	8D						
	SHDCHR		EXTERNAL		150U								
1	STAR	R	2DIM ARRAY	POINTEE	164P	157U	156P	153P	141U	138P	13P	12D	
	SWISSM		EXTERNAL		149U								
145	TIM	R	1DIM ARRAY		164P	157U	153P	141U	118P	117U	98S	11D	
313	TMAX	R	VARIABLE		118P	117S							
5	TME	R	1DIM ARRAY	STATS	98U	10D	9D						
2	VLS	R	2DIM ARRAY	STATS	108U/2	10D	8D						
	XNAME		EXTERNAL		133U	122U							
6	XS	R	1DIM ARRAY	STATS	113U	10D	9D						
307	XSTAT	R	VARIABLE		161U	145U	113S						
	YNAME		EXTERNAL		132U	123U							
7	YS	R	1DIM ARRAY	STATS	114U	10D	9D						
310	YSTAT	R	VARIABLE		161U	145U	114S						
10	ZS	R	1DIM ARRAY	STATS	115U	10D	9D						
311	ZSTAT	R	VARIABLE		161U	145U	115S						

TABLE OF EXTERNAL NAMES

$CLS	EXTERNAL	171U	170U	169U	
$ELA	EXTERNAL	161U	159U	147U	143U

$EFF	EXTERNAL	163U	161U	159U	157U	147U	145U	143U	141U
$EFI	EXTERNAL	163U	161U	159U	157U	147U	145U	143U	141U
$EFV	EXTERNAL	161U/4	159U	157U/2	145U/4	143U	141U/2		
$OPN	EXTERNAL	27U	26U	25U					
$REA	EXTERNAL	87U	37U						
$REF	EXTERNAL	87U	37U						
$REI	EXTERNAL	87U	37U						
$RNL	EXTERNAL	82U	28U						
$STOP	EXTERNAL	172U							
ADJUST	EXTERNAL	118U							
ANOTAT	EXTERNAL	134U	125U						
AREA2D	EXTERNAL	140U							
CONV3D	EXTERNAL	75U	69U	63U					
DATAIN	EXTERNAL	73U	67U	61U					
DISPRF	EXTERNAL	153U							
EZXY	EXTERNAL	164U							
FILEFC	EXTERNAL	89U	78U						
GDONE	EXTERNAL	168U							
GPLOT	EXTERNAL	29U							
HEADIN	EXTERNAL	152U	146U	144U	142U				
ISMAX I	EXTERNAL	156U	138U						
LIBDISP	EXTERNAL	31U							
LIBNCAR	EXTERNAL	33U							
PAGE	EXTERNAL	139U							
PROF3D	EXTERNAL	74U	68U	62U					
PROGRAM	EXTERNAL	1U							
PWRIT	EXTERNAL	162U	160U	158U					
READIN	EXTERNAL	45U							
RESTRF	EXTERNAL	88U	38U						
SHDCHR	EXTERNAL	150U							
SWISSM	EXTERNAL	149U							
XNAME	EXTERNAL	131U	122U						
YNAME	EXTERNAL	132U	123U						

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

MDMP	= 1	MDXX	= 1
MXX	= 1	NPAR	3
NSTA	= 1		

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
110	IF	36	79	500a	234
10	I	40	43	530b	15
100	LSLICE	46	77	551b	157
30	IP	57	58		INLINE
50	KK	60	64	610b	25
70	II	66	70	644c	25
90	JJ	72	76	700b	25
210	IF	86	110	744c	155
130	I	92	95	777a	14
150	I	97	98	1023a	6
200	KSTA	101	109	1037a	56
170	I	104	105	1057d	7
190	I	107	108	1100b	12
250	KSTA	111	167	1124b	404

BLOCK NAMES AND LENGTHS IN OCTAL

1544-FLIK3D 53-#IB 517-#CL 12-STATS 35-GRFCOM 1-NBUR

STATIC SPACE (IN OCTAL)

B SAVE:	31
T SAVE:	4
CONSTANTS:	132
VARIABLES:	165
TEMPORARIES:	535
CODE:	1225
TOTAL:	2336

```
  1.        subroutine conv3d (istep,restf,label,nlabel,lplane)
      c-----------------------------------------------------------------
      c    graphics for FAST3D.
      c    Conv3d plots, at the present time, 7 distinct graphs.  The graphs to
      c  plotted have a non zero value in their positions in the array mplot.
      c  The values mplot(8) through mplot(14) refer to the 7 plots in this
      c  subroutine.
      c    The plots are, in order, a contour plot of renormalized density,
      c  a contour plot of energy/volume, a contour plot of pressure (absolute)
      c  a contour plot of mach number, a velocity vector plot, a plot of trace
      c  particles used in the calculation, and a template grid to show the real
      c  spacing of the irregular grid used in the calculations.
      c    Flik3d can handle both regularly and irregularly spaced grids.  If
      c  an irregularly spaced grid is used, the flag interp will be set,
      c  and a call will be made to the subroutine conint to interpolate onto an
      c  regular grid.
      c    Also, contour plots can be done in color.  If colour is specified,
      c  the plots will be drawn by the routine colcon.  Otherwise, they will be
      c  drawn within conv3d itself, using either DISSPLA or NCAR calls, and
      c  calling on the shorter routine discon for DISSPLA.
      c      Warning:  a large array in blank common is used by the DISSPLA
      c  routine CONMAK.  Do not use blank common for anything without allowing
      c  for that block.
      c------------------ array dimensions ---------------------------
  2.        parameter (nnnx=72, nnny=33, nnnz=88)
  3.        parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
  4.        parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
      c------------------ plot variables ----------------------------
  5.        parameter (ndm1=72, ndm2=88)
  6.        real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
  7.        real erg(ndm1,ndm2)
  8.        real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
  9.        common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
      c ---------- fixed arrays of hydrodynamic data ------------------
 10.        parameter (nspec=1, nsum=4*nspec)
 11.        real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 12.        real sumpl(nsum,nnnx)
 13.        real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
      c
 14.        common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
          1     grapz, sum4, shrink, fsky, ratio
      c---------------- grid arrays ---------------------------------
 15.        real xcor(nnnxp), ycor(nnny), zcor(nnnz)
 16.        real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
 17.        real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
      c
 18.        common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
          1     , dtody2, dtodz2
      c------------------ global variables ---------------------------
 19.        common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
          1     , nstep, lllx, llly, lllz
      c------------------------- scratch space -----------------------
 20.        parameter (nnyz2d=nnny2*nnnz2, nvar=4*nspec)
 21.        parameter (ndm12=(ndm1+2)*(ndm2+2))
 22.        parameter (ndm1p2=ndm1+ndm2)
 23.        real dumv(ndm12,nvar), dumy(ndm12)
```

```
      common /holder/ space(ndm1,ndm2),valmin,valrng,
     +     space2(ndm1,ndm2)
      common /locom1/ dumv, dumy
c ----------------------------- pointers -----------------------------
      real arr(ndm1,ndm2), hcor(ndm1), vcor(ndm2)
      real h(ndm1p2,2), v(ndm1p2,2)
      real arh(ndm1,ndm2), arv(ndm1,ndm2)
      pointer (iparr,arr), (iphcor,hcor), (ipvcor,vcor)
      pointer (iparh,arh), (iparv,arv)
      pointer (ipminh,minh), (ipmaxh,maxh)
      pointer (ipminv,minv), (ipmaxv,maxv)
c --------------------- local declarations ----------------------------
      real pre(ndm1,ndm2), mac(ndm1,ndm2), gam(ndm1,ndm2)
      real eint(ndm1), scr1(ndm1,ndm2), scr2(ndm1,ndm2)
      equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
      equivalence (scr1,dumv(1,4)), (scr2,dumv(1,5)), (eint,dumy)
      real rscr(ndm1), pscr(ndm1), gscr(ndm1)
      real scra(ndm1), scrb(ndm1), scrc(ndm1)
      equivalence (rscr,scr1), (pscr,scr1(1,2)), (gscr,scr1(1,3))
      equivalence (scra,scr1(1,4)), (scrb,scr1(1,5)), (scrc,scr2)
c ---------------------------particles--------------------------------------
c   The particles are set up for two bursts only; changing the number of
c   bursts could cause an error in data statements for ichar and imark.
        parameter (nop=70,nburst=2)
        parameter (npart1=3*nop*nburst+nburst)
        real xp(nop,nburst), zp(nop,nburst)
        integer nopp(nburst), nchk(nop,nburst)
        common /part1/ xp,zp,nopp,nchk
      common /nbur/ nbu
c
      integer mplot(14)
      character *40 label
      character *8 restf
      integer ichar(nburst),imark(nburst)
      real glab1(7), glab2(7), glab3(7), glab4(7)
      logical lintp, colour, displa, darkhd
      data ksh, ksv /30,30/
      data ichar/8hx       ,8ho       /,imark/4,16/
      namelist /grfdat/ mslice,imin,imax,jmin,jmax,kmin,kmax,mplot
     1   ,lintp,colour
      common /grfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
     1   mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
      common /bound/ hmin, hmax, vmin, vmax
c
c   first print labels.
      encode (nlabel,250,glab1) label(1:nlabel)
      encode (40,260,glab2) time,istep,restf
c
c   set window sizes.
      go to (10,20,30), mslice
c
c   set up the pointers to specify the coordinates of the given slice
c   X Y slice.
   10 continue
      ipminh loc(imin)
      ipmaxh loc(imax)
```

```
398     64.             ipminv=loc(jmin)
399     65.             ipmaxv=loc(jmax)
400     66.             iphcor=loc(xcor)
401     67.             ipvcor=loc(ycor)
402     68.             iparh=loc(rvx)
403     69.             iparv=loc(rvy)
404     70.               if (displa) then
405     71.               encode (28,270,glab3) lplane,zcor(lplane)/100000.
406     72.             else
407     73.               encode (32,275,glab3) lplane,zcor(lplane)
408     74              endif
409     75.             go to 40
410             c
411             c       Y Z slice.
412     76.     20      continue
413     77.             ipminh=loc(jmin)
414     78.             ipmaxh=loc(jmax)
415     79.             ipminv=loc(kmin)
416     80.             ipmaxv=loc(kmax)
417     81.             iphcor=loc(ycor)
418     82.             ipvcor=loc(zcor)
419     83.             iparh=loc(rvy)
420     84.             iparv=loc(rvz)
421     85.             if (displa) then
422     86.               encode (28,280,glab3) lplane,xcor(lplane)/100000.
423     87.             else
424     88.               encode (32,285,glab3) lplane,xcor(lplane)
425     89.             endif
426     90.             go to 40
427             c
428             c       X-Z slice.
429     91.     30      continue
430     92.             ipminh=loc(imin)
431     93.             ipmaxh=loc(imax)
432     94.             ipminv=loc(kmin)
433     95.             ipmaxv=loc(kmax)
434     96              iphcor=loc(xcor)
435     97.             ipvcor=loc(zcor)
436     98.             iparh=loc(rvx)
437     99.             iparv=loc(rvz)
438    100.             if (displa) then
439    101.               encode (28,290,glab3) lplane,ycor(lplane)/100000.
440    102.             else
441    103.               encode (32,295,glab3) lplane,ycor(lplane)
442    104.             endif
443             c
444             c       define plot window.
445             c       what section of the plane is to be drawn?
446    105.     40      continue
447    106.             hmin=hcor(minh)
448    107.             hmax=hcor(maxh)
449    108.             vmin=vcor(minv)
450    109.             vmax=vcor(maxv)
451    110.             lh=maxh-minh+1
452    111             lv=maxv-minv+1
453    112             ni = lh / 11
```

```
454   113.            nj = lv / jl
455   114.            nij = ni * nj
456   115.            kh=lh
457   116.            kv=lv
458   117.            if (lintp) then
459   118.            delh=hmax-hmin
460   119.            delv=vmax-vmin
461   120.            kh=ndm1
462   121.            dhv=delh/float(kh-1)
463   122.            kv=1+int(delv/dhv)
464   123.            if (kv.le.ndm2) go to 45
465   124.            kv=ndm2
466   125.            dhv=delv/float(kv-1)
467   126.            kh=1+int(delh/dhv)
468   127.            endif
469   128.     45     call wsize (hmin,hmax,vmin,vmax,hlo,hhi,vlo,vhi)
470         c
471         c     generate contour and vector plots.
472   129.            ispec=1
473   130.            do 230 ip=8,14
474   131.            if (mplot(ip).eq.0) go to 230
475         c     for each plot, set up the page and titles.
476         c     The DISSPLA commands set the type of print, the size of the page,
477         c     the size of the plot area, frame the axes and label them, write the
478         c     titles, and prepare for drawing the plots.  The NCAR commands draw
479         c     the titles, set the size of the subplot area, frame it and draw
480         c     tickmarks.
481   132.     50     if (displa) then
482   133.              if (dackhd) then
483   134.                call swissm
484   135.                call shdchr (90.,1.0.003,1)
485   136               endif
486   137.              call page (10.,11.)
487   138.              call nobrdr
488   139.              call area2d (8.*(hmax-hmin)/(vmax-vmin),8.)
489   140.              call height (.16)
490   141.              call headin (glab1,nlabel-1,1.6,3)
491   142.              call headin (glab2,2%,1.6,3)
492   143.              call headin (glab3,27,1.6,3)
493   144.              call xname (' ',1)
494   145.              call yname (' ',1)
495   146.              call thkfrm (.02)
496   147.              call frame
497   148.              ystep = (vmax - vmin)/8.
498   149.                xstep = ystep
499   150.              call graf (hmin/100000.,xstep,hmax/100000.,
500               +                vmin/100000.,ystep,vmax/100000.)
501   151.              flo=0
502   152.              fhi=0
503   153.            else
504   154.              call pwrit (500,1010,glab1,nlabel,1,0,0)
505   155.              call pwrit (350,980,glab2,40,1,0,0)
506   156.              call pwrit (500,950,glab3,28,1,0,0)
507   157.              call set (hlo,hhi,vlo,vhi,hmin,hmax,vmin,vmax,1)
508   158.              call perim (1,0,1,0)
509   159.              flo=0
```

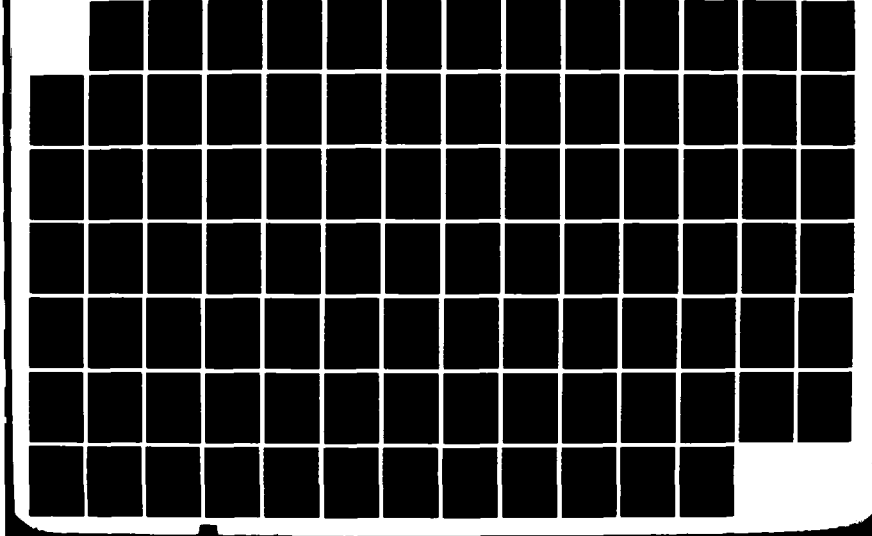
```
510   160.              fhi=0.
511   161.            endif
512         c
513   162             go to (160,100,110,140,170,400,420), ip-7
514         c
515         c     renormalized (!!!!) density contours.
516         c     these plots are scaled w/respect to the ambient density.  Three
517         c     species of density can be drawn (from ispec).  These are usually
518         c     total density, dust density, and explosive density.  Usually only
519         c     species 1 or species 1 and 2 are used.
520   163     60      go to (70,80,90), ispec
521         c     first, assign the pointer
522   164     70      ipart=loc(space2)
523   165             do 72 nv=minv,maxv
524   166             rscr(nv) cvmgt(rcamb(iplane),rcamb(nv+1),mslice.eq.1)
525   167             do 72 nh minh,maxh
526   168             space2(nh,nv)=rho(nh,nv)/rscr(nv)
527   169     72      space2(nh,nv)=cvmgt(1.0,space2(nh,nv),space2(nh,nv).gt.0.95
528               1     .and. rho(nh,nv).lt.1.05)
529   170             encode (18,300,glab4)
530         c     label the plot
531   171             if (displa) then
532   172               tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,17)/2.
533   173               call messag (glab4,17,tleng,-.75)
534   174             else
535   175               call pwrit (800,980,glab4,18,1,0,0)
536   176             endif
537   177             go to 160
538         c     first, assign the pointer (species 2)
539   178     80      ipart=loc(rha)
540   179             encode (20,310,glab4) ispec
541         c     label the plot
542   180             if (displa) then
543   181               tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
544   182               call messag (glab4,19,tleng,-.75)
545   183             else
546   184               call pwrit (800,980,glab4,20,1,0,0)
547   185             endif
548   186             flo=dmin
549   187             fhi=dmax
550   188             go to 160
551         c     first, assign the pointer (species 3)
552   189     90      ipart=loc(rhb)
553   190             encode (20,310,glab) ispec
554         c     label the plot
555   191             if (displa) then
556   192               tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
557   193               call messag (glab4,19,tleng,-.75)
558   194             else
559   195               call pwrit (800,980,glab4,20,1,0,0)
560   196             endif
561   197             go to 160
562         c
563         c     energy contours.
564         c     first, assign pointer to the energy array
565   198   100       ipart=loc(erg)
```

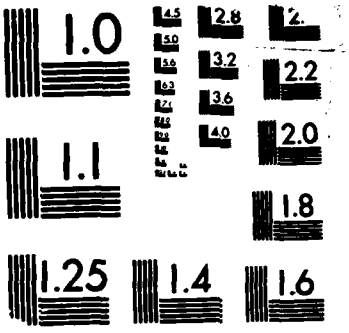
```
      199.            encode (20,320,glab4)
            c     label the plot
      200.            if (displa) then
      201.              tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
      202               call messag (glab4,19,tleng, .75)
      203.            else
      204.              call pwrit (800,980,glab4,20,1,0,0)
      205.            endif
      206.            go to 160
            c
            c     pressure contours.
            c     pressure is calculated in the equations of state routine eos
            c     (wst assign pointer to pressure array pre
      207.  110     ipar=loc(pre)
      208.            encode (23,330,glab4)
            c     label the plot
      209.            if (displa) then
      210.              tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,22)/2.
      211.              call messag (glab4,22,tleng, .75)
      212.            else
      213.              call pwrit (800,980,glab4,23,1,0,0)
      214.            endif
      215             do 130 nv=minv,maxv
      216             do 120 nh minh,maxh
      217.  120     eint(nh)=erg(nh,nv)-0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy
                1     (nh,nv)+rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
      218             call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
                1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
                2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
      219   130     continue
      220             go to 160
            c
            c     mach number contours.
            c     first assign pointer to mach number array mac
      221   140     ipar=loc(mac)
                    encode (19,340,glab4)
            c     label the plot
                    if (displa) then
                      tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,11)/2.
                      call messag (glab4,11,tleng,-.75)
                    else
                      call pwrit (800,980,glab4,13,1,0,0)
                    endif
                  c     calculate pressure from eos to use in calculating mach number
                        do 146 nv minv,maxv
                        do 145 nh minh,maxh
                        eint(nh,nv)=erg(nh,nv)-0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy
                        1     (nh,nv)+rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
                        call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
                        1     ,nv),pscr(minh),rscr(minh),pscr(minh),gscr(minh),scra(minh)
                        2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))

                              minv maxv
                                nh maxh
                                  mac(nh,nv)=rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
                                  1     +rvz(nh,nv)*rvz(nh,nv))/(gam(nh,nv)*pre(nh,nv)))
```

AD-A193 152 NUMERRCAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS 3/3
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

UNCLASSIFIED F/G 19/11 NN





```
237.          go to 160
238.    160   continue
        c
        c     plot contours.
        c     this section actually draws the contour lines.  If lintp is true, the
        c     irregular grid of values is interpolated onto a regular grid.  If
        c     colour is true, the routine colcon draws contours in color.
        c     The DISSPLA contours are actually draw by the subroutine discon, the
        c     NCAR contours by the routine conrec.
239.          if (displa) then
240.            if (lintp) then
241.              call conint (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
              1     lv,scr1,kh,kv)
242.              if (colour) then
243.                call colcon (scr1,1,1,kh,kv)
244.              else
245.                call discon (scr1,ndm1,space,kh,kv)
246.              endif
247.            else
248.              if (colour) then
249.                call colcon (arr,minh,minv,kh,kv)
250.              else
251.                call discon (arr(minh,minv),ndm1,space,kh,kv)
252.              endif
253.            endif
254.          else
255.            if (lintp) then
256.              call conint (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
              1     lv,scr1,kh,kv)
257.              if (colour) then
258.                call colcon (scr1,1,1,kh,kv)
259.              else
260.                call conrec (scr1,ndm1,kh,kv,flo,fhi,0,-1,0,0)
261.              endif
262.            else
263.              if (colour) then
264.                call colcon (arr,minh,minv,kh,kv)
265.              else
266.                call conrec(arr(minh,minv),ndm1,kh,kv,flo,fhi,0,-1,0,0)
267.              endif
268.            endif
269.            call framen
270.          endif
        c
        c     go back and draw the other species of density, if any
271.          ispec=ispec+1
272.          if (ispec.le.nspec.and.ip-7.eq.1) go to 50
273.          go to 230
        c
        c     plot velocity vectors.
        c     the velocity vector plots are plotted using the NCAR routine velvct,
        c     from information in arrays of h and v velocity components.
274.    170   encode (18,350,glab4)
        c     label the plot
275.          if (displa) then
276             xleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,17)/2
```

```
678   277.                call messag (glab4,17,lleng,-.75)
679   278.              else
680   279.                call pwrit (800,980,glab4,18,1,0,0)
681   280.              endif
682         c
683         c     first calculate velocities from momenta.
684   281.              do 180 nv=minv,maxv
685   282.              do 180 nh=minh,maxh
686   283.              pre(nh,nv)=1.0/rho(nh,nv)
687   284.              arh(nh,nv)=arh(nh,nv)*pre(nh,nv)
688   285.    180       arv(nh,nv)=arv(nh,nv)*pre(nh,nv)
689         c
690         c     plot vectors. interpolate onto uniform grid if lintp=.true.
691   286.              velmax=0.0
692   287.              velmin=1.e50
693   288.              if (lintp) then
694   289.              call conint (arh(minh,minv),hcor(minh),vcor(minv),ndm1,lh,lv
695               1     ,scr1,ksh,ksv)
696   290.              call conint (arv(minh,minv),hcor(minh),vcor(minv),ndm1,lh,lv
697               1     ,scr2,ksh,ksv)
698   291.              do 200 nv=1,ksv
699   292.              do 190 nh=1,ksh
700   293.    190       pre(nh,nv)=scr1(nh,nv)*scr1(nh,nv)+scr2(nh,nv)*scr2(nh,nv)
701   294               velmax=amax1(velmax,pre(ismax(ksh,pre(1,nv),1),nv))
702   295               velmin=amin1(velmin,pre(ismin(ksh,pre(1,nv),1),nv))
703   296.    200       continue
704   297.              velmax=sqrt(velmax)
705   298               velmin=sqrt(velmin)
706   299.              velmin=velmin+0.05*(velmax-velmin)
707         c     draw the velocity vectors.  Only NCAR is made use of (scaled to fit
708         c     the DISSPLA axes, if necessary), as only NCAR has a velocity plotting
709         c     routine.  libncar is called to initialize NCAR graphics, set to scale
710         c     the plot, if displa=.true.
711   300.              if (displa) then
712   301.                call libncar
713   302.                xa=(1.0 - .8*(hmax-hmin)/(vmax-vmin))/2. + 0.025
714   303.                xb=1.05 - xa
715   304.                call set (xa,xb,.110,.838,hmin,hmax,vmin,vmax,1)
716   305.                call velvct (scr1,ndm1,scr2,ndm1,ksh,ksv,velmin,velmax,-1,
717               1       0,0,0,0)
718   306.              else
719   307.                call velvct (scr1,ndm1,scr2,ndm1,ksh,ksv,velmin,velmax,-1,
720               1       0,0,0,0)
721   308.              endif
722   309               else
723   310.              do 220 nv=minv,minv+kv-1
724   311.              do 210 nh=minh,minh+kh-1
725   312.    210       pre(nh,nv)=arh(nh,nv)*arh(nh,nv)+arv(nh,nv)*arv(nh,nv)
726   313.              velmax=amax1(velmax,pre(ismax(kh,pre(minh,nv),1)+minh-1,nv))
727   314.              velmin=amin1(velmin,pre(ismin(kh,pre(minh,nv),1)+minh-1,nv))
728   315.    220       continue
729   316.              velmax=sqrt(velmax)
730   317.              velmin=sqrt(velmin)
731   318.              velmin=velmin+0.05*(velmax-velmin)
732         c     plot the vector plot (as described above), using uninterpolated grid.
733   319.              if (displa) then
```

```
 734   320.                call libncar
 735   321.                xa=(1.0 - .8*(hmax-hmin)/(vmax-vmin))/2. + .025
 736   322.                xb=1.05 - xa
 737   323.                call set (xa,xb,.110,.838,hmin,hmax,vmin,vmax,1)
 738   324.                call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
 739         1     ,velmin,velmax,1,0,0,0.0)
 740   325.              else
 741   326.                call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
 742         1     ,velmin,velmax,-1,0,0,0.0)
 743   327.              endif
 744   328.              endif
 745   329.              call framen
 746   330.              if (displa) call endpl (0)
 747   331.              go to 230
 748   332.              return
 749         c
 750         c     particles
 751         c
 752   333.  400         continue
 753   334.              encode (11,360,glab4)
 754         c       using either NCAR or DISSPLA, the plot is labeled, and markers are
 755         c       placed at points showing the positions of the trace particles.  A
 756         c       different marker is used for each burst, from the arrays ichar or
 757         c       imark, respectively.
 758   335.              if (displa) then
 759   336.                tleng = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,10)/2.
 760   337.                call messag (glab4,10,tleng,-0.75)
 761   338.                do 405 k=1,nbu
 762   339                call marker (imark(k))
 763   340.                do 406 ijk = 1, nopp(k)
 764   341.                xp(ijk,k) = 0.00001 * xp(ijk,k)
 765   342                 zp(ijk,k) = 0.00001 * zp(ijk,k)
 766   343.   406          continue
 767   344.                call curve (xp(1,k),zp(1,k),nopp(k),-1)
 768   345.   405          continue
 769   346.                call endpl(0)
 770   347.              else
 771   348.                call pwrit (800,980,glab4,11,1,0,0)
 772   349.                write (6,1021)
 773   350.                do 410 k=1,nbu
 774   351.                do 410 i=1,nopp(k)
 775   352.                call points(xp(i,k),zp(i,k),1,ichar(k),0)
 776   353.                write (6,1022) i,xp(i,k),zp(i,k)
 777   354.  410           continue
 778   355.                call framen
 779   356.              endif
 780   357.              go to 230
 781         c
 782         c          grid
 783         c       this draws the irregular grid on which the original calculation was
 784         c       done.  this serves to show where the resolution was finest, and where
 785         c       the most interpolation was done, on the contour plots.
 786         c
 787   358.   420      continue
 788         c
 789   359              do 421 i = 1, nn
```

```
      790    360.            h(i,1) = hcor(minh + ii * i - 1)
      791    361.            h(i,2) = h(i,1)
      792    362.     421    continue
      793           c
      794    363.            do 422 j = 1, nj
      795    364.            jj = j * ni
      796    365.            v(jj,1) = vcor(minv + jl * j - 1)
      797    366.            v(jj,2) = v(jj,1)
      798    367.     422    continue
      799           c
      800    368.            do 423 i = 1, ni
      801    369.            v(i,1) = v(ni+1,1)
      802    370.            v(i,2) = v(nij,1)
      803    371.     423    continue
      804           c
      805    372.            do 424 j = ni+1, nij
      806    373.            h(j,1) = h(1,1)
      807    374.            h(j,2) = h(ni,1)
      808    375.     424    continue
      809           c
      810    376.            if( nij .lt. ndm1p2 ) then
      811    377.              do 425 j = nij + 1, ndm1p2
      812    378.              v(j,1) = v(nij,1)
      813    379.              v(j,2) = v(nij,2)
      814    380.              h(j,1) = h(nij,1)
      815    381.              h(j,2) = h(nij,2)
      816    382.     425      continue
      817    383.            endif
      818           c
      819           c     draw the grid itself
      820    384.            if (displa) then
      821    385.              do 426 j=1,ndm1p2
      822    386.              space(1,1)=h(j,1)/100000.
      823    387.              space(2,1)=h(j,2)/100000.
      824    388               space(1,2)=v(j,1)/100000.
      825    389.              space(2,2)=v(j,2)/100000.
      826    390.              call curve (space(1,1),space(1,2),2,0)
      827    391.     426      continue
      828    392.              call endpl (0)
      829    393.            else
      830    394.              call anotat (lh$,lh$,1,1,0,0)
      831    395.              call agseti( ' win ', 1 )
      832    396.              call agsetf( ' grid/left. ', hlo )
      833    397.              call agsetf( ' grid/right. ', hhi )
      834    398.              call agsetf( ' grid/bottom. ',vlo )
      835    399.              call agsetf( ' grid/top. ', vhi )
      836    400.              call agsetf( ' x/minimum. ', h(1,1) )
      837    401.              call agsetf( ' x/maximum. ', h(ni,1) )
      838    402.              call agsetf( ' y/minimum. ', v(ni+1,1) )
      839    403.              call agsetf( ' y/maximum. ', v(nij,1) )
      840    404.              call agsetf( ' x/nice. ', 0. )
      841    405.              call agsetf( ' y/nice. ', 0. )
      842    406.              call agseti( ' row. ', 2 )
      843    407.              call ezmxy( h, v, ndm1p2, ndm1p2, 2, 0 )
      844    408.            endif
      845           c
```

```
 846   409.  230      continue
 847   410            return
 848         c
 849         c     this entry is a fossilized routine, implemented but not used in the
 850         c     current version of the code.  Ignore it, or better still, rip it out,
 851         c     if you have the time and inclination.
 852   411.            entry grfin(ipl)
 853         c ---- -------- -- - -------- -- ---------------- -- -- ------------
 854         c     reads in data for in-line graphics.
 855         c ---- -- -- -- -------- - ---------------- - ------------
 856   412.            read (5,grfdat)
 857   413.            write (6,grfdat)
 858   414.            if (mslice.eq.1) iplane=kmin
 859   415.            if (mslice.eq.2) iplane=imin
 860   416.            if (mslice.eq.3) iplane=jmin
 861   417.            ipl=iplane
 862   418.            do 240 nv=1,ndm2
 863   419.            do 240 nb=1,ndm1
 864   420.            rho(nb,nv)=cvmgt(rcamb(ipl+1),rcamb(nv+1),mslice.eq.1)
 865   421.            rvx(nb,nv)=0.0
 866   422.            rvy(nb,nv)=0.0
 867   423.            rvz(nb,nv)=0.0
 868   424.   240      erg(nb,nv)=cvmgt(ecamb(ipl+1),ecamb(nv+1),mslice.eq.1)
 869   425.            call gplot (ihu,4hplot,12)
 870   426.            if (displa) then
 871   427.              call libdisp
 872   428.            else
 873   429.              call libncar
 874   430.            endif
 875   431.            return
 876         c
 877   432.            entry grfend
 878         c --------- -- - ---------------------------------------------- ---
 879         c     terminates NCAR or DISSPLA for inline graphics.
 880         c -- -------- -- -- - -------   ------------------------------ -------
 881   433.            call gplone
 882   434.            return
 883         c
 884         c     formats, and other important goodies, for the titles and labels.
 885   435.   250 format (a)
 886   436.   260 format ('time=',f5.1,' sec, step',i5,', dump ',a)
 887   437.   270 format ('X Y plane ',i3,' at z=',f5.1,' km$')
 888   438.   275 format ('X Y plane ',i3,' at z=',e9.3,' cm$')
 889   439.   280 format ('Y Z plane ',i3,' at x=',f5.1,' km$')
 890   440.   285 format ('Y Z plane ',i3,' at x=',e9.3,' cm$')
 891   441.   290 format ('X Z plane ',i3,' at y=',f5.1,' km$')
 892   442.   295 format ('X Z plane ',i3,' at y=',e9.3,' cm$')
 893   443.   300 format ('density (g/cm**3)$')
 894   444.   310 format ('density ',i1,' (g/cm**3)$')
 895   445.   320 format ('energy (ergs/cm**3)$')
 896   446.   330 format ('pressure (dynes/cm**2)$')
 897   447.   340 format ('mach number$')
 898   448.   350 format ('velocity (cm/sec)$')
 899   449.   360 format ('particles $')
 900   450.  1021 format (/1x,'part       xp              zp')
 901   451.  1022 format (1x,i3,2(2x,1pe14.7))
```

```
CONV3D    PAGE 12                ON:ACDEILPQRSTVXZ          08/18/86  MX-V  12:07:27      CFT114b (05/16/86)  PAGE 24

    902    452.        end
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          167, P=    2374a
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          216, P=    2677b
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          230, P=    3035c
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          235, P=    3134a
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          282, P=    3423a
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          292, P=    3526b
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          311, P=    3661b
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          340, P=    4076b
O   AT SEQUENCE NUMBER -    352.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "XP" IN SEQUENCE NUMBER 353      ********** P=0250630D
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
O   AT SEQUENCE NUMBER -    352.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "ZP" IN SEQUENCE NUMBER 353      ********** P=0250630D
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          359, P=    4220a
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          363, P=    4241d
O   AT SEQUENCE NUMBER -    369.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "V" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER -    369.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370      ********** P=0250630D
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          368, P=    4266a
 CONV3D  CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.          368, P=    4266a
O   AT SEQUENCE NUMBER -    373.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "H" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER -    373.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374      ********** P=0250630D
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER -    378.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "V" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER -    379.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "V" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER -    380.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "H" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D                VECTOR LOOP BEGINS AT SEQ. NO.          377, P=    4340b
 CONV3D  CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.          377, P=    4340b
O   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D    COMMENT -      DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 386      ********** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
```

```
0   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D   COMMENT -      DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 387      ************************** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
0   AT SEQUENCE NUMBER -    390.
 PRNAME CONV3D   COMMENT -      DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 388      ************************** P=0250305C
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
 CONV3D              VECTOR LOOP BEGINS AT SEQ. NO.          419. P=     4575b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE	PROGRAM REFERENCES					
10	1727c	61L	60J					
20	1775c	76I	60J					
30	2043c	91I	60J					
40	2111a	105I	90J	75J				
45	2162b	128L	123J					
50	2177b	272J	132L					
60	2360b	163I	162J					
70	2366b	164L	163J					
72	UNDEF**	169L	167E	165E				
72A	2374a	165L						
72B	2431d	165L						
72C	2415b	167L						
72D	2427a	167L						
80	2467d	178L	163J					
90	2533a	189L	163J					
100	2574b	198L	162J					
110	2634a	207I	162J					
120	UNDEF**	217L	216E					
120A	2707d	216L						
120B	2731a	216I						
130	UNDEF**	219L	215E					
130A	2677b	215L						
130B	2771d	215L						
140	2772b	221I	162J					
145	UNDEF**	231L	230E					
145A	3046a	230L						
145B	3067b	230I						
146	UNDEF**	233L	229E					
146A	3035c	229I						
146B	3130a	229L						
150	UNDEF**	236L	235E	234E				
150A	3134a	234L						
150B	3174a	234L						
150C	3144c	235I						
150D	3171b	235L						
160	3174c	238L	237J	220J	206J	197J	188J	177J
170	3361c	274I	162J					
180	UNDEF**	285I	282E	281E				
180A	3423a	281L						
180B	3455b	281L						
180C	3436c	282L						
180D	3452c	282L						
190	UNDEF**	293L	292E					
190A	3535c	292I						
190B	3546d	292I						
200	UNDEF**	296L	291E					
200A	3526b	291L						
200B	3572a	291L						
210	UNDEF**	312L	311E					
210A	3675c	311L						

```
210B3705d      311I
220 UNDEF**    315I    310E
220A3661b      310I
220B3736c      310I
230 4533b      409I    357J    331J *  273J    131J    130E
230A2175d      130I
230BUNDEF**    130I
240 UNDEF**    424L    419E    418E
240A4575b      418L
240BUNDEF**    418L
240C4611b      419L
240DUNDEF**    419L
250 FN         435L     58R
260 FN         436L     59R
270 FN         437L     71R
275 FN         438L     73R
280 FN         439L     86R
285 FN         440L     88R
290 FN         441L    101R
295 FN         442L    103R
300 FN         443L    170R
310 FN         444I    190R    179R
320 FN         445L    199R
330 FN         446L    208R
340 FN         447L    222R
350 FN         448L    274R
360 FN         449L    334R
400 4046a      333L    162J
405 UNDEF**    345L    338E
405A4076b      338L
405B4136b      338L
406 UNDEF**    343I    340E
406A4113b      340I
406B4124b      340L
410 UNDEF**    354L    351E    350E
410A4160a      350I
410B4216c      350L
410C4165b      351L
410D4213d      351L
420 4220a      358L    162J
421 UNDEF**    362L    359E
421A4232a      359L
421B4241d      359I
422 UNDEF**    367I    363E
422A4253d      363I
422B4266a      363I
423 UNDEF**    371L    368E
423A4302d      368L
423B4324a      368L
424 UNDEF**    375L    372E
424A4332d      372L
424B4340b      372L
425 UNDEF**    382I    377E
425A4355c      377L
425B4405a      377I
426 UNDEF**    391L    385E
```

```
   426A4411c      385I
   426RUNDEF **   385L
  1021 FN         450I    349W
  1022 FN         451I    353W
 00002 1726a       60W
 00003 1762d       70L
 00004 1775a       72I
 00005 2030d       85I
 00006 2043a       87I
 00007 2076d      100I
 00008 2111a      102L
 00009 2162b      117I
 00010 2302b      132L
 00011 2206d      133L
 00012 2350a      153L
 00013 2354d      162W
 00014 2364d      163W
 00015 2457d      171L
 00016 2467b      174I
 00017 2521a      180I
 00018 2530c      183I
 00019 2564b      191L
 00020 2573d      194I
 00021 2624a      200L
 00022 2633c      203I
 00023 2663d      209L
 00024 2673b      212L
 00025 3022a      223L
 00026 3031c      226I
 00027 3260c      239I
 00028 3236b      240L
 00029 3227d      242L
 00030 3235d      244L
 00031 3260a      247L
 00032 3245d      248L
 00033 3260a      250I
 00034 3354a      254L
 00035 3325b      255L
 00036 3312d      257L
 00037 3324d      259L
 00038 3353a      262L
 00039 3334d      263I
 00040 3353a      265I
 00041 3407c      275L
 00042 3417a      278L
 00043 3654c      288I
 00044 3640c      300L
 00045 3654a      306I
 00046 4037a      309I
 00047 4014b      319L
 00048 4037a      325I
 00049 4043a      330I
 00050 4140d      335I
 00051 4217c      347I
 00052 4405a      376L
 00053 4432b      384I
```

```
 00054 4533b      393L
 00055 4563b      414I
 00056 4566c      415I
 00057 4571d      416I
 00058 4637c      426I
 00059 4640c      428L
 00060 4313b      371E
 00061 4324a      371E
 00062 UNDEF**    371E
 00063 4316b      371E
 00064 4371d      382E
 00065 4405a      382E
 00066 UNDEF**    382E
 00067 4376a      382E
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
------------------------------
```

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES								
	$EFA		EXTERNAL		59U	58U							
	$EFF		EXTERNAL		334U	274U	222U	208U	199U	190U	179U	170U	103U
					101U	88U	86U	73U	71U	59U	58U		
	$EFI		EXTERNAL		334U	274U	222U	208U	199U	190U	179U	170U	103U
					101U	88U	86U	73U	71U	59U	58U		
	$EFV		EXTERNAL		190U	179U	103U/2	101U/2	88U/2	86U/2	73U/2	71U/2	59U/2
	$RNL		EXTERNAL		412U								
	$WFF		EXTERNAL		353U	349U							
	$WFI		EXTERNAL		353U	349U							
	$WFV		EXTERNAL		353U/3								
	$WNL		EXTERNAL		413U								
	AGSETF		EXTERNAL		406U	405U	404U	403U	402U	401U	400U	399U	398U
					397U	396U							
	AGSETI		EXTERNAL		395U								
	AMAX1	R	INLINE FUNCT.		313U	294U							
	AMIN1	R	INLINE FUNCT.		314U	295U							
	ANDTAT		EXTERNAL		394U								
	AREA2D		EXTERNAL		139U								
4	ARH	R	2DIM ARRAY	POINTEE	326P	324P	312U/2	289P	284U	284S	30P	28D	
1	ARR	R	2DIM ARRAY	POINTEE	266P	264P	256P	251P	249P	241P	29P	26D	
5	ARV	R	2DIM ARRAY	POINTEE	326P	324P	312U/2	290P	285U	285S	30P	28D	
	COLCON		EXTERNAL		264U	258U	249U	243U					
26	COLOUR	I	VARIABLE	GRFCOM	263U	257U	248U	242U	56D	55S	52D		
	CONINT		EXTERNAL		290U	289U	256U	241U					
	CONREC		EXTERNAL		266U	260U							
1645	CONV3D		ENTRY		432D	411D	1D/2						
	CURVE		EXTERNAL		390U	344U							
	CVMGT		INLINE FUNCT.		424U	420U	169U	166U					
34	DARKID	L	VARIABLE	GRFCOM	133U	56D	52D						
1560	DELH	R	VARIABLE		126P	121U	118S						
1561	DELV	R	VARIABLE		125U	122P	119S						
1562	DIV	R	VARIABLE		126U	125S	122U	121S					
	DISCON		EXTERNAL		251U	245U							

31	DISPLA	I	VARIABLE	GRFCOM	426U	384U	335U	330U	319U	300U	275U	239U	223U
					209U	200U	191U	190U	171U	132U	100U	85U	70U
					56D	52D							
30	DMAX	R	VARIABLE	GRFCOM	187U	56D							
27	DMIN	R	VARIABLE	GRFCOM	186U	56D							
264	FCAMB	R	1DIM ARRAY	HYDFIX	424P/2	14D	11D						
101024	FINI	R	1D EQ. ARRAY	LOCOM1	232P	231S	218P	217S	36D	34D			
	ENDPL		EXTERNAL		392U	346U	330U						
	EOS		EXTERNAL		232U	218U							
45100	FRG	R	2DIM ARRAY	PLTVAR	424S	231U	217U	198P	9D	7D			
	EZMXY		EXTERNAL		407U								
1574	FHI	R	VARIABLE		266P	260P	187S	160S	152S				
1573	FLO	R	VARIABLE		266P	260P	186S	159S	151S				
	FLOAT	R	INLINE FUNCT.		125U	121U							
	FRAME		EXTERNAL		147U								
	FRAMEN		EXTERNAL		355U	329U	269U						
32010	GAM	R	2D EQ. ARRAY	LOCOM1	236U	232P	218P	35D	33D				
	GDONE		EXTERNAL		433U								
1600	GLAB	R	VARIABLE		190S								
1513	GLAB1	R	1DIM ARRAY		154P	141P	58S	51D					
1522	GLAB2	R	1DIM ARRAY		155P	142P	59S	51D					
1531	GLAB3	R	1DIM ARRAY		156P	143P	103S	101S	88S	86S	73S	71S	51D
1540	GLAB4	R	1DIM ARRAY		348P	337P	336P	334S	279P	277P	276P	274S	227P
					225P	224P	222S	213P	211P	210P	208S	204P	202P
					201P	199S	195P	193P	192P	184P	182P	181P	179S
					175P	173P	172P	170S	51D				
	GPLOT		EXTERNAL		425U								
	GRAF		EXTERNAL		150U								
1612	GRFDAT		NAMELIST		413P	412P	55S						
4644	GRFEND		ENTRY		432D/2								
4542	GRFIN		ENTRY		411D/2								
47234	GSCR	R	1D EQ. ARRAY	LOCOM1	232P	218P	39D	37D					
276	H	R	2DIM ARRAY		407P	401P	400P	387U	386U	381U	381S	380U	380S
					374U	374S	373U	373S	361U	361S	360S	27D	
2	HCOR	R	1DIM ARRAY	POINTEE	360U	290P	289P	256P	241P	107U	106U	29P	26D
	HEADIN		EXTERNAL		143U	142U	141U						
	HEIGHT		EXTERNAL		140U								
1564	HHI	R	VARIABLE		397P	157P	128P						
1563	HLO	R	VARIABLE		396P	157P	128P						
1	HMAX	R	VARIABLE	BOUND	336U	323P	321U	304P	302U	276U	224U	210U	201U
					192U	181U	172U	157P	150P	139U	128P	118U	107S
					57D								
0	HMIN	R	VARIABLE	BOUND	336U	323P	321U	304P	302U	276U	224U	210U	201U
					192U	181U	172U	157P	150P	139U	128P	118U	106S
					57D								
1607	I	I	VARIABLE		370U	369U	368I	361U/2	360U/2	359I	353U/3	352U/2	351I
1507	ICHAR	I	1DIM ARRAY		352P	54S	50D						
1606	IJK	I	VARIABLE		342U/2	341U/2	340I						
32	IL	I	VARIABLE	GRFCOM	360U	112U	56D						
1511	IMARK	I	1DIM ARRAY		339P	54S	50D						
2	IMAX	I	VARIABLE	GRFCOM	93P	63P	56D	55S					
1	IMIN	I	VARIABLE	GRFCOM	415U	92P	62P	56D	55S				
	INT	I	INLINE FUNCT.		126U	122U							
1570	IP	I	VARIABLE		272U	162P	131U	130I					
1501	IPARH	*I	VARIABLE		98S	83S	68S	30P					
1476	IPARR	*I	VARIABLE		221S	207S	198S	189S	178S	164S	29P		

```
 1502 IPARV   *I  VARIABLE                 99S     84S     69S     30P
 1477 IPHCOR  *I  VARIABLE                 96S     81S     66S     29P
 1504 IPMAXI  *I  VARIABLE                 93S     78S     63S     31P
 1506 IPMAXV  *I  VARIABLE                 95S     80S     65S     32P
 1503 IPMINI  *I  VARIABLE                 92S     77S     62S     31P
 1505 IPMINV  *I  VARIABLE                 94S     79S     64S     32P
 1500 IPVCOR  *I  VARIABLE                 97S     82S     67S     29P
      ISMAX    I  EXTERNAL                 313U    294U
      ISMIN    I  EXTERNAL                 314U    295U
 1567 ISPEC    I  VARIABLE                 272U    271U    271S    190U    179U    163P    129S
    1 ISTEP    I  VARIABLE     DUM.ARG.    59U     1D
 1610 J        I  VARIABLE                 389U    388U    387U    386U    385I    381U    380U    379U    378U
                                           377I    374U    373U    372I    365U    364U    363I
 1611 JJ       I  VARIABLE                 366U/2  365U    364S
   33 JL       I  VARIABLE     GRFCOM      365U    113U    56D
    4 JMAX     I  VARIABLE     GRFCOM      78P     65P     56D     55S
    3 JMIN     I  VARIABLE     GRFCOM      416U    77P     64P     56D     55S
 1605 K        I  VARIABLE                 353U/2  352U/3  351N    350I    344U/3  342U/2  341U/2  340N    339U
                                           338I
 1556 KH       I  VARIABLE                 326P    324P    314P    313P    311N    266P    264P    260P    258P
                                           256P    251P    249P    245P    243P    241P    126S    121P    120S
                                           115S
    6 KMAX     I  VARIABLE     GRFCOM      95P     80P     56D     55S
    5 KMIN     I  VARIABLE     GRFCOM      414U    94P     79P     56D     55S
 1547 KSH      I  VARIABLE                 307P    305P    295P    294P    292N    290P    289P    53S
 1550 KSV      I  VARIABLE                 307P    305P    291N    290P    289P    53S
 1557 KV       I  VARIABLE                 326P    324P    310N    266P    264P    260P    258P    256P    251P
                                           249P    245P    243P    241P    125P    124S    123U    122S    116S
    3 LABEL   CH  VARIABLE     DUM.ARG.    58U     48D     1D
 1551 LH       I  VARIABLE                 290P    289P    256P    241P    115U    112U    110S
      LIBDISP     EXTERNAL                 427U
      LIBNCAR     EXTERNAL                 429U    320U    301U
   25 LINTP    L  VARIABLE     GRFCOM      288U    255U    240U    117U    56D     55S     52D
      LOC      I  INLINE FUNCT.            221U    207U    198U    189U    178U    164U    99U     98U     97U
                                           96U     95U     94U     93U     92U     84U     83U     82U     81U
                                           80U     79U     78U     77U     69U     68U     67U     66U     65U
                                           64U     63U     62U
    6 LPI      I  VARIABLE     DUM.ARG.    424U    420U    417S    411D
    5 LPLANE   I  VARIABLE     DUM.ARG.    417U    416S    415S    414S    166U    103U/2  101U/2  88U/2   86U/2
                                           73U/2   71U/2   1D
 1552 LV       I  VARIABLE                 290P    289P    256P    241P    116U    113U    111S
15004 MAC      R  2D EQ  ARRAY LOCOM1      236S    221P    35D     33D
      MAIN.       ENTRY
      MARKER      EXTERNAL                 339U
    7 MAXH     I  VARIABLE     POINTEE     282N    235N    232P    230N    218P    216N    167N    110U    107U
                                           31P
   11 MAXV     I  VARIABLE     POINTEE     281N    234N    229N    215N    165N    111U    109U    32P
      MESSAG      EXTERNAL                 337U    277U    225U    211U    202U    193U    182U    173U
    6 MINH     I  VARIABLE     POINTEE     360U    326U/2  324U/2  314U/2  313U/2  311N/2  290U/2  289U/2  282N
                                           266U    264P    256U/2  251U    249P    241U/2  235N    232U/2  230N
                                           218U/2  216N    167N    110U    106U    31P
   10 MINV     I  VARIABLE     POINTEE     365U    326U/2  324U/2  310N/2  290U/2  289U/2  281N    266U    264P
                                           256U/2  251U    249P    241U/2  234N    229N    215N    165N    111U
                                           108U    32P
    7 MPLOT    I  1DIM ARRAY   GRFCOM      131U    56D     55S     47D
    0 MSLICE   I  VARIABLE     GRFCOM      424P    420P    416U    415U    414U    166P    60P     56D     55S
```

0	NBU	I	VARIABLE	NBUR	350N	338N	46D					
	NBURST	I	PARAMETER		50P/2	44P/2	43P/2	41S				
	NDM1	I	PARAMETER		419N	326P/2	324P/2	307P/2	305P/2	290P	289P	266P	260P
					256P	251P	245P	241P	120U	38P/3	37P/3	34P/3	33P/3
					28P/2	26P/2	24P/2	8P/3	7P	6P/3	5S		
	NDM12	I	PARAMETER		23P/2	21S							
	NDM1P2	I	PARAMETER		407P/2	385N	377N	376U	27P/2	22S			
	NDM2	I	PARAMETER		418N	124U	123U	34P/2	33P/3	28P/2	26P/2	24P/2	8P/3
					7P	6P/3	5S						
1576	NH	I	VARIABLE		424U	423U	422U	421U	420U	419I	312U/5	311I	293U/5
					292I	285U/3	284U/3	283U/2	282I	236U/2	235I	231U/9	230I
					217U/9	216I	169U/4	168U/2	167I				
1553	NI	I	VARIABLE		402U	401U	374U	372N	369U	368N	364U	359N	114U
					112S								
1555	NIJ	I	VARIABLE		403U	381U	380U	379U	378U	377N	376U	372N	370U
					114S								
1554	NJ	I	VARIABLE		363N	114U	113S						
4	NLABEL	I	VARIABLE	DUM.ARG.	154P	141P	58U	58P	1D				
	NNNX	I	PARAMETER		12P	2S							
	NNNX2	I	PARAMETER		4S								
	NNNXP	I	PARAMETER		17P	16P	15P	3S					
	NNNY	I	PARAMETER		15P	2S							
	NNNY2	I	PARAMETER		17P	16P	4S						
	NNNYP	I	PARAMETER		3S								
	NNNZ	I	PARAMETER		15P	2S							
	NNNZ2	I	PARAMETER		17P	16P	13P/4	11P/3	4S				
	NNNZP	I	PARAMETER		3S								
	NNYZ2D	I	PARAMETER		20S								
	NOBRDR		EXTERNAL		138U								
	NOP	I	PARAMETER		44P	43P/2	41S						
430	NOPP	I	1DIM ARRAY	PARTL	351N	344P	340N	45D	44D				
	NPARTL	I	PARAMETER		42S								
	NSPEC	I	PARAMETER		272U	232P	218P	10S					
	NSUM	I	PARAMETER		12P	10S							
1575	NV	I	VARIABLE		424U/2	423U	422U	421U	420U/2	418I	314U/2	313U/2	312U/5
					310I	295U/2	294U/2	293U/5	291I	285U/3	284U/3	283U/2	281I
					236U/2	234I	232U/5	231U/8	229I	218U/5	217U/8	215I	169U/4
					168U/3	166U/2	165I						
	NVAR	I	PARAMETER		23P	20S							
	PAGE		EXTERNAL		137U								
	PERIML		EXTERNAL		158U								
	POINTS		EXTERNAL		352U								
0	PRE	R	2D EQ. ARRAY	LOCOM1	314P/2	313P/2	312S	295P/2	294P/2	293S	285U	284U	283S
					236U	232P	218P	207P	35D	33D			
47124	PSCR	R	1D EQ. ARRAY	LOCOM1	232P	218P	39D	37D					
	PWRIT		EXTERNAL		348U	279U	227U	213U	204U	195U	184U	175U	156U
					155U	154U							
0	RCAMB	R	1DIM ARRAY	HYDFIX	420P/2	166P/2	14D	11D					
2	RESTF	CH	VARIABLE	DUM.ARG.	59U	49D	1D						
75700	RHA	R	2DIM ARRAY	PLTVAR	232P	218P	178P	9D	8D				
112200	RHB	R	2DIM ARRAY	PLTVAR	232P	218P	189P	9D	8D				
61400	RHO	R	2DIM ARRAY	PLTVAR	420S	283U	236U	232P	231U	218P	217U	169U	168U
					9D	8D							
47014	RSCR	R	1D EQ. ARRAY	LOCOM1	232P	218P	168U	166S	39D	37D			
0	RVX	R	2DIM ARRAY	PLTVAR	421S	236U/2	231U/2	217U/2	98P	68P	9D	6D	
14300	RVY	R	2DIM ARRAY	PLTVAR	422S	236U/2	231U/2	217U/2	83P	69P	9D	6D	

```
30600 RVZ      R  2DIM ARRAY    PLIVAR   423S   236U/2  231U/2  217U/2  99P   84P   9D    6D
47014 SCR1     R  2D EQ. ARRAY  LOCOM1   307P   305P    293U/2  289P    260P  258P  256P  245P  243P
                                         241P   40D/2   39D/3   36D     34D
64020 SCR2     R  2D EQ. ARRAY  LOCOM1   307P   305P    293U/2  290P    40D   36D   34D
47344 SCRA     R  1D EQ. ARRAY  LOCOM1   232P   218P    40D     38D
47454 SCRB     R  1D EQ. ARRAY  LOCOM1   232P   218P    40D     38D
64020 SCRC     R  1D EQ. ARRAY  LOCOM1   232P   218P    40D     38D
      SFT         EXTERNAL               323U   304U    157U
      SHDCHR      EXTERNAL               135U
    0 SPACE    R  2DIM ARRAY    HOLDER   390P/2 389S    388S    387S    386S  251P  245P  24D
14302 SPACE2   R  2DIM ARRAY    HOLDER   169P/2 169S    168S    164P    24D
      SQRT     R  EXTERNAL               317U   316U    298U    297U    236U
      SWISSM      EXTERNAL               134U
      THKFRM      EXTERNAL               146U
    3 TIME     R  VARIABLE      GLOVAR   59U    19D
 1577 TLENG    R  VARIABLE               337P   336S    277P    276S    225P  224S  211P  210S  202P
                                         201S   193P    192S    182P    181S  173P  172S
  776 V        R  2DIM ARRAY             407P   403P    402P    389U    388U  379U  379S  378U  378S
                                         370U   370S    369U    369S    366U  366S  365S  27D
    3 VCOR     R  1DIM ARRAY    POINTEE  365U   290P    289P    256P    241P  109U  108U  29P   26D
 1601 VELMAX   R  VARIABLE               326P   324P    318U    316P    316S  313P  313S  307P  305P
                                         299U   297P    297S    294P    294S  286S
 1602 VELMIN   R  VARIABLE               326P   324P    318U/2  318S    317P  317S  314P  314S  307P
                                         305P   299U/2  299S    298P    298S  295P  295S  287S
      VELVCT      EXTERNAL               326U   324U    307U    305U
 1566 VHI      R  VARIABLE               399P   157P    128P
 1565 VLO      R  VARIABLE               398P   157P    128P
    3 VMAX     R  VARIABLE      BOUND    336U   323P    321U    304P    302U  276U  224U  210U  201U
                                         192U   181U    172U    157P    150P  148U  139U  128P  119U
                                         109S   57D
    2 VMIN     R  VARIABLE      BOUND    336U   323P    321U    304P    302U  276U  224U  210U  201U
                                         192U   181U    172U    157P    150P  148U  139U  128P  113U
                                         108S   57D
      WSIZE       EXTERNAL               128U
 1603 XA       R  VARIABLE               323P   322U    321S    304P    303U  302S
 1604 XB       R  VARIABLE               323P   322S    304P    303S
    0 XCOR     R  1DIM ARRAY    GRDVAR   96P    88U     86U     66P     18D   15D
      XMESS    R  EXTERNAL               336U   276U    224U    210U    201U  192U  181U  172U
      XNAME       EXTERNAL               144U
    0 XP       R  2DIM ARRAY    PARTL    353U   352P    344P    341U    341S  45D   43D
 1572 XSTEP    R  VARIABLE               150P   149S
  111 YCOR     R  1DIM ARRAY    GRDVAR   103U   101U    81P     67P     18D   15D
      YNAME       EXTERNAL               145U
 1571 YSTEP    R  VARIABLE               150P   149U    148S
  152 ZCOR     R  1DIM ARRAY    GRDVAR   97P    82P     73U     71U     18D   15D
  214 ZP       R  2DIM ARRAY    PARTL    353U   352P    344P    342U    342S  45D   43D
TABLE OF EXTERNAL NAMES
-----------------------

      $EFA        EXTERNAL               59U    58U
      $EFF        EXTERNAL               334U   274U    222U    208U    199U  190U  179U  170U  103U
                                         101U   88U     86U     73U     71U   59U   58U
      $EFI        EXTERNAL               334U   274U    222U    208U    199U  190U  179U  170U  103U
                                         101U   88U     86U     73U     71U   59U   59U
      $EFV        EXTERNAL               190U   179U    103U/2  101U/2  88U/2 86U/2 73U/2 71U/2 59U/2
```

$RNI		EXTERNAL	412U								
$WII		EXTERNAL	353U	349U							
$WIL		EXTERNAL	353U	349U							
$WIV		EXTERNAL	353U/3								
$WNL		EXTERNAL	413U								
AGSETF		EXTERNAL	406U	405U	404U	403U	402U	401U	400U	399U	398U
			397U	396U							
AGSETI		EXTERNAL	395U								
ANOTAT		EXTERNAL	394U								
AREA2D		EXTERNAL	139U								
COLCON		EXTERNAL	264U	258U	249U	243U					
CONINF		EXTERNAL	290U	289U	256U	241U					
CONREC		EXTERNAL	266U	260U							
CURVE		EXTERNAL	390U	344U							
DISCON		EXTERNAL	251U	245U							
ENDPL		EXTERNAL	392U	346U	330U						
EOS		EXTERNAL	232U	218U							
EZMXY		EXTERNAL	407U								
FRAME		EXTERNAL	147U								
FRAMEN		EXTERNAL	355U	329U	269U						
GDONE		EXTERNAL	433U								
GPLOT		EXTERNAL	425U								
GRAF		EXTERNAL	150U								
HEADIN		EXTERNAL	143U	142U	141U						
HEIGHT		EXTERNAL	140U								
ISMAX	I	EXTERNAL	313U	294U							
ISMIN	I	EXTERNAL	314U	295U							
LIBDISP		EXTERNAL	427U								
LIBNCAR		EXTERNAL	429U	320U	301U						
MARKER		EXTERNAL	339U								
MESSAG		EXTERNAL	337U	277U	225U	211U	202U	193U	182U	173U	
NOBRDR		EXTERNAL	138U								
PAGE		EXTERNAL	137U								
PERIM		EXTERNAL	158U								
POINTS		EXTERNAL	352U								
PWRIT		EXTERNAL	348U	279U	227U	213U	204U	195U	184U	175U	156U
			155U	154U							
SET		EXTERNAL	323U	304U	157U						
SIDCHR		EXTERNAL	135U								
SQRT	R	EXTERNAL	317U	316U	298U	297U	236U				
SWISSM		EXTERNAL	134U								
THKERM		EXTERNAL	146U								
VELVCT		EXTERNAL	326U	324U	307U	305U					
WSIZE		EXTERNAL	128U								
XMESS	R	EXTERNAL	336U	276U	224U	210U	201U	192U	181U	172U	
XNAME		EXTERNAL	144U								
YNAME		EXTERNAL	145U								

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

NBURST	= 2	NDM1	= 72
NDM12	= 6660	NDM1P2	= 160
NDM2	= 88	NNNX	= 72
NNNX2	= 74	NNNXP	= 73
NNNY	= 33	NNNY2	= 35
NNNYP	= 34	NNNZ	= 88
NNNZ2	= 90	NNNZP	= 89
NNYZ2D	= 3150	NOP	= 70
NPARTL	= 422	NSPEC	= 1
NSUM	= 5	NVAR	= 5

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
230	IP	130	409	2175d	2341
72	NV	165	169	2374a	36
72	NH	167	169	2415b	12
130	NV	215	219	2677b	73
120	NH	216	217	2707d	22
146	NV	229	233	3035c	73
145	NH	230	231	3046a	21
150	NV	234	236	3134a	40
150	NH	235	236	3144c	25
180	NV	281	285	3423a	32
180	NH	282	285	3436c	14
200	NV	291	296	3526b	44
190	NH	292	293	3535c	11
220	NV	310	315	3661b	56
210	NH	311	312	3675c	10
405	K	338	345	4076b	40
406	IJK	340	343	4113b	11
410	K	350	354	4160a	37
410	I	351	354	4165b	26
421	I	359	362	4232a	7
422	J	363	367	4253d	11
423	I	368	371	4302d	10
424	J	372	375	4332d	6
425	J	377	382	4355c	14

426	J	385	391	4411c	16
240	NV	418	424	4575b	34
240	NH	419	424	4611b	15

BLOCK NAMES AND LENGTHS IN OCTAL

4655	CONV3D	65	#1B	1243	#CL
126500	PLTVAR	1741	HYDFIX	1116	GRDVAR
14	GLOVAR	30602	HOLDER	116030	LOCOM1
646	PART1	1	NBUR	35	GRFCOM
4	ROUND				

STATIC SPACE (IN OCTAL)

B SAVE:	47
I SAVE:	10
CONSTANTS:	274
VARIABLES:	1316
TEMPORARIES:	1251
CODE:	3043
TOTAL:	6205

```
903
904      1            subroutine dispf (arrx,arry,length)
905      2            real arrx(length), arry(length)
906              c
907              c  ***************************************************************
908              c  *    This Subroutine plots profile graphs from the input data.  *
909              c  *     It is called to do profiles on part of an array, using    *
910              c  *     DISSPLA calls to the routine curve.                       *
911              c  ***************************************************************
912              c
913      3            xmax=arrx(1)
914      4.           xmin=arrx(1)
915      5.           ymax=arry(1)
916      6.           ymin=arry(1)
917              c     calculate the maximum and minimum x and y values
918      7.           do 10 icount=1,length
919      8.           if (arrx(icount) .lt. xmin) xmin=arrx(icount)
920      9.           if (arry(icount) .lt. ymin) ymin=arry(icount)
921     10.           if (arrx(icount) .gt. xmax) xmax=arrx(icount)
922     11.           if (arry(icount) .gt. ymax) ymax=arry(icount)
923     12.    10     continue
924     13.           if (ymax .gt. 0.0) then
925     14.             ymax=ymax*1.02
926     15.           else
927     16.             ymax=ymax*0.98
928     17.           endif
929     18.           if (ymin .gt. 0.0) then
930     19.             ymin=ymin*0.98
931     20.           else
932     21.             ymin=ymin*1.02
933     22.           endif
934     23.           if ((ymax .eq. ymin) .and. (ymin .eq. 0.)) then
935     24.             ymax=1.0
936     25              ymin=-1.0
937     26.           endif
938     27.           call thkfrm (.02)
939     28.           call frame
940              c     set up the axes with an appropriate range.
941              c     then draw the curve.
942     29.           call graf (xmin,'scale',xmax,ymin,'scale',ymax)
943     30.           call curve (arrx,arry,length,0)
944     31.           call endpl (0)
945     32.           return
946     33.           end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
---------------------------

 NUMBER USE    SOURCE PROGRAM REFERENCES

   10 UNDEF**     12I        7I
   10A36d          7I
   10B55a          7I
 00002 UNDEF**     8I
 00003 UNDEF**     9I
 00004 UNDEF**    10L
 00005 UNDEF**    11L
 00006 60a        13L
 00007 61d        15L
 00008 64d        18L
 00009 66c        20I
 00010 74a        23L
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
-------------------------------

 ADDRESS NAME    TYPE MAIN USAGE  BLOCK       SOURCE PROGRAM REFERENCES

      1 ARRX      R   1DIM ARRAY  DUM.ARG.     30P   10U/2  8U/2   4U   3U   2D   1D
      2 ARRY      R   1DIM ARRAY  DUM.ARG.     30P   11U/2  9U/2   6U   5U   2P   1D
        CURVE         EXTERNAL                 30U
     16 DISPRF        ENTRY                     1D/2
        ENDPL         EXTERNAL                 31U
        FRAME         EXTERNAL                 28U
        GRAF          EXTERNAL                 29U
     14 ICOUNT    I   VARIABLE                 11U/2 10U/2  9U/2   8U/2 7L
      3 LENGTH    I   VARIABLE    DUM.ARG.     30P    7N    2P/2   1D
        MAIN.         ENTRY
        THKFRM        EXTERNAL                 27U
     10 XMAX      R   VARIABLE                 29P   10S    10U    3S
     11 XMIN      R   VARIABLE                 29P    8S    8U     4S
     12 YMAX      R   VARIABLE                 29P   24S    23U    16U  16S  14U  14S  13U  11S
                                               11U    5S
     13 YMIN      R   VARIABLE                 29P   25S    23U/2  21U  21S  19U  19S  18U  9S
                                                9U    6S
TABLE OF EXTERNAL NAMES
-----------------------

        CURVE         EXTERNAL                 30U
        ENDPL         EXTERNAL                 31U
        FRAME         EXTERNAL                 28U
        GRAF          EXTERNAL                 29U
        THKFRM        EXTERNAL                 27U
```

```
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

      A  USED IN FORTRAN ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN DECLARATIVE STATEMENT     R  FORMAT USED IN A READ STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP    S  STORED SO CONTENTS MAY BE CHANGED
      I  INDEX OF A DO OR IMPLIED DO LOOP     U  NAME USED IN EXECUTABLE STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER    W  FORMAT USED IN A WRITE STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER    *  DEFINED OR DECLARED BUT NOT USED
      N  NAME USED AS A DO LOOP PARAMETER     ?  TEN OR MORE REFERENCES TO SYMBOL

 TABLE OF LOOPS ENCOUNTERED

  LABEL INDEX     FROM - TO   ADDRESS - LENGTH

     10 ICOUNT       7    12       36d        17
  BLOCK NAMES AND LENGTHS IN OCTAL

       120-DISPRF           23-#IB              21-#CL
 STATIC SPACE (IN OCTAL)
 -------------------------
 B SAVE:               16
 T SAVE:                5
 CONSTANTS:             6
 VARIABLES:             7
 TEMPORARIES:          21
 CODE:                103
                   -------
 TOTAL:               164
```

```
      subroutine distpl (arrx,arry,length,space)
      real arrx(length), arry(length)
      real space(length)
c
c ****************************************************************
c *  This Subroutine plots profile graphs from the input data.    *
c *   It is used for plots versus distance, changing from cm to km. *
c ****************************************************************
      do 5 icount=1,length
    5 space(icount)=arrx(icount)/100000.
c
      xmax=space(1)
      xmin=space(1)
      ymax=arry(1)
      ymin=arry(1)
c     calculate maximum and minimum x and y values
      do 10 icount=1,length
      if (space(icount) .lt. xmin) xmin=space(icount)
      if (arry(icount) .lt. ymin) ymin=arry(icount)
      if (space(icount) .gt. xmax) xmax=space(icount)
      if (arry(icount) .gt. ymax) ymax=arry(icount)
   10 continue
      if (ymax .gt. 0.0) then
        ymax=ymax*1.02
      elseif (ymax .lt. 0.0) then
        ymax=ymax*0.98
      endif
      if (ymin .gt. 0.0) then
        ymin=ymin*0.98
      elseif (ymin .lt. 0.0) then
        ymin=ymin*1.02
      endif
      if ((ymax .eq. ymin) .and. (ymin .eq. 0.)) then
        ymax=1.0
        ymin=-1.0
      endif
      call thkfrm (.02)
      call frame
c     set up axes with an appropriate range of values.
c     then draw the curve.
      call graf (xmin,'scale',xmax,ymin,'scale',ymax)
      call curve (space,arry,length,0)
      call endpl (0)
      return
      end
DISTPL          VECTOR LOOP BEGINS AT SEQ. NO.          4, P=       23b
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
---- ------          ----

  NUMBER USE     SOURCE PROGRAM REFERENCES

     5 UNDEF**        5L      4F
     5A35b            4I
     5B43d            4L
    10 UNDEF**       15L     10F
    10A57d           10I
    10B76a           10I
 00002 UNDEF**       11L
 00003 UNDEF**       12I
 00004 UNDEF**       13I
 00005 UNDEF**       14L
 00006 101a          16L
 00007 103d          18I
 00008 103d          18L
 00009 106d          21L
 00010 111c          23L
 00011 111c          23L
 00012 117a          26I
  (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
-------------------------------

 ADDRESS NAME     TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

       1 ARRX      R   1DIM ARRAY   DUM.ARG.       5U      2D      1D
       2 ARRY      R   1DIM ARRAY   DUM.ARG.      33P    14U/2   12U/2     9U      8U      2P      1D
         CURVE         EXTERNAL                   33U
      17 DISTPL        ENTRY                       1D/2
         ENDPL         EXTERNAL                   34U
         FRAME         EXTERNAL                   31U
         GRAF          EXTERNAL                   32U
      11 ICOUNT    I   VARIABLE                  14U/2   13U/2   12U/2   11U/2    10I     5U/2    4I
       3 LENGTH    I   VARIABLE     DUM.ARG.      33P     10N      4N      3P     2P/2     1D
         MAIN.         ENTRY
       4 SPACE     R   1DIM ARRAY   DUM.ARG.      33P    13U/2   11U/2     7U      6U      5S      3D      1D
         THKFRM        EXTERNAL                   30U
      12 XMAX      R   VARIABLE                   32P     13S     13U      6S
      13 XMIN      R   VARIABLE                   32P     11S     11U      7S
      14 YMAX      R   VARIABLE                   32P     27S     26U     19U     19S     18U     17U     17S     16U
                                                  14S     14U      8S
      15 YMIN      R   VARIABLE                   32P     28S    26U/2    24U     24S     23U     22U     22S     21U
                                                  12S     12U      9S
TABLE OF EXTERNAL NAMES
-------------- -------

         CURVE         EXTERNAL                   33U
         ENDPL         EXTERNAL                   34U
```

```
   FRAME      EXTERNAL                     31U
   GRAF       EXTERNAL                     32U
   THKFRM     EXTERNAL                     30U
ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
-- ---- --- -- ----
A  USED IN FORTRAN ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
D  DEFINED IN DECLARATIVE STATEMENT      R  FORMAT USED IN A READ STATEMENT
E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED SO CONTENTS MAY BE CHANGED
I  INDEX OF A DO OR IMPLIED DO LOOP      U  NAME USED IN EXECUTABLE STATEMENT
J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT USED IN A WRITE STATEMENT
L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED OR DECLARED BUT NOT USED
N  NAME USED AS A DO LOOP PARAMETER      ?  TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED
-------------- ----------
 LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

    5 ICOUNT        4     5        35b        6
   10 ICOUNT       10    15        57d       17
 BLOCK NAMES AND LENGTHS IN OCTAL

    143-DISTPL            30-#IB            21 #CL
STATIC SPACE (IN OCTAL)
-----------------------
B SAVE:                  23
I SAVE:                   5
CONSTANTS:                7
VARIABLES:                7
TEMPORARIES:             21
CODE:                   125
                   --------
TOTAL:                  214
```

```
        1.          subroutine discon (conarr,mdim,cnhold,kh,kv)
        2.          real conarr (mdim,kv), cnhold (kh,kv)
        3.          common work (5000)
              c
              c ****************************************************************
              c *      This Subroutine Draws contour plots using DISSPLA calls,   *
              c *       plotting the specified part of the array                 *
              c ****************************************************************
              c
        4.          do 10 icount=1,kh
        5.          do 10 jcount=1,kv
        6.          cnhold(icount,jcount)=conarr(icount,jcount)
        7.   10     continue
              c     set up the contour levels
        8.          call bcomon (5000)
        9.          call conmak (cnhold,kh,kv,'scale')
              c     specify the appearance of the lines, and their labels
       10.          call height (.10)
       11.          call conlin (0,'solid','labels',1,3)
       12.          call conang (60.)
       13.          call conthn (0.06)
              c     draw the contour lines
       14.          call contur (1,'labels','draw')
       15.          call endpl(0)
       16.          return
       17.          end
DISCON          VECTOR LOOP BEGINS AT SEQ. NO.          5, P=      35d
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE PROGRAM REFERENCES		
10	UNDEF**	7I	5I	4E
10A35d		4I		
10B62c		4I		
10C51b		5I		
10D57d		5L		

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES			
	BCOMON		EXTERNAL		8U			
3	CNHOLD	R	2D1M ARRAY	DUM.ARG.	9P	6S	2P	1D
	CONANG		EXTERNAL		12U			
1	CONARR	R	2D1M ARRAY	DUM.ARG.	6U	2D	1D	
	CONLIN		EXTERNAL		11U			
	CONMAK		EXTERNAL		9U			
	CONTIN		EXTERNAL		13U			
	CONTUR		EXTERNAL		14U			
24	DISCON		ENTRY		1D/2			
	ENDPL		EXTERNAL		15U			
	HEIGHT		EXTERNAL		10U			
21	ICOUNT	I	VARIABLE		6U/2	4I		
22	JCOUNT	I	VARIABLE		6U/2	5I		
4	KH	I	VARIABLE	DUM.ARG.	9P	4N	2P	1D
5	KV	I	VARIABLE	DUM.ARG.	9P	5N	2P/2	1D
	MAIN.		ENTRY					

TABLE OF EXTERNAL NAMES

BCOMON	EXTERNAL	8U
CONANG	EXTERNAL	12U
CONLIN	EXTERNAL	11U
CONMAK	EXTERNAL	9U
CONTIN	EXTERNAL	13U
CONTUR	EXTERNAL	14U
ENDPL	EXTERNAL	15U
HEIGHT	EXTERNAL	10U

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM -	TO	ADDRESS -	LENGTH
10	ICOUNT	4	7	356	26
10	JCOUNT	5	7	516	6

BLOCK NAMES AND LENGTHS IN OCTAL

124-DISCON	31-#IB	31-#CL	11610-//

STATIC SPACE (IN OCTAL)

B SAVE:	27
I SAVE:	2
CONSTANTS:	17
VARIABLES:	4
TEMPORARIES:	31
CODE:	101
TOTAL:	206

```
   1           subroutine prof3d (istep,restf,label,nlabel,iplane)
         c----------------------------------------------------------------
         c     draws profiles for fast3d.
         c     Prof3d is the routine which draws the profile plots.  At present it
         c can produce seven distinct plots, those being (in order) density,
         c energy, dynamic pressure, pressure (absolute), renormalized density,
         c renormalized pressure, and z velocity.  These plots are specified by
         c non-zero values in the array mplot(1) to mplot (7).
         c     These profiles are drawn in a plane specified by the absolute value
         c of mplot.  If the value is negative, it will plot in the "vertical"
         c direction, if positive in the "horizontal" direction.  That is, if the
         c value of mplot is -36 it will plot quantity vs. distance in the z direction
         c at x = 36.
         c     These plots are fairly consistent in the way they are implemented.
         c Interpolation is handled as it is in conv3d, through the flag lintp and
         c the routine conint.  In DISSPLA, the curves are actually plotted in the
         c routine distpl, rather than in prof3d itself.
         c------------------------ array dimensions -----------------------
   2.          parameter (nnnx=72, nnny=33, nnnz=88)
   3.          parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
   4.          parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
         c------------------------ plot variables ------------------------
   5.          parameter (ndm1=72, ndm2=88)
   6.          real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
   7.          real erg(ndm1,ndm2)
   8.          real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
   9.          common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
         c------------------- grid arrays ---------------------------------
  10.          real xcor(nnnxp), ycor(nnny), zcor(nnnz)
  11.          real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
  12.          real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
         c
  13.          common /holder/ space(ndm1,ndm2),valmin,valrng,
          +       space2(ndm1,ndm2)
  14.          common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
          1    , dtody2, dtodz2
  15.          real rcamb(nnnz2),pcamb(nnnz2)
  16.          common /hydfix/ rcamb,pcamb
         c------------------------ global variables -----------------------
  17.          common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
          1    , nstep, lllx, llly, lllz
         c---------------------------- scratch space ----------------------
  18.          parameter (nnyz2d=nnny2+nnnz2, nspec=1, nvar=4+nspec)
  19.          parameter (ndm12=(ndm1+2)+(ndm2+2))
  20.          real dumv(ndm12,nvar), dumy(ndm12)
  21.          common /locom1/ dumv, dumy
         c------------------------------ pointers -------------------------
  22.          real arr(ndm1,ndm2), hcor(ndm1), vcor(ndm2)
  23.          pointer (iparr,arr), (iphcor,hcor), (ipvcor,vcor)
  24.          pointer (ipminh,minh), (ipmaxh,maxh)
  25.          pointer (ipminv,minv), (ipmaxv,maxv)
         c------------------------ local declarations ---------------------
  26.          parameter (nhv=88)
  27.          real pre(ndm1,ndm2), mac(ndm1,ndm2), gam(ndm1,ndm2)
  28.          real dpi(ndm1,ndm2), eint(nhv), scrh(nhv)
```

```
  29.          equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
  30.          equivalence (dpr,dumv(1,4)), (eint,dumv(1,5)), (scrh,dumy)
  31.          real rscr(ndm1), pscr(ndm1), qscr(ndm1)
  32.          real scra(ndm1), scrb(ndm1), scrc(ndm1)
  33.          equivalence (rscr,mac), (pscr,mac(1,2)), (qscr,mac(1,3))
  34.          equivalence (scra,mac(1,4)), (scrb,mac(1,5)), (scrc,mac(1,6))
       c
  35.          integer mplot(14)
  36.          character *40 label
  37.          character *8 restf
  38.          real glab1(7), glab2(7), glab3(7)
  39.          logical displa, darkhd
  40.          common /qifcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
            1    mplot, lintp, colour, dmin, dmax, displa, i1, j1, darkhd
       c
       c    first print labels.  use encode to set up the titles
  41.          encode (nlabel,310,glab1) label(1:nlabel)
  42.          encode (40,320,glab2) time,istep,restf
       c
       c    set window sizes.
  43.          go to (10,20,30), mslice
       c
       c    X-Y slice.
  44.     10   continue
  45.          ipminh=loc(imin)
  46.          ipmaxh=loc(imax)
  47.          ipminv=loc(jmin)
  48.          ipmaxv=loc(jmax)
  49.          iphcor=loc(xcor)
  50.          ipvcor=loc(ycor)
  51.          go to 40
       c
       c    Y-Z slice.
  52.     20   continue
  53.          ipminh=loc(jmin)
  54.          ipmaxh=loc(jmax)
  55.          ipminv=loc(kmin)
  56.          ipmaxv=loc(kmax)
  57.          iphcor=loc(ycor)
  58.          ipvcor=loc(zcor)
  59.          go to 40
       c
       c    X-Z slice.
  60.     30   continue
  61.          ipminh=loc(imin)
  62.          ipmaxh=loc(imax)
  63.          ipminv=loc(kmin)
  64.          ipmaxv=loc(kmax)
  65.          iphcor=loc(xcor)
  66.          ipvcor=loc(zcor)
  67.     40   continue
       c
       c    plot profiles.
  68.          ispec=1
  69.          do 240 ip=1,7
  70.          if (mplot(ip).eq.0) go to 240
```

```
1182      c     make DISSPLA calls to set the page size, print type, plot area,
1183      c     and draw the titles, or NCAR calls to write the titles.
1184   71    50     if (displa) then
1185   72.             call page (10.,11.)
1186   73.             call nobrde
1187   74              call area2d (6.,8.)
1188   75.             if (darkbd) then
1189   76                call shdche (90.,1, 003,1)
1190   77.               call swissm
1191   78.             endif
1192   79.             call height (.16)
1193   80.             call headin (glab1,nlabel-1,1.5,3)
1194   81.             call headin (glab2,25,1.5,3)
1195   82            else
1196   83.             call pwrit (550,50,glab1,nlabel,2,0,0)
1197   84.             call pwrit (550,25,glab2,40,1,0,0)
1198   85.           endif
1199      c
1200   86.           go to (60,100,110,130,1000,1010,1020), ip
1201      c
1202      c     density profile
1203      c     plots up to 3 species of absolute density, usually total density (1),
1204      c     dust density (2), and explosive density (3).
1205   87.   60     go to (70,80,90), ispec
1206      c     label the axes
1207   88    70     if (displa) then
1208   89.             call xname ('distance (km)',13)
1209   90.             call yname ('density (g/cm**3)',17)
1210   91.           else
1211   92.             call anotat (14hdistance, km.$,15hdensity, gm/cc$,1,1,0,0)
1212   93.           endif
1213      c     set pointer to array
1214   94.           ipar=loc(rho)
1215   95.           go to 160
1216      c     label the axes
1217   96.   80     if (displa) then
1218   97              call xname ('distance (km)',13)
1219   98.             call yname ('density 2 (g/cm**3)',19)
1220   99.           else
1221  100.             call anotat (14hdistance, km.$,17hdensity 2, gm/cc$,1,1,0,0)
1222  101.           endif
1223      c     set pointer to array
1224  102.           ipar=loc(rho)
1225  103.           go to 160
1226      c     label the axes
1227  104.   90     if (displa) then
1228  105.             call xname ('distance (km)',13)
1229  106.             call yname ('density 3 (g/cm**3)',19)
1230  107.           else
1231  108.             call anotat (14hdistance, km.$,17hdensity 3, gm/cc$,1,1,0,0)
1232  109            endif
1233      c     set pointer to array
1234  110.           ipar=loc(rhb)
1235  111            go to 160
1236      c
1237      c     energy profile
```

```
           c      label the axes
  112.    100       if (displa) then
  113                 call xname ('distance (km)',13)
  114.                call yname ('energy (ergs/cm**3)',19)
  115.              else
  116.                call anotat (14hdistance, km.$,16henergy, ergs/cc$,1,1,0,0)
  117.              endif
           c      set pointer to array
  118.              iparr=loc(erg)
  119.              go to 160
           c
           c      dynamic pressure profile.
           c      label the axes
  120.    110       if (displa) then
  121.                call xname ('distance (km)',13)
  122.                call yname ('dynamic pressure (dynes/cm**2)',30)
  123.              else
  124.                call anotat(14hdistance, km.$,
              +          30hdynamic pressure, dynes/cm**2$,1,1,0,0)
  125.              endif
           c      set pointer to array
  126.              iparr=loc(dpr)
           c      calculate the dynamic pressure from momenta and density
  127.              do 120 nv=minv,maxv
  128.              do 120 nh=minh,maxh
  129.    120       dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
              1     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
  130.              go to 160
           c
           c      pressure profile.
           c      this is a profile of absolute pressure.
           c
           c      label the axes
  131.    130       if (displa) then
  132.                call xname ('distance (km)',13)
  133.                call yname ('pressure (dynes/cm**2)',22)
  134.              else
  135                 call anotat (14hdistance, km.$,22hpressure, dynes/cm**2$,
              +            1,1,0,0)
  136.              endif
           c      set pointer to array
  137.              iparr=loc(pre)
           c      calculate the dynamic pressure, for use in calculating pressure
  138.              do 135 nv=minv,maxv
  139.              do 135 nh=minh,maxh
  140.    135       dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
              1     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
           c      calculate the pressure, using equations of state subroutine eos
  141.              do 150 nv=minv,maxv
  142.              do 140 nh=minh,maxh
  143.    140       eint(nh)=erg(nh,nv)-dpr(nh,nv)
  144               call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
              1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
              2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
  145.    150       continue
  146.              go to 160
```

```
1244          c
1245          c  Renormalized density -- with respect to ambient density
1246          c      label the axes
1247     147    1000      if (displa) then
1248     148.                call yname ('distance (km)',13)
1249     149.                call yname ('renormalized density',22)
1250     150.              else
1251     151.                call anotat (14hdistance, cm.$,
1252                  +       21hrenormalized density$,1,1,0,0)
1253     152.              endif
1254          c      set pointer to array
1255     153.              ipar=loc(space2)
1256          c      renormalize the density with the ambient
1257     154.              do 1002 jjv=minv,maxv
1258     155.              rscr(jjv)=cvmgt(rcamb(iplane),rcamb(jjv+1),mslice .eq. 1)
1259     156.              do 1002 iih=minh,maxh
1260     157.              space2(iih,jjv)=rho(iih,jjv)/rscr(jjv)
1261     158.    1002      continue
1262     159               go to 160
1263          c
1264          c  Renormalized pressure
1265          c      label the axes
1266     160.    1010      if (displa) then
1267     161.                call yname ('distance (km)',13)
1268     162.                call yname ('renormalized pressure',21)
1269     163.              else
1270     164.                call anotat (14hdistance, cm.$,
1271                  +       22hrenormalized pressure$,1,1,0,0)
1272     165.              endif
1273          c      set pointer to the array
1274     166.              ipar=loc(space2)
1275          c      calculate dynamic pressure to use in calculating pressure
1276     167.              do 1011 nv=minv,maxv
1277     168.              do 1011 nh=minh,maxh
1278     169.    1011      dpr(nh,nv)=0.5*(rvx(nh,nv)*rvx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
1279                  1     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
1280          c      calculate pressure (absolute) from equations of state routine eos
1281     170.              do 1013 nv minv,maxv
1282     171.              do 1012 nh minh,maxh
1283     172.    1012      eint(nh)=erg(nh,nv)-dpr(nh,nv)
1284     173.              call eos (maxh-minh+1,nspec,rho(minh,nv),eint(minh),gam(minh
1285                  1     ,nv),pre(minh,nv),rscr(minh),pscr(minh),gscr(minh),scra(minh)
1286                  2     ,scrb(minh),scrc(minh),rha(minh,nv),rhb(minh,nv))
1287     174.    1013      continue
1288          c      renormalize density with the ambient
1289     175.              do 1015 jjv minv,maxv
1290     176.              pscr(jjv)=cvmgt(pcamb(iplane),pcamb(jjv+1),mslice .eq. 1)
1291     177.              do 1015 iih=minh,maxh
1292     178.              space2(iih,jjv)=pre(iih,jjv)/pscr(jjv)
1293     179.    1015      continue
1294     180.              go to 160
1295          c
1296          c  Z velocity profile
1297          c      label the axes
1298     181    1020      if (displa) then
1299     182.                call yname ('distance (km)',13)
```

```
1300    183.              call vname ('z velocity (cm/sec)',19)
1301    184.            else
1302    185.              call anotat (14hdistance, cm.$,
1303          +             19hz velocity, cm/sec$,1,1,0,0)
1304    186.            endif
1305          c      set pointer to array
1306    187.            iparr=loc(space2)
1307          c      calculate z velocity from z momentum and density
1308    188.            do 1022 jjv=minv,maxv
1309    189.            do 1022 iih=minh,maxh
1310    190.            space2(iih,jjv)=rvz(iih,jjv)/rho(iih,jjv)
1311    191.     1022   continue
1312    192.            go to 160
1313          c
1314    193.     160    mpabs=iabs(mplot(ip))
1315    194.            go to (170,190,210), mslice
1316          c
1317          c      Draw the profile on the appropriate plane, drawing from the value
1318          c      of mslice (xy,yz,or xz plane) and mplot (direction and location
1319          c      of profile).  First draw the last title (specifying which plane,
1320          c      direction, and location), then plot the profile itself.  In DISSPLA,
1321          c      the actual drawing is done in the routine distpl, which converts from
1322          c      centimeters to kilometers and plots the proper portion of the grid.
1323          c      X-Y slice.
1324    195.     170    if (displa) then
1325    196.              if (mplot(ip).gt.0) then
1326    197.                encode(31,250,glab3)mpabs,vcor(mpabs)/100000.,
1327          +                 iplane,zcor(iplane)/100000.
1328    198.                call headin (glab3,30,1.5,3)
1329    199.                call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
1330    200.              else
1331    201.                encode(31,260,glab3)mpabs,hcor(mpabs)/100000.,
1332          +                 iplane,zcor(iplane)/100000.
1333    202.                call headin (glab3,30,1.5,3)
1334    203                 do 180 nv=minv,maxv
1335    204.     180        scrh(nv)=arr(mpabs,nv)
1336    205.                call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
1337    206.              endif
1338    207.            else
1339    208.              if (mplot(ip).gt.0) then
1340    209.                encode(39,255,glab3)mpabs,vcor(mpabs),iplane,zcor(iplane)
1341    210.                call pwrit (550,1010,glab3,39,1,0,0)
1342    211.                call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,1h$)
1343    212.              else
1344    213.                encode(39,265,glab3)mpabs,hcor(mpabs),iplane,zcor(iplane)
1345    214.                call pwrit (550,1010,glab3,39,1,0,0)
1346    215.                do 185 nv=minv,maxv
1347    216.     185        scrh(nv)=arr(mpabs,nv)
1348    217.                call ezxy (vcor(minv),scrh(minv),maxv-minv+1,1h$)
1349    218.              endif
1350    219.            endif
1351    220.            go to 230
1352          c
1353          c      Y Z slice.
1354    221.     190    if (displa) then
1355    222.              if (mplot(ip).gt.0) then
```

```
                encode(31,270,glab3)mpabs,vcor(mpabs)/100000.,
     *            lplane,zcor(lplane)/100000.
                call headin (glab3,30,1.5,3)
                call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
              else
                encode(31,280,glab3)mpabs,hcor(mpabs)/100000.,
     *            lplane,zcor(lplane)/100000.
                call headin (glab3,30,1.5,3)
                do 200 nv=minv,maxv
  200           scrh(nv)=arr(mpabs,nv)
                call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
              endif
            else
              if (mplot(ip).gt.0) then
                encode(39,275,glab3)mpabs,vcor(mpabs),lplane,zcor(lplane)
                call pwrit (550,1010,glab3,39,1,0,0)
                call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,1h$)
              else
                encode(39,285,glab3)mpabs,hcor(mpabs),lplane,zcor(lplane)
                call pwrit (550,1010,glab3,39,1,0,0)
                do 205 nv=minv,maxv
  205           scrh(nv)=arr(mpabs,nv)
                call ezxy (vcor(minv),scrh(minv),maxv-minv+1,1h$)
              endif
            endif
            go to 230
c
c     X-Z slice.
  210       if (displa) then
              if (mplot(ip).gt.0) then
                encode(31,290,glab3)mpabs,vcor(mpabs)/100000.,
     *            lplane,ycor(lplane)/100000.
                call headin (glab3,30,1.5,3)
                call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
              else
                encode(31,300,glab3)mpabs,hcor(mpabs)/100000.,
     *            lplane,ycor(lplane)/100000.
                call headin (glab3,30,1.5,3)
                do 220 nv=minv,maxv
  220           scrh(nv)=arr(mpabs,nv)
                call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
              endif
            else
              if (mplot(ip).gt.0) then
                encode(39,295,glab3)mpabs,vcor(mpabs),lplane,ycor(lplane)
                call pwrit (550,1010,glab3,39,1,0,0)
                call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,1h$)
              else
                encode(39,305,glab3)mpabs,hcor(mpabs),lplane,ycor(lplane)
                call pwrit (550,1010,glab3,39,1,0,0)
                do 225 nv=minv,maxv
  225           scrh(nv)=arr(mpabs,nv)
                call ezxy (vcor(minv),scrh(minv),maxv-minv+1,1h$)
              endif
            endif
c
```

```
    1412               c     go back and plot the other species of density, if any
    1413     272.   230         ispec ispec+1
    1414     273.               if (ispec.le.nspec .and. ip .eq. 1) go to 50
    1415     274.   240         continue
    1416     275.               return
    1417               c
    1418               c     format statements for the titles of these plots
    1419     276.   250 format ('jy,y,kz,z=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1420     277.   260 format ('ix,x,kz,z=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1421     278.   270 format ('kz,z,ix,x=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1422     279.   280 format ('jy,y,ix,x=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1423     280.   290 format ('kz,z,jy,y=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1424     281.   300 format ('ix,x,jy,y=',i3,',',f5.1,',',1x,i3,',',f5.1,'$')
    1425     282.   310 format (a)
    1426     283.   255 format ('jy,y,kz,z=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1427     284.   265 format ('ix,x,kz,z=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1428     285.   275 format ('kz,z,ix,x=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1429     286.   285 format ('jy,y,ix,x=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1430     287.   295 format ('kz,z,jy,y=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1431     288.   305 format ('ix,x,jy,y=',i3,',',e9.3,',',1x,i3,',',e9.3,'$')
    1432     289.   320 format ('time=',f5.1,' sec, step',i5,', dump ',a)
    1433     290.       end
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        128, P=     1031d
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        139, P=     1113d
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        142, P=     1152c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        156, P=     1265b
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        168, P=     1345d
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        171, P=     1404c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        177, P=     1474a
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        189, P=     1554c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        203, P=     1612c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        215, P=     1742b
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        229, P=     2071c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        241, P=     2221b
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        255, P=     2350c
  PROF3D                VECTOR LOOP BEGINS AT SEQ. NO.        267, P=     2500b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE	PROGRAM REFERENCES							
10	530c	44L	43J							
20	543c	52I	43J							
30	556c	60I	43J							
40	571a	67I	59J	51J						
50	574c	273J	71L							
60	660a	87I	86J							
70	666a	88I	87J							
80	711b	96I	87J							
90	734c	104L	87J							
100	757d	112L	86J							
110	1003a	120I	86J							
120	UNDEF**	129I	128F	127F						
120A	1031d	127L								
120B	1064c	127I								
120C	1042b	128L								
120D	1061d	128I								
130	1065a	131I	86J							
135	UNDEF**	140I	139F	138E						
135A	1113d	138I								
135B	1146c	138L								
135C	1124b	139L								
135D	1143d	139I								
140	UNDEF**	143L	142E							
140A	1164b	142L								
140B	1175b	142L								
150	UNDEF**	145L	141E							
150A	1152c	141L								
150B	1236a	141I								
160	1601d	193I	192J	180J	159J	146J	130J	119J	111J	103J
		95J								
170	1612c	195I	194J							
180	UNDEF**	204L	203F							
180A	1721b	203L								
180B	1727b	203I								
185	UNDEF**	216L	215F							
185A	2050c	215L								
185B	2056c	215I								
190	2071c	221I	194J							
200	UNDEF**	230I	229F							
200A	2200b	229I								
200B	2206b	229L								
205	UNDEF**	242I	241F							
205A	2327c	241L								
205B	2335c	241I								
210	2350c	247L	194J							
220	UNDEF**	256I	255E							
220A	2457b	255I								
220B	2465b	255L								
225	UNDEF**	268I	267F							
225A	2606c	267L								

```
    225B2614c      267L
    230 2627a      272I      246J      220J
    240 2633b      274I       70J       69I
    240A573a        69L
    240BUNDEF**     69I
    250 FN         276L      197R
    255 FN         283L      209R
    260 FN         277I      201R
    265 FN         284L      213R
    270 FN         278L      223R
    275 FN         285L      235R
    280 FN         279I      227R
    285 FN         286I      239R
    290 FN         280L      249R
    295 FN         287I      261R
    300 FN         281L      253R
    305 FN         288L      265R
    310 FN         282I       41R
    320 FN         289L       42R
   1000 1236c      147L       86J
   1002 UNDEF**    158L      156I      154E
   1002A1265b      154L
   1002B1316c      154I
   1002C1304b      156L
   1002D1313d      156I
   1010 1317a      160L       86J
   1011 UNDEF**    169L      168F      167F
   1011A1345d      167I
   1011B1400c      167L
   1011C1356b      168L
   1011D1375d      168L
   1012 UNDEF**    172I      171F
   1012A1416b      171I
   1012B1427b      171L
   1013 UNDEF**    174L      170F
   1013A1404c      170I
   1013B1470a      170I
   1015 UNDEF**    179L      177I      175F
   1015A1474a      175L
   1015B1525b      175L
   1015C1513a      177L
   1015D1522c      177I
   1020 1525d      181L       86J
   1022 UNDEF**    191L      189F      188E
   1022A1554c      188I
   1022B1601b      188L
   1022C1565a      189L
   1022D1576c      189L
  00002 527a        43W
  00003 631b        71I
  00004 613a        75L
  00005 650a        82L
  00006 654c        86W
  00007 664c        87W
  00008 701a        88L
  00009 707a        91L
```

```
 00010 724b     96L
 00011 732b     99L
 00012 747c    104L
 00013 755c    107L
 00014 772d    112L
 00015 1000d   115L
 00016 1016a   120L
 00017 1024a   123L
 00018 1100a   131L
 00019 1106a   134L
 00020 1251c   147L
 00021 1257c   150L
 00022 1332a   160L
 00023 1340a   163L
 00024 1540d   181L
 00025 1546d   184L
 00026 1611a   194W
 00027 1742b   195L
 00028 1660c   196L
 00029 1741d   200L
 00030 2071a   207L
 00031 2007c   208L
 00032 2071a   212L
 00033 2221b   221L
 00034 2137c   222L
 00035 2220d   226L
 00036 2350a   233L
 00037 2266c   234L
 00038 2350a   238L
 00039 2500b   247L
 00040 2416c   248L
 00041 2477d   252L
 00042 2627a   259L
 00043 2545c   260L
 00044 2627a   264L
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
  (FN=FORMAT NUMBER, UNDEF*-UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
 ------------------------

ADDRESS NAME     TYPE MAIN USAGE   BLOCK      SOURCE PROGRAM REFERENCES

        $FFA          EXTERNAL                  42U     41U
        $FFF          EXTERNAL                 265U    261U    253U    249U    239U    235U    227U    223U    213U
                                               209U    201U    197U     42U     41U
        $FFI          EXTERNAL                 265U    261U    253U    249U    239U    235U    227U    223U    213U
                                               209U    201U    197U     42U     41U
        $FFV          EXTERNAL                 265U/4  261U/4  253U/4  249U/4  239U/4  235U/4  227U/4  223U/4  213U/4
                                               209U/4  201U/4  197U/4   42U/2
        ANDIAT        EXTERNAL                 185U    164U    151U    135U    124U    116U    108U    100U     92U
        AREA2D        EXTERNAL                  74U
      1 ARR       R   2DIM ARRAY   POINTEE     268U    263P    256U    251P    242U    237P    230U    225P    216U
                                               211P    204U    199P     23P     22D
        CVMGT         INLINE FUNCT.            176U    155U
```

```
  34 DARKHD    I   VARIABLE        GRFCOM      75U     40D     39D
  31 DISPLA    I   VARIABLE        GRFCOM     247U    221U    195U    181U    160U    147U    131U    120U    112U
                                              104U     96U     88U     71U     40D     39D
     DISTPL        EXTERNAL                   257U    251U    231U    225U    205U    199U
47014 DPR      R   2D EQ. ARRAY LOCOM1        172U    169S    143U    140S    129S    126P     30D     28D
64020 FINT     R   1D EQ. ARRAY LOCOM1        173P    172S    144P    143S     30D     28D
     FOS           EXTERNAL                   173U    144U
45100 ERG      R   2DIM ARRAY   PLTVAR        172U    143U    118P      9D      7D
     EZXY          EXTERNAL                   269U    263U    243U    237U    217U    211U
32010 GAM      R   2D EQ. ARRAY LOCOM1        173P    144P     29D     27D
  413 GLAB1    R   1DIM ARRAY                  83P     80P     41S     38D
  422 GLAB2    R   1DIM ARRAY                  84P     81P     42S     38D
  431 GLAB3    R   1DIM ARRAY                 266P    265S    262P    261S    254P    253S    250P    249S    240P
                                              239S    236P    235S    228P    227S    224P    223S    214P    213S
                                              210P    209S    202P    201S    198P    197S     38D
15224 GSCR     R   1D EQ. ARRAY LOCOM1        173P    144P     33D     31D
    2 HCOR     R   1DIM ARRAY   POINTEE       265U    263P    253U    251P    239U    237P    227U    225P    213U
                                              211P    201U    199P     23P     22D
     HEADIN        EXTERNAL                   254U    250U    228U    224U    202U    198U     81U     80U
     HEIGHT        EXTERNAL                    79U
     IABS      I   INLINE FUNCT.              193U
  445 III      I   VARIABLE                   190U/3  189I    178U/2  177I    157U/2  156I
    2 IMAX     I   VARIABLE     GRFCOM         62P     46P     40D
    1 IMIN     I   VARIABLE     GRFCOM         61P     45P     40D
  441 IP       I   VARIABLE                   273U    260U    248U    234U    222U    208U    196U    193U     86P
                                               70U     69I
  404 IPARR   *I   VARIABLE                   187S    166S    153S    137S    126S    118S    110S    102S     94S
                                               23P
  405 IPHCOR  *I   VARIABLE                    65S     57S     49S     23P
  410 IPMAXH  *I   VARIABLE                    62S     54S     46S     24P
  412 IPMAXV  *I   VARIABLE                    64S     56S     48S     25P
  407 IPMINH  *I   VARIABLE                    61S     53S     45S     24P
  411 IPMINV  *I   VARIABLE                    63S     55S     47S     25P
  406 IPVCOR  *I   VARIABLE                    66S     58S     50S     23P
  440 ISPEC    I   VARIABLE                   273U    272U    272S     87P     68S
    1 ISTEP    I   VARIABLE     DUM.ARG.       42U      1D
  444 JJV      I   VARIABLE                   190U/3  188I    178U/3  176U/2  175I    157U/3  155U/2  154I
    4 JMAX     I   VARIABLE     GRFCOM         54P     48P     40D
    3 JMIN     I   VARIABLE     GRFCOM         53P     47P     40D
    6 KMAX     I   VARIABLE     GRFCOM         64P     56P     40D
    5 KMIN     I   VARIABLE     GRFCOM         63P     55P     40D
    3 LABEL    CH  VARIABLE     DUM.ARG.       41U     36D      1D
     LOC       I   INLINE FUNCT.              187U    166U    153U    137U    126U    118U    110U    102U     94U
                                               66U     65U     64U     63U     62U     61U     58U     57U     56U
                                               55U     54U     53U     50U     49U     48U     47U     46U     45U
    5 LPLANE   I   VARIABLE     DUM.ARG.      265U/2  261U/2  253U/2  249U/2  239U/2  235U/2  227U/2  223U/2  213U/2
                                              209U/2  201U/2  197U/2  176U    155U      1D
     MAIN.         ENTRY
    5 MAXH     I   VARIABLE     POINTEE       263P    251P    237P    225P    211P    199P    189N    177N    173P
                                              171N    168N    156N    144P    142N    139N    128N     24P
    7 MAXV     I   VARIABLE     POINTEE       269P    267N    257P    255N    243P    241N    231P    229N    217P
                                              215N    205P    203N    188N    175N    170N    167N    154N    141N
                                              138N    127N     25P
    4 MINH     I   VARIABLE     POINTEE       263U/3  251U/3  237U/3  225U/3  211U/3  199U/3  189N    177N    173U/?
                                              171N    168N    156N    144U/?  142N    139N    128N     24P
    6 MINV     I   VARIABLE     POINTEE       269U/3  267N    257U/3  255N    243U/3  241N    231U/3  229N    217U/3
```

					215N	205U/3	203N	188N	175N	170N	167N	154N	141N
					138N	127N	25P						
446	MPABS	I	VARIABLE		268U	265U/2	263U	261U/2	256U	253U/2	251U	249U/2	242U
					239U/2	237U	235U/2	230U	227U/2	225U	223U/2	216U	213U/2
					211U	209U/2	204U	201U/2	199U	197U/2	193S		
7	MPLOT	I	1DIM ARRAY	GRFCOM	260U	248U	234U	222U	208U	196U	193P	70U	40D
					35D								
0	MSLICE	I	VARIABLE	GRFCOM	194P	176P	155P	43P	40D				
	NDM1	I	PARAMETER		32P/3	31P/3	28P	27P/3	22P/2	13P/2	8P/3	7P	6P/3
					5S								
	NDM12	I	PARAMETER		20P/2	19S							
	NDM2	I	PARAMETER		28P	27P/3	22P/2	13P/2	8P/3	7P	6P/3	5S	
443	NH	I	VARIABLE		172U/3	171I	169U/8	168I	143U/3	142I	140U/8	139I	129U/8
					128I								
	NHV	I	PARAMETER		28P/2	26S							
4	NLABEL	I	VARIABLE	DUM.ARG.	83P	80P	41U	41P	1D				
	NNNX	I	PARAMETER		2S								
	NNNX2	I	PARAMETER		4S								
	NNNXP	I	PARAMETER		12P	11P	10P	3S					
	NNNY	I	PARAMETER		10P	2S							
	NNNY2	I	PARAMETER		12P	11P	4S						
	NNNYP	I	PARAMETER		3S								
	NNNZ	I	PARAMETER		10P	2S							
	NNNZ2	I	PARAMETER		15P/2	12P	11P	4S					
	NNNZP	I	PARAMETER		3S								
	NNYZ2	I	PARAMETER		18S								
	NOBRDR		EXTERNAL		73U								
	NSPEC	I	PARAMETER		273U	173P	144P	18S					
442	NV	I	VARIABLE		268U/2	267I	256U/2	255I	242U/2	241I	230U/2	229I	216U/2
					215I	204U/2	203I	173U/5	172U/2	170I	169U/8	167I	144U/5
					143U/2	141I	140U/8	138I	129U/8	127I			
	NVAR	I	PARAMETER		20P	18S							
	PAGE		EXTERNAL		72U								
132	PCAMB	R	1DIM ARRAY	HYDFIX	176P/2	16D	15D						
0	PRE	R	2D EQ. ARRAY	LOCOM1	178U	173P	144P	137P	29D	27D			
454	PROF3D		ENTRY		1D/2								
15114	PSCR	R	1D EQ. ARRAY	LOCOM1	178U	176S	173P	144P	33D	31D			
	PWRIT		EXTERNAL		266U	262U	240U	236U	214U	210U	84U	83U	
0	RCAMB	R	1DIM ARRAY	HYDFIX	155P/2	16D	15D						
2	RESTF	CH	VARIABLE	DUM.ARG.	42U	37D	1D						
75700	RHA	R	2DIM ARRAY	PLTVAR	173P	144P	102P	9D	8D				
112200	RHB	R	2DIM ARRAY	PLTVAR	173P	144P	110P	9D	8D				
61400	RHO	R	2DIM ARRAY	PLTVAR	190U	173P	169U	157U	144P	140U	129U	94P	9D
					8D								
15004	RSCR	R	1D EQ. ARRAY	LOCOM1	173P	157U	155S	144P	33D	31D			
0	RVX	R	2DIM ARRAY	PLTVAR	169U/2	140U/2	129U/2	9D	6D				
14300	RVY	R	2DIM ARRAY	PLTVAR	169U/2	140U/2	129U/2	9D	6D				
30600	RVZ	R	2DIM ARRAY	PLTVAR	190U	169U/2	140U/2	129U/2	9D	6D			
15334	SCRA	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
15444	SCRB	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
15554	SCRC	R	1D EQ. ARRAY	LOCOM1	173P	144P	34D	32D					
101024	SCRH	R	1D EQ. ARRAY	LOCOM1	269P	268S	257P	256S	243P	242S	231P	230S	217P
					216S	205P	204S	30D	28D				
	SHDCHR		EXTERNAL		76U								
0	SPACE	R	2DIM ARRAY	HOLDER	257P	251P	231P	225P	205P	199P	13D		
14302	SPACE2	R	2DIM ARRAY	HOLDER	190S	187P	178S	166P	157S	153P	13D		

```
        SWISSM          EXTERNAL                        77U
      3 TIME       R    VARIABLE        GLOVAR          42U     17D
      3 VCOR       R    1DIM ARRAY      POINTEE         269P    261U    257P    249U    243P    235U    231P    223U    217P
                                                        209U    205P    197U    23P     22D
      0 XCOR       R    1DIM ARRAY      GRDVAR          65P     49P     14D     10D
        XNAME           EXTERNAL                        182U    161U    148U    132U    121U    113U    105U    97U     89U
    111 YCOR       R    1DIM ARRAY      GRDVAR          265U    261U    253U    249U    57P     50P     14D     10D
        YNAME           EXTERNAL                        183U    162U    149U    133U    122U    114U    106U    98U     90U
    152 ZCOR       R    1DIM ARRAY      GRDVAR          239U    235U    227U    223U    213U    209U    201U    197U    66P
                                                        58P     14D     10D
TABLE OF EXTERNAL NAMES
-----------------------

        $EFA            EXTERNAL                        42U     41U
        $EFF            EXTERNAL                        265U    261U    253U    249U    239U    235U    227U    223U    213U
                                                        209U    201U    197U    42U     41U
        $EFI            EXTERNAL                        265U    261U    253U    249U    239U    235U    227U    223U    213U
                                                        209U    201U    197U    42U     41U
        $EFV            EXTERNAL                        265U/4  261U/4  253U/4  249U/4  239U/4  235U/4  227U/4  223U/4  213U/4
                                                        209U/4  201U/4  197U/4  42U/2
        ANOTAT          EXTERNAL                        185U    164U    151U    135U    124U    116U    108U    100U    92U
        AREA2D          EXTERNAL                        74U
        DISTPL          EXTERNAL                        257U    251U    231U    225U    205U    199U
        EOS             EXTERNAL                        173U    144U
        FZXY            EXTERNAL                        269U    263U    243U    237U    217U    211U
        HEADIN          EXTERNAL                        254U    250U    228U    224U    202U    198U    81U     80U
        HEIGHT          EXTERNAL                        79U
        NOBRDR          EXTERNAL                        73U
        PAGE            EXTERNAL                        72U
        PWRIT           EXTERNAL                        266U    262U    240U    236U    214U    210U    84U     83U
        SHDCHR          EXTERNAL                        76U
        SWISSM          EXTERNAL                        77U
        XNAME           EXTERNAL                        182U    161U    148U    132U    121U    113U    105U    97U     89U
        YNAME           EXTERNAL                        183U    162U    149U    133U    122U    114U    106U    98U     90U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      ---------------------------------

      A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
-------------------------------

NDM1      = 72                                                   NDM12     = 6660
NDM2      = 88                                                   NHV       = 88
NNNX      = 72                                                   NNNX2     = 74
NNNXP     = 73                                                   NNNY      = 33
NNNY2     = 35                                                   NNNYP     = 34
NNNZ      = 88                                                   NNNZ2     = 90
NNNZP     = 89                                                   NNYZ2D    = 3150
```

NSPEC = 1 NVAR = 5

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
240	IP	69	274	573a	2043
120	NV	127	129	1031d	34
120	NH	128	129	1042b	17
135	NV	138	140	1113d	34
135	NH	139	140	1124b	17
150	NV	141	145	1152c	64
140	NH	142	143	1164b	11
1002	JJV	154	158	1265b	32
1002	IIH	156	158	1304b	7
1011	NV	167	169	1345d	34
1011	NH	168	169	1356b	17
1013	NV	170	174	1404c	64
1012	NH	171	172	1416b	11
1015	JJV	175	179	1474a	31
1015	IIH	177	179	1513a	7
1022	JJV	188	191	1554c	25
1022	IIH	189	191	1565a	11
180	NV	203	204	1721b	6
185	NV	215	216	2050c	6
200	NV	229	230	2200b	6
205	NV	241	242	2327c	6
220	NV	255	256	2457b	6
225	NV	267	268	2606c	6

BLOCK NAMES AND LENGTHS IN OCTAL

2641	-PROF3D	57	#IB	706	#CL	126500	PLTVAR	30602	-HOLDER	1116	-GRDVAR
264	-HYDFIX	14	GLOVAR	116030	LOCOM1	35	GRFCOM				

STATIC SPACE (IN OCTAL)

B SAVE:	47
I SAVE:	3
CONSTANTS:	402
VARIABLES:	45
TEMPORARIES:	713
CODE:	2172
TOTAL:	3626

```
  1.          subroutine fileio (ifetch,nf)
       c  ----------------------------------------------------------
       c     performs disk reads using the CFILIB random access I/O routine
       c     RDABSF. checks for I/O completion before returning to calling
       c     program.
       c  ------- array dimensions ---------------------------------
  2.          parameter (nnnx=72, nnny=33, nnnz=88)
  3.          parameter (nnyz2d=(nnny+2)*(nnnz+2))
       c-------------------- hydro arrays ---------------------------
  4.          parameter (nspec=1, nvar=4+nspec, npl=1)
  5.          parameter (jkvar=nvar*nnyz2d)
  6.          real dvar(nnyz2d,nvar,npl)
       c
  7.          common /hydvar/ dvar
  8.          real var(nnyz2d,nvar,npl)
  9.          pointer (ipvar,var)
       c-------------------- local declarations ---------------------
 10.          logical lfetch, lstore
 11.          character *8 file
       c
       c     set pointer.
 12.          ipvar=loc(dvar)
       c
       c     perform I/O.
 13.          call rdabsf (lunit,var(1,1,nf),jkvar,(ifetch-1)*jkvar)
 14.          return
       c
 15.          entry filefo(file,lun,nsiz)
       c-------------------------------------------------------------
       c     entry FILEFO opens the disk file.
       c     the arguments:
       c      file - a real variable containing the hollerith string filename
       c      lun    logical unit number
       c      nsiz   file size (words)
       c-------------------------------------------------------------
 16.          lunit=lun
 17.          nsize=nsiz
       c
       c     open the disk file.
 18.          call assign (lunit,file,0)
 19.          call famwait (lunit,1)
 20.          call famsiz (lunit,nsize)
 21.          return
       c
 22.          entry filefc
       c  ----------------------------------------------------------
       c     entry FILEFC closes the disk file after ensuring I/O completion.
       c-------------------------------------------------------------
       c     confirm that all previous disk i/o requests have been completed.
 23.          if (unit(lunit)) 10,30,40
       c
       c     close the file.
 24.    10    continue
 25.          call close (lunit)
 26.          return
       c
```

```
27.          entry fileid
        c    ----------------------------------------------------------
        c    entry FILEID destroys the scratch disk I/O file.
        c    ----------------------------------------------------------
28           call drabsf (lunit,file,0,ierr)
29           write (6,70) ierr
30.          return
        c
31.          entry fileck
        c    ----------------------------------------------------------
        c    entry FILECK waits for all pending I/O requests to be completed
        c    before returning to the calling program.
        c    ----------------------------------------------------------
32.          if (unit(lunit)) 20,30,40
        c
33.     20   return
        c
        c    error  messages.
34.     30   write (6,50) lunit
35.          stop
        c
36.     40   write (6,60) lunit
37.          stop
        c
38      50 format (/' EOF or EOI condition present for unit ','lunit = ',i3)
39.     60 format (///' I/O error occured for unit lunit = ',i3)
40.     70 format (//1x,'ierr from subroutine DRABSF=',i3)
41.        end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

  NUMBER USE    SOURCE PROGRAM REFERENCES

    10 UNDEF**     24L      23L      23J
    20 UNDEF**     33L      32L      32J
    30 217c        34L      32J      23J
    40 230a        36L      32J      23J
    50 FN          38L      34W
    60 FN          39L      36W
    70 FN          40L      29W
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
  (FN-FORMAT NUMBER, UNDEF**-UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

 ADDRESS NAME      TYPE MAIN USAGE   BLOCK        SOURCE PROGRAM REFERENCES

         $STOP          EXTERNAL                    37U    35U
         $WFF           EXTERNAL                    36U    34U    29U
         $WFI           EXTERNAL                    36U    34U    29U
         $WFV           EXTERNAL                    36U    34U    29U
         ASSIGN         EXTERNAL                    18U
         CLOSE          EXTERNAL                    25U
         DRABSF         EXTERNAL                    28U
       0 DVAR      R    3DIM ARRAY   HYDVAR         12P    7D     6D
         FAMSIZ         EXTERNAL                    20U
         FAMWAIT        EXTERNAL                    19U
       3 FILE      CH   VARIABLE     DUM.ARG.       28P    18P    15D    11D
     204 FILECK         ENTRY                       31D/2
     136 FILEFC         ENTRY                       22D/2
     155 FILEFO         ENTRY                       27D/2
     102 FILEFI         ENTRY                       15D/2
      50 FILEIO         ENTRY                       31D    27D    22D    15D    1D/2
      46 IERR      I    VARIABLE                    29U    28P
       1 IFETCH    I    VARIABLE     DUM.ARG.       13U    1D
      41 IPVAR     *I   VARIABLE                    12S    9P
         JKVAR     I    PARAMETER                   13U    13P    5S
         LOC       I    INLINE FUNCT                12U
       4 LUN       I    VARIABLE     DUM.ARG.       16U    15D
      44 LUNIT     I    VARIABLE                    36U    34U    32P    28P    25P    23P    20P    19P    18P
                                                    16S    13P
         MAIN.          ENTRY
       2 NF        I    VARIABLE     DUM.ARG.       13U    1D
         NNNX      I    PARAMETER                   2S
         NNNY      I    PARAMETER                   2S
         NNNZ      I    PARAMETER                   2S
         NNYZ2D    I    PARAMETER                   8P     6P     3S
         NPL       I    PARAMETER                   8P     6P     4S
       5 NSIZ      I    VARIABLE     DUM.ARG.       17U    15D
      45 NSIZE     I    VARIABLE                    20P    17S
         NSPEC     I    PARAMETER                   4S
```

```
       NVAR       I  PARAMETER                      8P      6P      45
       RDABSI        EXTERNAL                      13U
       UNIT       R  EXTERNAL                      32U     23U
     1 VAR        R  3DIM ARRAY   POINTEE          13P      9P      8D
TABLE OF EXTERNAL NAMES
-----------------------

       $STOP         EXTERNAL                      37U     35U
       $WFF          EXTERNAL                      36U     34U     29U
       $WFI          EXTERNAL                      36U     34U     29U
       $WFV          EXTERNAL                      36U     34U     29U
       ASSIGN        EXTERNAL                      18U
       CLOSE         EXTERNAL                      25U
       DRABSF        EXTERNAL                      28U
       FAMSIZ        EXTERNAL                      20U
       FAMWAIT       EXTERNAL                      19U
       RDABSF        EXTERNAL                      13U
       UNIT       R  EXTERNAL                      32U     23U
     ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
     -------------------------------
     A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
     D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
     E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
     I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
     J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
     L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
     N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
-------------------------------

 JKVAR     = 15750                                              NNNX      = 72
 NNNY      = 33                                                 NNNZ      = 88
 NNYZ2D    = 3150                                               NPL       = 1
 NSPEC     = 1                                                  NVAR      = 5

 BLOCK NAMES AND LENGTHS IN OCTAL

    241-FILEIO          5 #IB              77 #CL            36606 HYDVAR
STATIC SPACE (IN OCTAL)
-----------------------
B SAVE:                4
I SAVE:                0
CONSTANTS:            37
VARIABLES:            10
TEMPORARIES:         100
CODE:                172
                  ------
TOTAL:               345
```

```
  1.            subroutine pltarr (ii,jj,kk,nloc)
          c ---------------------------------------------------------------
          c     fills plot arrays rvx, rvy, rvz, erg, rho, rha, rhb.
          c ---------------------- array dimensions -------------------------
  2.            parameter (nnnx=72, nnny=33, nnnz=88)
  3.            parameter (nnny2=nnny+2, nnnz2=nnnz+2)
  4.            parameter (nnyz2d=nnny2*nnnz2)
          c ----------------------- hydro arrays ----------------------------
  5.            parameter (nspec=1, nvar=4+nspec, npl=1)
  6.            real var(nnyz2d,nvar,npl)
  7.            common /hydvar/ var
          c------------ fixed arrays of hydrodynamic data -------------------
  8.            parameter (nsum=4+nspec)
  9.            real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 10.            real sumpl(nsum,nnnx)
 11.            real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
          c
 12.            common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
               1 grapz, sum4, shrink, fsky, ratio
          c --------------------- plot variables ----------------------------
 13.            parameter (ndm1=72, ndm2=88)
 14.            real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
 15.            real erg(ndm1,ndm2)
 16.            real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
 17.            common /pltvar/ rvx, rvy, rvz, erg, rho, rha, rhb
          c ------------------ local declarations ---------------------------
 18.            logical lintp, colour, displa, darkhd
 19.            real mplot(14)
 20.            common /grfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
               1 mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
          c
          c     get data along the appropriate slice.
 21.            go to (10,90,170), mslice
          c
          c     X-Y slice.
 22.       10   jk=kk*nnny2+1
 23.            do 20 j=jmin,jmax
 24.            rvx(ii,j)=var(j+jk,1,nloc)
 25.            rvy(ii,j)=var(j+jk,2,nloc)
 26.            rvz(ii,j)=var(j+jk,3,nloc)
 27.       20   erg(ii,j)=var(j+jk,4,nloc)
 28.            go to (70,50,30), nspec
 29.       30   do 40 j=jmin,jmax
 30.       40   rhb(ii,j)=cvmgt(var(j+jk,7,nloc),0.0,var(j+jk,7,nloc).gt.
               1 1.e-6*var(j+jk,5,nloc))
 31.       50   do 60 j=jmin,jmax
 32.       60   rha(ii,j)=cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc).gt.
               1 1.e-6*var(j+jk,5,nloc))
 33.       70   do 80 j=jmin,jmax
 34.       80   rho(ii,j)=var(j+jk,5,nloc)
 35.            go to 250
          c
          c     Y-Z slice.
 36.       90   do 100 k=kmin,kmax
 37.            jk=k*nnny2+1
 38.            do 100 j=jmin,jmax
```

```
1574    39.                 rvx(j,k)=var(j+jk,1,nloc)
1575    40.                 rvy(j,k)=var(j+jk,2,nloc)
1576    41.                 rvz(j,k)=var(j+jk,3,nloc)
1577    42.       100       erg(j,k)=var(j+jk,4,nloc)
1578    43.                 go to (150,130,110), nspec
1579    44.       110       do 120 k=kmin,kmax
1580    45.                 jk=k*nnny2+1
1581    46.                 do 120 j=jmin,jmax
1582    47.       120       rhb(j,k)=cvmgt(var(j+jk,7,nloc),0.0,var(j+jk,7,nloc).gt.
1583                1        1.e-6*var(j+jk,5,nloc))
1584    48.       130       do 140 k=kmin,kmax
1585    49.                 jk=k*nnny2+1
1586    50.                 do 140 j=jmin,jmax
1587    51.       140       rha(j,k)=cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc).gt.
1588                1        1.e-6*var(j+jk,5,nloc))
1589              c  140     rha(j,k)=cvmgt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc).ge.
1590              c      1    dmin.and.var(j+jk,6,nloc).le.dmax)
1591    52.       150       do 160 k=kmin,kmax
1592    53.                 jk=k*nnny2+1
1593    54.                 do 160 j=jmin,jmax
1594    55.       160       rho(j,k)=var(j+jk,5,nloc)
1595    56.                 go to 250
1596              c
1597              c       X-Z slice.
1598    57.       170       jk=jj+1
1599    58.                 do 180 k=kmin,kmax
1600    59.                 jkl=jk+k*nnny2
1601    60.                 rvx(ii,k)=var(jkl,1,nloc)
1602    61.                 rvy(ii,k)=var(jkl,2,nloc)
1603    62.                 rvz(ii,k)=var(jkl,3,nloc)
1604    63.       180       erg(ii,k)=var(jkl,4,nloc)
1605    64.                 go to (230,210,190), nspec
1606    65.       190       do 200 k=kmin,kmax
1607    66.       200       rhb(ii,k)=cvmgt(var(jk+k*nnny2,7,nloc),0.0,
1608                1        var(jk+k*nnny2,7,nloc).gt.1.e-6*var(jk+k*nnny2,5,nloc))
1609    67.       210       do 220 k=kmin,kmax
1610    68.       220       rha(ii,k)=cvmgt(var(jk+k*nnny2,6,nloc),0.0,
1611                1        var(jk+k*nnny2,6,nloc).gt.1.e-6*var(jk+k*nnny2,5,nloc))
1612              c  220     rha(ii,k)=cvmgt(var(jk+k*nnny2,6,nloc),0.0,
1613              c      1    var(jk+k*nnny2,6,nloc).ge.dmin.and.var(jk+k*nnny2,6,nloc)
1614              c      2    .le.dmax)
1615    69.       230       do 240 k=kmin,kmax
1616    70.       240       rho(ii,k)=var(jk+k*nnny2,5,nloc)
1617    71.       250       continue
1618    72.                 return
1619    73.               end
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          23, P=     27b
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          29, P=     75c
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          31, P=     123b
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          33, P=     151a
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          38, P=     175c
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          46, P=     247c
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          50, P=     304c
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          54, P=     341c
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          65, P=     422d
PLTARR                  VECTOR LOOP BEGINS AT SEQ. NO.          67, P=     451c
```

PLTARR VECTOR LOOP BEGINS AT SEQ NO. 69, P= 5000

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER USE	SOURCE PROGRAM REFERENCES		
10 27b	22L	21J	
20 UNDEF**	27I	23F	
20A43c	23I		
20B70a	23I		
30 75c	29L	28J	
40 UNDEF**	30L	29F	
40A110b	29I		
40B123b	29I		
50 123b	31L	28J	
60 UNDEF**	32I	31E	
60A136a	31L		
60B151a	31L		
70 151a	33L	28J	
80 UNDEF**	34L	33E	
80A162c	33I		
80B172a	33L		
90 172c	36L	21J	
100 UNDEF**	42I	38E	36F
100A175c	36L		
100B237a	36L		
100C211d	38L		
100D234b	38I		
110 244c	44I	43J	
120 UNDEF**	47L	46F	44F
120A247c	44I		
120B301c	44I		
120C265a	46L		
120D276d	46L		
130 301c	48L	43J	
140 UNDEF**	51L	50F	48F
140A304c	48I		
140B336c	48I		
140C322a	50I		
140D333d	50I		
150 336c	52I	43J	
160 UNDEF**	55I	54F	52F
160A341c	52I		
160B370b	52I		
160C356c	54I		
160D365c	54I		
170 370d	57I	21J	
180 UNDEF**	63I	58F	
180A403c	58I		
180B415b	58I		
190 422d	65I	64J	
200 UNDEF**	66I	65F	
200A435c	65I		
200B451c	65I		
210 451c	67I	64J	
220 UNDEF**	68I	67F	

```
   220A464b        67I
   2208500b        67I
   230 500b        69I       64J
   240 UNDEF**     70I       69F
   240A511d        69L
   2408521d        69L
   250 521d        71L       56J       35J
00002 25d          21W
00003 74a          28W
00004 243a         43W
00005 421b         64W
 (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
-----------------------------
```

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES								
	CVMGT		INLINE FUNCT.		68U	66U	51U	47U	32U	30U			
45100	ERG	*R	2DIM ARRAY	PLTVAR	63S	42S	27S	17D	15D				
1	II	I	VARIABLE	DUM.ARG.	70U	68U	66U	63U	62U	61U	60U	34U	32U
					30U	27U	26U	25U	24U	1D			
5	J	I	VARIABLE		55U/2	54I	51U/4	50I	47U/4	46I	42U/2	41U/2	40U/2
					39U/2	38I	34U/2	33J	32U/4	31I	30U/4	29I	27U/2
					26U/2	25U/2	24U/2	23I					
2	JJ	I	VARIABLE	DUM.ARG.	57U	1D							
4	JK	I	VARIABLE		70U	68U/3	66U/3	59U	57S	55U	53S	51U/3	43S
					47U/3	45S	42U	41U	40U	39U	37S	34U	32U/3
					30U/3	27U	26U	25U	24U	22S			
7	JK1	I	VARIABLE		63U	62U	61U	60U	59S				
4	JMAX	I	VARIABLE	GRFCOM	54N	50N	46N	38N	33N	31N	29N	23N	20D
3	JMIN	I	VARIABLE	GRFCOM	54N	50N	46N	38N	33N	31N	29N	23N	20D
6	K	I	VARIABLE		70U/2	69I	68U/4	67I	66U/4	65I	63U	62U	61U
					60U	59U	58I	55U	53U	52I	51U	49U	48I
					47U	45U	44I	42U	41U	40U	39U	37U	36I
3	KK	I	VARIABLE	DUM.ARG.	22U	1D							
6	KMAX	I	VARIABLE	GRFCOM	69N	67N	65N	58N	52N	48N	44N	36N	20D
5	KMIN	I	VARIABLE	GRFCOM	69N	67N	65N	58N	52N	48N	44N	36N	20D
	MAIN.		ENTRY										
0	MSLICE	I	VARIABLE	GRFCOM	21P	20D							
	NDM1	I	PARAMETER		16P/3	15P	14P/3	13S					
	NDM2	I	PARAMETER		16P/3	15P	14P/3	13S					
4	NLOC	I	VARIABLE	DUM.ARG.	70U	68U/3	66U/3	63U	62U	61U	60U	55U	51U/3
					47U/3	42U	41U	40U	39U	34U	32U/3	30U/3	27U
					26U	25U	24U	1D					
	NNNX	I	PARAMETER		10P	2S							
	NNNY	I	PARAMETER		2S								
	NNNY2	I	PARAMETER		70U	68U/3	66U/3	59U	53U	43U	45U	37U	22U
					3S								
	NNNZ	I	PARAMETER		2S								
	NNNZ2	I	PARAMETER		11P/4	9P/3	3S						
	NNYZ2D	I	PARAMETER		6P	4S							
	NPL	I	PARAMETER		6P	5S							
	NSPEC	I	PARAMETER		64P	43P	28P	5S					

	NSUM	I	PARAMETER		10P	8S								
	NVAR	I	PARAMETER		6P	5S								
15	PLTARR		ENTRY		1D/2									
75700	RHA	*R	2D1M ARRAY	PLTVAR	68S	51S	32S	17D	16D					
112200	RHB	*R	2D1M ARRAY	PLTVAR	66S	47S	30S	17D	16D					
61400	RHD	*R	2D1M ARRAY	PLTVAR	70S	55S	34S	17D	16D					
0	RVX	*R	2D1M ARRAY	PLTVAR	60S	39S	24S	17D	14D					
14300	RVY	*R	2D1M ARRAY	PLTVAR	61S	40S	25S	17D	14D					
30600	RVZ	*R	2D1M ARRAY	PLTVAR	62S	41S	26S	17D	14D					
0	VAR	R	3D1M ARRAY	HYDVAR	70U	68U	68P/2	66U	66P/2	63U	62U	61U	60U	
					55U	51U	51P/2	47U	47P/2	42U	41U	40U	39U	
					34U	32U	32P/2	30U	30P/2	27U	26U	25U	24U	
					7D	6D								

TABLE OF EXTERNAL NAMES

ABBREVIATIONS USED ABOVE (UNLESS ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

NDM1	= 72	NDM2	= 88
NNNX	= 72	NNNY	= 33
NNNY2	= 35	NNN7	= 88
NNNZ2	= 90	NNYZ2D	= 3150
NPL	= 1	NSPEC	= 1
NSUM	= 5	NVAR	= 5

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
20	J	23	27	43c	25
40	J	21	30	110b	13
60	J	31	32	136a	13
80	J	33	34	162c	10
100	K	36	42	175c	42
100	J	38	42	211d	23
120	K	44	47	247c	33
120	J	46	47	265a	11
140	K	48	51	304c	33
140	J	50	51	322a	11
160	K	52	55	341c	27

160	J	54	55	356c	7
180	K	58	63	403c	12
200	K	65	66	435c	14
220	K	67	68	464b	14
240	K	69	70	511d	10

BLOCK NAMES AND LENGTHS IN OCTAL

525-PLTARR 42-#IB 0-#CL 36606-HYDVAR 1741-HYDFIX 126500-PLTVAR
35-GRFCOM

STATIC SPACE (IN OCTAL)

B SAVE	41
T SAVE	1
CONSTANTS	2
VARIABLES	6
TEMPORARIES	0
CODE	515
TOTAL	567

```
  1.          subroutine restrt (istep,mcore,restf,label)
        c-------------------------------------------------------------------
        c     read data from dump file for graphics.
        c----------------------------- array dimensions --------------------
  2.          parameter (nnnx=72, nnny=33, nnnz=88)
  3.          parameter (nnnxp=nnnx+1, nnnyp=nnny+1, nnnzp=nnnz+1)
  4.          parameter (nnnx2=nnnx+2, nnny2=nnny+2, nnnz2=nnnz+2)
  5.          parameter (nnyz2d=nnny2*nnnz2)
        c---------------------- hydro arrays -------------------------------
  6.          parameter (nspec=1, nvar=4+nspec, npl=1)
  7.          parameter (jkpl=nnnx*nvar*nnyz2d, jkall=nnnx2*nvar*nnyz2d)
  8.          real dvar(nnyz2d,nvar,npl)
  9.          common /hydvar/ dvar
 10.          real var(nnyz2d,nvar,npl)
 11           pointer (ipvar,var)
        c------------ fixed arrays of hydrodynamic data --------------------
 12.          parameter (nsum=4+nspec, nfix=7*nnnz2+nnnx*nsum+3)
 13.          real rcamb(nnnz2), pcamb(nnnz2), ecamb(nnnz2)
 14.          real sumpl(nsum,nnnx)
 15.          real rcmin(nnnz2), gravz(nnnz2), grapz(nnnz2), fsky(nnnz2)
        c
 16.          common /hydfix/ rcamb, pcamb, ecamb, sumpl, rcmin, gravz,
           1    grapz, sum4, shrink, fsky, ratio
        c----------------- grid arrays -------------------------------------
 17.          parameter (ngrd=2*(nnnxp+nnnyp+nnnzp))
 18.          real xcor(nnnxp), ycor(nnny), zcor(nnnz)
 19           real twodx(nnnxp), twody2(nnny2), twodz2(nnnz2)
 20.          real dtodx(nnnxp), dtody2(nnny2), dtodz2(nnnz2)
        c
 21.          common /grdvar/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
           1    , dtody2, dtodz2
        c------------------------ global variables -------------------------
 22.          parameter (ntim=9)
 23.          common /glovar/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
           1    , nstep, lllx, llly, lllz
        c------------------- body indices ----------------------------------
 24.          parameter (nnbd=1, nbodx=1, nbodz=1)
 25.          parameter (nbody=1+2*nnbd*(1+nbodx*(1+nbodz)))
        c       integer imnbd(nnbd), imxbd(nnbd)
        c       integer kmnbd(nbodx,nnbd), kmxbd(nbodx,nnbd)
        c       integer jmnbd(nbodz,nbodx,nnbd), jmxbd(nbodz,nbodx,nnbd)
        cc
        c       common /bodcom/ nbod, imnbd, imxbd, kmnbd, kmxbd, jmnbd, jmxb
        c----------------------- station arrays ----------------------------
 26.          parameter (mxx=1, nsta=1)
 27.          parameter (nstat=1+4*mxx*nsta+mxx+3*nsta+1)
 28.          logical lstat
 29.          real rhs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta)
 30.          real prs(mxx,nsta), tme(mxx), xs(nsta), ys(nsta), zs(nsta)
 31.          common /stats/ lstat, rhs, vls, gms, prs, tme, xs, ys, zs, nxx
        c
        c------------------------------ particles---------------------------
 32.          parameter (nop=70,nburst=2)
 33.          parameter (npartl=3*nop*nburst+nburst)
 34.          real xp(nop,nburst), zp(nop,nburst)
 35.          integer nopp(nburst),nchk(nop,nburst)
```

```
36.        common /partl/ xp,zp,nopp,nchk
     c
     c                     active grid boundaries ----- --- ---
37.        parameter (nscn=6, nact=17)
38.        logical lscan, lsides, ltop, botm, lrear, lfront
39.        common /scan/ lscan, lsides, ltop, botm, lrear, lfront
40.        common /active/ ixr, ixl, ixro, ixlo, jyp, jya, jypo, jyao,
         1 kzt, kzb, kzto, kzbo, jks1, jks2, jkd1, jkd2, ktherm
     c                 local declarations --------------- -- ---
41.        parameter (nlab=5, nch=8*nlab)
42.        character *8 restf
43.        character *40 label
44         logical incore
     c
45.        parameter (nsize=jkall+ntim+nlab+nscn+nact+ngrd+nfix+nbody
         1 +nstat+npartl)
46.        data lunit /16/, ldump /17/
     c
     c     set pointer
47.        ipvar=loc(dvar)
     c
     c     open dump file.
48.        call ffeto (restf,lunit,nsize)
     c
     c     first read all the data stored at the end of the file. the
     c     conserved variables on the nnnx2 planes are stored at the
     c     beginning
     c     read file number, step, time, timestep and label.
49.        call rdabsf (lunit,time,ntim,jkall)
50.        call rdabsf (lunit,label,nlab,jkall+ntim)
51.        call fileck
     c
     c     write messages.
52.        write (6,30) nfile,nstep,time
53.        istep=nstep
     c
     c     error exits.
54.        if (lllx.ne.nnnx.or.llly.ne.nnny.or.lllz.ne.nnnz) then
55.        write (6,40) lllx,llly,lllz,nnnx,nnny,nnnz
56.        stop
57.        endif
     c
     c     check if this is a time history run.
58         if (lstat) then
59         iadrs=jkall+ntim+nlab+nscn+nact+ngrd
60.        go to 10
61.        endif
     c
     c     read active grid switches and boundary indices.
62.        call rdabsf (lunit,lscan,nscn,jkall+ntim+nlab)
63.        call rdabsf (lunit,ixr,nact,jkall+ntim+nlab+nscn)
     c
     c     read coordinate data.
64.        call rdabsf (lunit,xcor,ngrd,jkall+ntim+nlab+nscn+nact)
65.        iadrs=jkall+ntim+nlab+nscn+nact+ngrd
     c
```

```
1732          c     read fixed arrays of hydrodynamic data.
1733     66.            call rdabsf (lunit,rcamb,nfix,iadrs)
1734          c
1735          c     read station arrays.
1736     67.       10      call rdabsf (lunit,lstat,nstat,iadrs+nfix)
1737          c
1738          c     read body indices.
1739          c           call rdabsf (lunit,nbod,nbody,iadrs+nfix+nstat)
1740          c
1741          c
1742          c    read particles
1743          c
1744     68.            call rdabsf (lunit,xp,nparti,iadrs+nfix+nstat+nbody)
1745          c
1746     69.            return
1747          c
1748     70.            entry readin
1749          c-------------------------------------------------------------------
1750          c     read in all the planes for incore calculation.
1751          c-------------------------------------------------------------------
1752     71.            call rdabsf (lunit,var,jkpl,0)
1753     72.            call filefc
1754     73.            return
1755          c
1756     74.            entry datain(imin,imax,jj,kk)
1757          c-------------------------------------------------------------------
1758          c     fill the plot arrays.
1759          c-------------------------------------------------------------------
1760     75.            do 20 ii=imin,imax
1761     76.            if ( not.incore) then
1762     77.            call fileck
1763     78.            call fileio (ii,1)
1764     79.            call fileck
1765     80.            endif
1766     81.            iloc=cvmgt(ii,1,incore)
1767     82.            call pltarr (ii,jj,kk,iloc)
1768     83.       20      continue
1769     84.            return
1770          c
1771     85.       30 format (1x,'reading file',i3,' at time, step=',e12.5,i6)
1772     86.       40 format ('***error***  wrong restart file'/1x,'dump file contains n
1773                  1nnx,nnny,nnnz=',3i4/1x,'fast3d expects nnnx,nnny,nnnz=',3i4/1x,'ex
1774                  2ecution terminated')
1775     87.          end
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE	PROGRAM REFERENCES
10	227c	67L	60J
20	UNDEF**	83L	75E
20A	312c	75L	
20B	331d	75L	
30	FN	85I	52W
40	FN	86L	55W
00002	167c	54I	
00003	174d	58I	
00004	320b	76I	

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES							
	$STOP		EXTERNAL		56U							
	$WEF		EXTERNAL		55U	52U						
	$WFI		EXTERNAL		55U	52U						
	$WFV		EXTERNAL		55U/6	52U/3						
	CVMGT		INLINE FUNCT.		81U							
271	DATAIN		ENTRY		74D/2							
0	DVAR	R	3DIM ARRAY	HYDVAR	47P	9D	8D					
	FILECK		EXTERNAL		79U	77U	51U					
	FILEFC		EXTERNAL		72U							
	FILEFO		EXTERNAL		48U							
	FILEIO		EXTERNAL		78U							
60	IADRS	I	VARIABLE		68P	67P	66P	65S	59S			
61	II	I	VARIABLE		82P	81P	78P	75I				
62	ILOC	I	VARIABLE		82P	81S						
6	IMAX	I	VARIABLE	DUM.ARG.	75N	74D						
5	IMIN	I	VARIABLE	DUM.ARG.	75N	74D						
2	INCORR	I	VARIABLE	DUM.ARG.	81P	76U	44D	1D				
55	IPVAR	*I	VARIABLE		47S	11P						
1	ISTEP	*I	VARIABLE	DUM.ARG.	53S	1D						
0	IXR	I	VARIABLE	ACTIVE	63P	40D						
7	JJ	I	VARIABLE	DUM.ARG.	82P	74D						
	JKALL	I	PARAMETER		65U	64P	63P	62P	59U	50P	49P	7S
	JKPL	I	PARAMETER		71P	7S						
10	KK	I	VARIABLE	DUM.ARG.	82P	74D						
4	LABEL	CH	VARIABLE	DUM.ARG.	50P	43D	1D					
57	LDUMP	*I	VARIABLE		46S							
11	LLLX	I	VARIABLE	GLOVAR	55U	54U	23D					
12	LLLY	I	VARIABLE	GLOVAR	55U	54U	23D					
13	LLLZ	I	VARIABLE	GLOVAR	55U	54U	23D					
	LOC	I	INLINE FUNCT		47U							
0	ISCAN	L	VARIABLE	SCAN	62P	39D	38D					
0	ISTAT	L	VARIABLE	STATS	67P	58U	31D	28D				

56	UNIT	I	VARIABLE		71P	68P	67P	66P	64P	63P	62P	50P	49P
					48P	46S							
	MAIN.		ENTRY										
	MXX	I	PARAMETER		30P/2	29P/3	26S						
	NACT	I	PARAMETER		65U	64U	63P	59U	37S				
	NBODX	I	PARAMETER		24S								
	NBODY	I	PARAMETER		68U	25S							
	NBODZ	I	PARAMETER		24S								
	NBURST	I	PARAMETER		35P/2	34P/2	32S						
	NCH	I	PARAMETER		41S								
7	NFILE	I	VARIABLE	GLOVAR	52U	23D							
	NFIX	I	PARAMETER		68U	67U	66P	12S					
	NGRD	I	PARAMETER		65U	64P	59U	17S					
	NLAB	I	PARAMETER		65U	64U	63U	62U	59U	50P	41S		
	NNRD	I	PARAMETER		24S								
	NNNX	I	PARAMETER		55U	54U	14P	2S					
	NNNX2	I	PARAMETER		4S								
	NNNXP	I	PARAMETER		20P	19P	18P	3S					
	NNNY	I	PARAMETER		55U	54U	18P	2S					
	NNNY2	I	PARAMETER		20P	19P	4S						
	NNNYP	I	PARAMETER		3S								
	NNNZ	I	PARAMETER		55U	54U	18P	2S					
	NNNZ2	I	PARAMETER		20P	19P	15P/4	13P/3	4S				
	NNNZP	I	PARAMETER		3S								
	NNYZ2D	I	PARAMETER		10P	8P	5S						
	NOP	I	PARAMETER		35P	34P/2	32S						
	NPARTL	I	PARAMETER		68P	33S							
	NPL	I	PARAMETER		10P	8P	6S						
	NSCN	I	PARAMETER		65U	64U	63U	62P	59U	37S			
	NSIZE	I	PARAMETER		48P	45S							
	NSPEC	I	PARAMETER		6S								
	NSTA	I	PARAMETER		30P/4	29P/3	26S						
	NSTAT	I	PARAMETER		68U	67P	27S						
10	NSTEP	I	VARIABLE	GLOVAR	53U	52U	23D						
	NSUM	I	PARAMETER		14P	12S							
	NTIM	I	PARAMETER		65U	64U	63U	62U	59U	50U	49P	22S	
	NVAR	I	PARAMETER		10P	8P	6S						
	PLTARR		EXTERNAL		82U								
0	RCAMB	R	1DIM ARRAY	HYDFIX	66P	16D	13D						
	RDABSF		EXTERNAL		71U	68U	67U	66U	64U	63U	62U	50U	49U
252	READIN		ENTRY		70D/2								
3	RESTF	CH	VARIABLE	DUM.ARG.	48P	42D	1D						
64	RESTRT		ENTRY		74D	70D	1D/2						
3	TIME	R	VARIABLE	GLOVAR	52U	49P	23D						
1	VAR	R	3DIM ARRAY	POINTER	71P	11P	10D						
0	XCOR	R	1DIM ARRAY	GRDVAR	64P	21D	18D						
0	XP	R	2DIM ARRAY	PARTL	68P	36D	34D						

TABLE OF EXTERNAL NAMES

	$STOP	EXTERNAL	56U		
	$WEF	EXTERNAL	55U	52U	
	$WFI	EXTERNAL	55U	52U	
	$WFV	EXTERNAL	55U/6	52U/3	
	FILECK	EXTERNAL	79U	77U	51U

```
        FILEFC       EXTERNAL                       72U
        FILEFO       EXTERNAL                       48U
        FILEIO       EXTERNAL                       78U
        PLTARR       EXTERNAL                       82U
        RDABST       EXTERNAL                       71U     68U     67U     66U     64U     63U     62U     50U     49U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      -------------------------------------
      A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OR IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL
   TABLE OF PARAMETERS ENCOUNTERED
   -------------------------------

    JKALL    = 1165500                                          JKPL     = 1134000
    MXX      = 1                                                NACT     = 17
    NBODX    = 1                                                NBODY    = 7
    NBODZ    = 1                                                NBURST   = 2
    NCH      = 40                                               NFIX     = 993
    NGPD     = 392                                              NLAB     = 5
    NNBD     = 1                                                NNNX     = 72
    NNNX2    = 74                                               NNNXP    = 73
    NNNY     = 33                                               NNNY2    = 35
    NNNYP    = 34                                               NNNZ     = 88
    NNNZ2    = 90                                               NNNZP    = 89
    NNYZ2D   = 3150                                             NOP      = 70
    NPARTL   = 422                                              NPL      = 1
    NSCN     = 6                                                NSIZE    = 1167361
    NSPEC    = 1                                                NSTA     = 1
    NSTAT    = 10                                               NSUM     = 5
    NTIM     = 9                                                NVAR     = 5

   TABLE OF LOOPS ENCOUNTERED
   --------------------------
    LABEL INDEX      FROM    TO   ADDRESS - LENGTH

      20 II           75     83     312c       20
    BLOCK NAMES AND LENGTHS IN OCTAL

       335-RESTRT          24-#IB            141 #CL           36606-HYDVAR        1741-HYDFIX        1116-GRDVAR
        14-GLOVAR          12 STATS          646 PARTI             6-SCAN            21-ACTIVE
   STATIC SPACE (IN OCTAL)
   -----------------------
   B SAVE:              10
   T SAVE:               4
   CONSTANTS:           53
```

VARIABLES:	10
TEMPORARIES:	151
CODE	252
TOTAL	522

```
1776      1.         subroutine colcon (arr,imin,jmin,kh,kv)
1777           c------------------------------------------------------------------
1778           c     produces color contour plots
1779           c     this routine divides the contours into five ranges, and draws them
1780           c     in different colors according to which range their values fall in.
1781           c------------------------------------------------------------------
1782      2.         parameter (ndm1=72, ndm2=88, nr=5)
1783      3          real arr(ndm1,ndm2), hue(nr)
1784      4.         common /holder/ space(ndm1,ndm2),valmin,valrng
1785      5.         logical displa, darkhd
1786      6.         common /grfcom/ mslice, iamin, imax, jamin, jmax, kmin, kmax,
1787              1    mplot, lintp, colour, dmin, dmax, displa, il, jl, darkhd
1788      7.         data hue /3hred,4hblue,5hgreen,6hyellow,5hclear/
1789           c
1790           c     determine maximum and minimum values.
1791      8.         armax=0.0
1792      9          armin=1.e50
1793     10.         do 10 j=jmin,jmin+kv-1
1794     11.         imx=ismax(kh,arr(imin,j),1)+imin-1
1795     12.         arrmax=amax1(arrmax,arr(imx,j))
1796     13.         imn=ismin(kh,arr(imin,j),1)+imin-1
1797     14.         arrmin=amin1(arrmin,arr(imn,j))
1798     15.    10   continue
1799     16.         drange=(arrmax-arrmin)/float(nr)
1800           c
1801           c     plot contours for each range.
1802           c     DISSPLA plots the colour contours in the subroutine discol.  NCAR
1803           c     draws the contour lines a few at a time, changing color between.
1804     17.         if (displa) then
1805     18.           valmin=arrmin
1806     19.           valrng=drange
1807     20.           call setclr ('RED')
1808     21.           call discol (arr(imin,jmin),ndm1,space,kh,kv)
1809     22.         else
1810     23.           do 30 ir=1,nr
1811     24.           flo=arrmin+float(ir-1)*drange
1812     25.           fhi=arrmin+float(ir)*drange
1813     26.           call flush
1814     27.           call gcolor (hue(ir))
1815     28.           call conrec (arr(imin,jmin),ndm1,kh,kv,flo,fhi,6,-1,0,0)
1816     29.           call flush
1817     30.    30     continue
1818     31.         endif
1819     32.         return
1820     33.       end
O   AT SEQUENCE NUMBER -     11.
 PRNAME COLCON   COMMENT -     DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 12     ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
O   AT SEQUENCE NUMBER -     11.
 PRNAME COLCON   COMMENT -     DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 13     ****************************** P=02506300
 EXPLANATION:  ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE PROGRAM REFERENCES	
10	UNDEF**	15I	10E
10A50c		10I	
10B77a		10I	
30	UNDEF**	30I	23L
30A132d		23L	
30BUNDEF**		23L	
00002	121c	17I	
00003	161a	22I	

(SN=STATEMENT NUMBER, GSN GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF**=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES							
	AMAX1	R	INLINE FUNCT.		12U							
	AMIN1	R	INLINE FUNCT.		14U							
1	ARR	R	2DIM ARRAY	DUM.ARG.	28P	21P	14P	13P	12P	11P	3D	1D
17	ARRMAX	R	VARIABLE		16U	12P	12S	8S				
20	ARRMIN	R	VARIABLE		25U	24U	18U	16U	14P	14S	9S	
31	COLCON		ENTRY		1D/2							
	CONREC		EXTERNAL		28U							
	DISCOL		EXTERNAL		21U							
14	DISPLA	L	VARIABLE	GRFCOM	17U	6D	5D					
24	DRANGE	R	VARIABLE		25U	24U	19U	16S				
27	FHI	R	VARIABLE		28P	25S						
26	FLO	R	VARIABLE		28P	24S						
	FLOAT	R	INLINE FUNCT		25U	24U	16U					
	FLUSH		EXTERNAL		29U	26U						
	GCOLOR		EXTERNAL		27U							
12	HUE	R	1DIM ARRAY		27P	7S	3D					
2	IMIN	I	VARIABLE	DUM.ARG.	28U	21U	13U/2	11U/2	1D			
23	IMN	I	VARIABLE		14U	13S						
22	IMX	I	VARIABLE		12U	11S						
25	IR	I	VARIABLE		27U	25P	24P	23I				
	ISMAX	I	EXTERNAL		11U							
	ISMIN	I	EXTERNAL		13U							
21	J	I	VARIABLE		14U	13U	12U	11U	10I			
3	JMIN	I	VARIABLE	DUM.ARG.	28U	21U	10N/2	1D				
4	KH	I	VARIABLE	DUM.ARG.	28P	21P	13P	11P	1D			
5	KV	I	VARIABLE	DUM.ARG.	28P	21P	10N	1D				
	MAIN.		ENTRY									
	NDM1	I	PARAMETER		28P	21P	4P	3P	2S			
	NDM2	I	PARAMETER		4P	3P	2S					
	NR	I	PARAMETER		23N	16P	3P	2S				
	SETCLR		EXTERNAL		20U							
0	SPACE	R	2DIM ARRAY	HOLDER	21P	4D						
14300	VALMIN	*R	VARIABLE	HOLDER	18S	4D						

```
   14301 VALRNG  *R  VARIABLE      HOLDER          195        4D
TABLE OF EXTERNAL NAMES
-- ----------------- --

       CONREC       EXTERNAL                        28U
       DISCOL       EXTERNAL                        21U
       FLUSH        EXTERNAL                        29U       26U
       GCOLOR       EXTERNAL                        27U
       ISMAX     I  EXTERNAL                        11U
       ISMIN     I  EXTERNAL                        13U
       SETCLR       EXTERNAL                        20U
     ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
     ------------- ---- -----

     A  USED IN FORTRAN  ASSIGN STATEMENT      P  USED IN CALL/FUNC CALL OR ARRAY DEF
     D  DEFINED IN  DECLARATIVE STATEMENT      R  FORMAT  USED  IN A  READ  STATEMENT
     E  STATEMENT NUMBER ENDING A DO LOOP      S  STORED  SO CONTENTS  MAY BE CHANGED
     I  INDEX OF A DO  OR IMPLIED DO LOOP      U  NAME  USED IN EXECUTABLE  STATEMENT
     J  STATEMENT NUMBER USED IN TRANSFER      W  FORMAT  USED IN A  WRITE  STATEMENT
     L  SOURCE LINE OF A STATEMENT NUMBER      *  DEFINED  OR DECLARED  BUT NOT  USED
     N  NAME USED AS A  DO LOOP PARAMETER      ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
--- -- ---------- -----------

 NDM1     = 72                                                NDM2      = 88
 NR       = 5

TABLE OF LOOPS ENCOUNTERED
-- ---------- -----------
 LABEL INDEX      FROM -  TO   ADDRESS - LENGTH

   10 J            10     15        50c       27
   30 IR           23     30       132d       27
 BLOCK NAMES AND LENGTHS IN OCTAL

     164-COLCON            27-#IB             37 #CL            14302-HOLDER           20-GRFCOM
STATIC SPACE (IN OCTAL)
------ ----- -- -----
B SAVE:              23
I SAVE:               4
CONSTANTS:           10
VARIABLES:           20
TEMPORARIES:         37
CODE:               134
                 ------
TOTAL:              252
```

```
 1821
 1822      1.            subroutine discol (arrin,mdim,cnhold,kh,kv)
 1823      2.            real arrin(mdim,kv),cnhold(kh,kv)
 1824      3.            common work(5000)
 1825                c
 1826                c ****************************************************************
 1827                c *    this subroutine uses DISSPLA graphics to draw contour     *
 1828                c *    plots in color.  The actual color calls are in the        *
 1829                c *    user-defined subroutine MYCON.                            *
 1830                c ****************************************************************
 1831                c
 1832                c     store the portion of the array to be plotted
 1833      4.            do 10 i=1,kh
 1834      5.            do 10 j=1,kv
 1835      6.            cnhold(i,j)=arrin(i,j)
 1836      7.       10   continue
 1837                c     set up the contour lines
 1838      8.            call bcomon (5000)
 1839      9.            call conmak (cnhold,kh,kv,'scale')
 1840                c     specify the characteristics of the lines and labels, including their
 1841                c     color.  Color and other characteristics is set in the routine mycon.
 1842     10.            call conlin (0,'MYCON','LABELS',1,3)
 1843     11.            call conang (60.)
 1844     12.            call conthn (0.06)
 1845                c     draw the contour lines
 1846     13.            call contur (1,'labels','draw')
 1847     14.            return
 1848     15.            end
DISCOL                VECTOR LOOP BEGINS AT SEQ. NO.          5, P=      36d
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE PROGRAM REFERENCES		
10	UNDEF**	7I	5F	4F
10A36d		4I		
10B63c		4I		
10C52b		5L		
10D60d		5L		

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES			
1	ARRIN	R	2DIM ARRAY	DUM.ARG.	6U	2D	1D	
	BCOMON		EXTERNAL		8U			
3	CNIOLD	R	2DIM ARRAY	DUM.ARG.	9P	6S	2P	1D
	CONANG		EXTERNAL		11U			
	CONLIN		EXTERNAL		10U			
	CONMAK		EXTERNAL		9U			
	CONTIN		EXTERNAL		12U			
	CONTUR		EXTERNAL		13U			
25	DISCOL		ENTRY		1D/2			
22	I	I	VARIABLE		6U/2	4I		
23	J	I	VARIABLE		6U/2	5I		
4	KH	I	VARIABLE	DUM.ARG.	9P	4N	2P	1D
5	KV	I	VARIABLE	DUM.ARG.	9P	5N	2P/2.	1D
	MAIN.		ENTRY					

TABLE OF EXTERNAL NAMES

BCOMON	EXTERNAL	8U	
CONANG	EXTERNAL	11U	
CONLIN	EXTERNAL	10U	
CONMAK	EXTERNAL	9U	
CONTIN	EXTERNAL	12U	
CONTUR	EXTERNAL	13U	

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM -	TO	ADDRESS -	LENGTH
10	I	4	7	360	26
10	J	5	7	526	6

BLOCK NAMES AND LENGTHS IN OCTAL

121-DISCOL 31 #IB 25 #CL 11610-//

STATIC SPACE (IN OCTAL)

B SAVE:	27
I SAVE:	2
CONSTANTS:	20
VARIABLES:	4
TEMPORARIES:	25
CODE	75
TOTAL:	177

```
1849
1850      1.            subroutine mycon (raray,iaray)
1851      2.            parameter (ndm1=72, ndm2=88, nr=5)
1852      3.            real raray(2),hue(nr)
1853      4.            integer iaray(9)
1854      5.            common /holder/ space(ndm1,ndm2),valmin,valrng,
1855           +          space2(ndm1,ndm2)
1856      6.            data hue /3hred,4hblue,5hgreen,6hyellow,5hwhite/
1857            c
1858            c  ***************************************************************
1859            c  *  This is the user-defined routine MYCON which sets up the line  *
1860            c  *  color for each contour line, using DISSPLA graphics calls.     *
1861            c  ***************************************************************
1862            c
1863      7.            do 10 i=1,nr
1864            c     set color according to level of contour value
1865      8.            if ((raray(i) .gt. valmin+float(nr-1)*valrng) .and.
1866                1         (raray(i) .lt. valmin+float(nr)*valrng))
1867                2         call setclr (hue(i))
1868      9.    10      continue
1869     10.            return
1870     11.            end
```

```
TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)
-- - -----------
  NUMBER USE   SOURCE PROGRAM REFERENCES

    10 UNDEF**      9I        7E
    10A23U          7I
    10BUNDEF**      7I
 00002 33b          8L
  (SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
  (FN-FORMAT NUMBER, UNDEF*-UNDEFINED STATEMENT NUMBER)
TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)
------------------- -----

ADDRESS NAME     TYPE MAIN USAGE  BLOCK        SOURCE PROGRAM REFERENCES

        FLOAT     R   INLINE FUNCT.                 8U/2
      3 HUE       R   1DIM ARRAY                    8P       6S       3D
     10 I         I   VARIABLE                      8U/3     7I
        MAIN.         ENTRY
     12 MYCON         ENTRY                         10/2
        NDM1      I   PARAMETER                     5P/2     2S
        NDM2      I   PARAMETER                     5P/2     2S
        NR        I   PARAMETER                     8P/2     7N       3P       2S
      1 RARAY     R   1DIM ARRAY   DUM.ARG.         8U/2     3D       1D
        SETCLR        EXTERNAL                      8U
  14300 VALMIN    R   VARIABLE     HOLDER           8U/2     5D
  14301 VALRNG    R   VARIABLE     HOLDER           8U/2     5D
TABLE OF EXTERNAL NAMES
----------------------- -----

        SETCLR        EXTERNAL                      8U
      ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)
      - ----------- -- -
      A  USED IN FORTRAN  ASSIGN STATEMENT     P  USED IN CALL/FUNC CALL OR ARRAY DEF
      D  DEFINED IN  DECLARATIVE STATEMENT     R  FORMAT  USED  IN A  READ  STATEMENT
      E  STATEMENT NUMBER ENDING A DO LOOP     S  STORED  SO CONTENTS  MAY BE CHANGED
      I  INDEX OF A DO  OP IMPLIED DO LOOP     U  NAME  USED IN EXECUTABLE  STATEMENT
      J  STATEMENT NUMBER USED IN TRANSFER     W  FORMAT  USED IN A  WRITE  STATEMENT
      L  SOURCE LINE OF A STATEMENT NUMBER     *  DEFINED  OR DECLARED  BUT NOT  USED
      N  NAME USED AS A  DO LOOP PARAMETER     ?  TEN OR  MORE REFERENCES  TO  SYMBOL
TABLE OF PARAMETERS ENCOUNTERED
---- ------------- - -- ----

NDM1      = 72                                                  NDM2      = 88
NR        = 5
```

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
10	1	7	9	23d	13

BLOCK NAMES AND LENGTHS IN OCTAL

41 MYCON	15 #IB	2 #CI	30602 HOLDER	

STATIC SPACE (IN OCTAL)

B SAVE	13
I SAVE	2
CONSTANTS	1
VARIABLES	10
TEMPORARIES	2
CODE	30
TOTAL	60

```
1871
1872      1.            subroutine adjust (tim,array,n,tmax,length)
1873         c----------------------------------------------------------------
1874         c     chops away portions of time history that are not of interest.
1875         c----------------------------------------------------------------
1876      2.            real tim(n), array(n)
1877         c
1878      3.            length=0
1879      4.            tmax=tim(n)
1880      5.            amax=array(ismax(n,array,1))
1881      6.            amin=array(1)+(amax-array(1))*0.1
1882         c
1883         c     chop to the right.
1884      7.            imax=n
1885      8.            do 10 i=n,1,-1
1886      9.            if (array(i).gt.amin) go to 20
1887     10.       10   continue
1888     11.            go to 30
1889     12.       20   imax=i
1890     13.       30   tmax=tim(imax)
1891         c
1892         c     chop to the left.
1893     14.            imin=1
1894     15.            do 40 i=1,imax
1895     16.            if (array(i).gt.amin) go to 50
1896     17.       40   continue
1897     18.            go to 60
1898     19.       50   imin=i
1899     20.       60   length=imax-imin+1
1900     21.            do 70 i=imin,imax
1901     22.            tim(i-imin+1)=tim(i)
1902     23.       70   array(i-imin+1)=array(i)
1903     24.            return
1904     25.          end
O   AT SEQUENCE NUMBER      22.
 PRNAME ADJUST   COMMENT -     DEPENDENCY INVOLVING ARRAY "TIM" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
O   AT SEQUENCE NUMBER      23.
 PRNAME ADJUST   COMMENT       DEPENDENCY INVOLVING ARRAY "ARRAY" ********** P=0250024C
 EXPLANATION:  AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 ADJUST              VECTOR LOOP BEGINS AT SEQ. NO.          21, P=     65d
 ADJUST  CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO.          21, P=     65d
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE	PROGRAM REFERENCES
10	UNDEF**	10L	8F
10A	42c	8L	
10B	47c	8L	
20	50a	12L	9J
30	51a	13L	11J
40	UNDEF**	17L	15E
40A	57b	15L	
40B	64b	15L	
50	64d	19L	16J
60	65d	20L	18J
70	UNDEF**	23L	21L
70A	105d	21L	
70B	133c	21L	
00002	116a	23L	
00003	133c	23E	
00004	UNDEF**	23F	
00005	124d	23F	

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES								
13	ADJUST		ENTRY		1D/2								
5	AMAX	R	VARIABLE		6U	5S							
6	AMIN	R	VARIABLE		16U	9U	6S						
2	ARRAY	R	1DIM ARRAY	DUM.ARG.	23U	23S	16U	9U	6U/2	5P	5U	2P	1D
10	I	I	VARIABLE		23U/2	22U/2	21I	19U	16U	15I	12U	9U	8I
7	IMAX	I	VARIABLE		21N	20U	15N	13U	12S	7S			
11	IMIN	I	VARIABLE		23U	22U	21N	20U	19S	14S			
	ISMAX	I	EXTERNAL		5U								
5	LENGTH	*I	VARIABLE	DUM.ARG.	20S	3S	1D						
	MAIN.		ENTRY										
3	N	I	VARIABLE	DUM.ARG.	8N	7U	5P	4U	3U	2P/2	1D		
1	TIM	R	1DIM ARRAY	DUM.ARG.	22U	22S	13U	4U	2D	1D			
4	TMAX	*R	VARIABLE	DUM.ARG.	13S	4S	1D						

TABLE OF EXTERNAL NAMES

ISMAX	I	EXTERNAL	5U

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM -	TO	ADDRESS -	LENGTH
10	I	8	10	42c	6
40	I	15	17	57b	5
70	I	21	23	105d	10

BLOCK NAMES AND LENGTHS IN OCTAL

136 ADJUST 40 #IB 4-#CI

STATIC SPACE (IN OCTAL)

B SAVE:	36
T SAVE:	2
CONSTANTS:	3
VARIABLES:	7
TEMPORARIES:	4
CODE:	124
TOTAL:	202

```
 1.          subroutine conint (arrin,xin,yin,nx,lx,ny,arrout,mx,my)
       c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
       c   this subroutine interplotates values from an irregularly spaced
       c   mesh to an evenly spaced mesh in two dimensions.
       c   it is meant to be an inteface to the NCAR contour plotter
       c   CONREC which works only on regularly spaced grids.
       c - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 2.          parameter (mxy=90)
 3.          real arrin(nx,ny), arrout(nx,my)
 4.          real xin(lx), yin(ny)
 5.          common /bound/ xmin, xmax, ymin, ymax
       c
 6.          integer iin(mxy)
 7.          real xfacm(mxy), xfacp(mxy)
       c
 8.          dy=(ymax-ymin)/float(my-1)
 9.          dx=(xmax-xmin)/float(mx-1)
10.          nym1=ny-1
11.          lxm1=lx-1
       c
       c   find the interval in the xin array in which x resides
       c       and find the scale factors for the x direction
       c
12.          imin=1
13.          do 30 iout=1,mx
14.            x=xmin+(iout-1)*dx
15.            if (iout.gt.1) imin=iin(iout-1)
16.            do 10 i=imin,lxm1
17.              if (x.gt.xin(i)) go to 10
18.                iin(iout)=max0(1,i-1)
19.                go to 20
20.    10      continue
21.            iin(iout)=lxm1
22.    20      in=iin(iout)
23.            xfacm(iout)=(x-xin(in))/(xin(in+1)-xin(in))
24.            xfacp(iout)=1.0-xfacm(iout)
25.    30    continue
       c
       c   cycle over the y values
       c
26.          jin=1
27.          do 70 jout=1,my
28.            y=ymin+(jout-1)*dy
       c
       c   find the interval in the yin array in which y resides
       c
29.            jmin=jin
30.            do 40 j=jmin,nym1
31.              if (y.gt.yin(j)) go to 40
32.                jin=max0(1,j-1)
33.                go to 50
34.    40      continue
35             jin=nym1
36.    50      continue
       c
       c   compute  scale factors for y direction
```

```
1961          c
1962     37.                yfacm (y-yin(jin))/(yin(jin+1)-yin(jin))
1963     38.                yfacp=1.0-yfacm
1964          c
1965          c   cycle over the x values
1966          c
1967     39.                do 60 iout=1,mx
1968     40.                   in=iin(iout)
1969     41.                   arrout(iout,jout)=yfacm*xfacm(iout)*arrin(in+1,jin+1)
1970                1             +yfacm*xfacp(iout)*arrin(in,jin+1)+yfacp*xfacm(iout)*arrin
1971                2             (in+1,jin)+yfacp*xfacp(iout)*arrin(in,jin)
1972     42.        60         continue
1973     43.        70       continue
1974          c
1975     44.                return
1976     45           end
CONINT                  VECTOR LOOP BEGINS AT SEQ. NO.          39, P=     552b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE	PROGRAM REFERENCES	
10	510b	20I	17J	16F
10A502d		16I		
10B513a		16I		
20	514c	22I	19J	
30	UNDEF**	25I	13E	
30A470b		13I		
30B525c		13I		
40	546c	34I	31J	30E
40A541b		30I		
40B551b		30I		
50	552b	36I	33J	
60	UNDEF**	42I	39E	
60A573d		39I		
60B636a		39I		
70	UNDEF**	43I	27E	
70A531a		27I		
70B640d		27I		
00002	476b	15I		
00003	606a	42E		
00004	605c	42E		
00005	616a	42E		
00006	615c	42E		
00007	622a	42E		
00008	621c	42E		
00009	626c	42E		
00010	626a	42E		

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES							
1	ARRIN	R	2DIM ARRAY	DUM.ARG.	41U/4	3D	1D					
7	ARROUT	*R	2DIM ARRAY	DUM.ARG.	41S	3P	1D					
442	CONINT		ENTRY		1D/2							
422	DX	R	VARIABLE		14U	9S						
421	DY	R	VARIABLE		28U	8S						
	FLOAT	R	INLINE FUNCT		9U	8U						
430	I	I	VARIABLE		18P	17U	16I					
3	IIN	I	1DIM ARRAY		40U	22U	21S	18S	15U	6D		
425	IMIN	I	VARIABLE		16N	15S	12S					
431	IN	I	VARIABLE		41U/4	40S	23U/3	22S				
426	IOUT	I	VARIABLE		41U/5	40U	39I	24U/2	23U	22U	21U	18U	15U/2
					14U	13I						
436	J	I	VARIABLE		32P	31U	30I					
432	JIN	I	VARIABLE		41U/4	37U/3	35S	32S	29U	26S		
435	JMIN	I	VARIABLE		30N	29S						

433	JOUT	I	VARIABLE		41U	28U	27I	
5	LX	I	VARIABLE	DUM.ARG.	11U	4P	1D	
424	LXM1	I	VARIABLE		21U	16N	11S	
	MAIN.		ENTRY					
	MAXO	I	INLINE FUNCT		32U	18U		
10	MX	I	VARIABLE	DUM.ARG.	39N	13N	9P	1D
	MXY	I	PARAMETER		7P/2	6P	2S	
11	MY	I	VARIABLE	DUM.ARG.	27N	8P	3P	1D
6	NY	I	VARIABLE	DUM.ARG.	10U	4P	3P	1D
423	NYM1	I	VARIABLE		35U	30N	10S	
427	X	R	VARIABLE		23U	17U	14S	
135	XFACM	R	1DIM ARRAY		41U/2	24U	23S	7D
267	XFACP	R	1DIM ARRAY		41U/2	24S	7D	
2	XIN	R	1DIM ARRAY	DUM.ARG.	23U/3	17U	4D	1D
1	XMAX	R	VARIABLE	BOUND	9U	5D		
0	XMIN	R	VARIABLE	BOUND	14U	9U	5D	
434	Y	R	VARIABLE		37U	31U	28S	
437	YFACM	R	VARIABLE		41U/2	38U	37S	
440	YFACP	R	VARIABLE		41U/2	38S		
3	YIN	R	1DIM ARRAY	DUM.ARG.	37U/3	31U	4P	1D
3	YMAX	R	VARIABLE	BOUND	8U	5D		
2	YMIN	R	VARIABLE	BOUND	28U	8U	5D	

TABLE OF EXTERNAL NAMES

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

MXY = 90

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
30	IOUT	16	25	470b	36
10	I	16	20	502d	11
70	JOUT	27	43	531a	110
40	J	30	34	541b	10
60	IOUT	39	42	573d	43

BLOCK NAMES AND LENGTHS IN OCTAL

644-CONINT 45 #IB 0 #CI 4 BOUND

STATIC SPACE (IN OCTAL)

B SAVE:	40
T SAVE:	5
CONSTANTS:	1
VARIABLES:	440
TEMPORARIES:	0
CODE:	203
TOTAL:	711

```
      subroutine wsize (xmin,xmax,ymin,ymax,xlo,xhi,ylo,yhi)
c-----------------------------------------------------------
c     sets plot window size.
c-----------------------------------------------------------
      data fmin /0.1/, fmax /0.9/, fav /0.5/
c
      delx=xmax-xmin
      dely=ymax-ymin
      if (dely.gt.delx) go to 10
      xlo=fmin
      xhi=fmax
      fy=(xhi-xlo)*dely/delx
      yhi=fav+0.5*fy
      ylo=yhi-fy
      return
   10 ylo=fmin
      yhi=fmax
      fx=(yhi-ylo)*delx/dely
      xhi=fav+0.5*fx
      xlo=xhi-fx
      return
      end
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER USE SOURCE PROGRAM REFERENCES

10 36d 12I 5U

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES				
6	DELX	R	VARIABLE		14U	8U	5U	3S	
7	DELY	R	VARIABLE		14U	8U	5U	4S	
5	FAV	R	VARIABLE		15U	9U	2S		
4	FMAX	R	VARIABLE		13U	7U	2S		
3	FMIN	R	VARIABLE		12U	6U	2S		
11	FX	R	VARIABLE		16U	15U	14S		
10	FY	R	VARIABLE		10U	9U	8S		
	MAIN.		ENTRY						
13	WSIZE		ENTRY		1D/2				
6	XHI	R	VARIABLE	DUM.ARG.	16U	15S	8U	7S	1D
5	XLO	R	VARIABLE	DUM.ARG.	16S	8U	6S	1D	
2	XMAX	R	VARIABLE	DUM.ARG.	3U	1D			
1	XMIN	R	VARIABLE	DUM.ARG.	3U	1D			
10	YHI	R	VARIABLE	DUM.ARG.	14U	13S	10U	9S	1D
7	YLO	R	VARIABLE	DUM.ARG.	14U	12S	10S	1D	
4	YMAX	R	VARIABLE	DUM.ARG.	4U	1D			
3	YMIN	R	VARIABLE	DUM.ARG.	4U	1D			

TABLE OF EXTERNAL NAMES

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

BLOCK NAMES AND LENGTHS IN OCTAL

52-WSIZE 14 #1B 0 #00

STATIC SPACE (IN OCTAL)

B SAVE:	14
T SAVE:	0
CONSTANTS:	1
VARIABLES:	11
TEMPORARIES:	0
CODE:	40
TOTAL:	66

```
  1.        subroutine eosp1 (rrr,eee,n,gamma,ppp)
      c------------------------------------------------------------
      c     equation of state routine for air.
      c
      c     input varible definitions.
      c          rrr = mass density
      c          eee = internal energy per unit volume
      c                (converted for internal use to energy per unit mass)
      c            n = number of entries in arrays rrr & eee
      c
      c     this routine calculates gamma and ppp as follows:
      c        gamma = 1.0 + p/(rho*e)
      c        ppp   = pressure(units of eee)
      c------------------------------------------------------------
  2.        parameter (m=92)
      c
  3.        dimension rrr(n), eee(n), gamma(n), ppp(n)
  4.        dimension t11(m), t12(m), t21(m), t22(m), rho(m), e(m), int(m)
  5.        dimension p(m), omp(m), q(m), i(m), j(m), tem(m)
  6.        dimension g(8,105), jcy(m), js(m)
  7.        dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
  8.        dimension g7(120), gf(840)
      c
  9.        equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
 10.        equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
 11.        equivalence (g7(1),g(1,91)), (int(1),tem(1)), (gf(1),g(1,1))
      c
 12.           data rzeroi /774.413/
 13.        data xl16e /2.7725887222397744835689081810414791107177734375/
 14.        data rl16e /0.360673760222240957773465197533369064331054687 5/
      c------------------------------------------------------------
      c     q = gamma - 1.0 is stored for 32 bit word machines in powers of
      c     16 across for mass density variation and intermediate values
      c     1 - 16 for powers of 16 vertically which represent the internal
      c     energy variation.
      c
      c     16**(2) .ge. rho .ge. 16**(-6)
      c     16**(8) .le. e   .le. 16**(15)
      c------------------------------------------------------------
 15.        data g1 /8*.4222,8*.4152,8*.4110,8*.4081,8*.4058,8*.4040,8*.4024,8
           1 *.4011,8*.3998,8*.3988,8*.3978,8*.3969,8*.3961,8*.3953,8*.3935/
 16.        data g2 /.3918,.3918,.3918,.3918,.3918,.3916,.3918,.3918,.3723,.37
           1 15,.3707,.3699,.3690,.3680,.3663,.3637,.3555,.3538,.3522,.3502,.3
           2 476,.3430,.3344,.3238,.3370,.3370,.3370,.3364,.3347,.3277,.3099,.
           3 2885,.3257,.3227,.3201,.3134,.3062,.3014,.2884,.2591,.3166,.3110,
           4 .3063,.2946,.2831,.2783,.2677,.2358,.3111,.3006,.2940,.2787,.2635
           5 ,.2588,.2502,.2236,.3075,.2906,.2810,.2665,.2466,.2418,.2350,.213
           6 1,.3043,.2819,.2695,.2554,.2317,.2269,.2216,.2038,.2929,.2740,.25
           7 93,.2455,.2206,.2136,.2097,.1955,.2840,.2672,.2500,.2366,.2166,.2
           8 015,.1988,.1879,.2764,.2611,.2429,.2285,.2125,.1890,.1890,.1611,.
           9 2714,.2555,.2384,.2210,.2079,.1818,.1799,.1747,.2669,.2504,.2343,
           $ .2141,.2037,.1822,.1709,.1689,.2624,.2473,.2304,.2096,.1998,.1828
           $ ,.1684,.1639/
 17.        data g3 /.2599,.2446,.2268,.2087,.1961,.1834,.1673,.1601,.2401,.21
           1 91,.1972,.1775,.1592,.1444,.1358,.1203,.2002,.1960,.1749,.1536,.1
           2 376,.1252,.1107,.1044,.1911,.1829,.1633,.1420,.1266,.1101,.1012,.
```

```
2055                3 0933, .1950, .1781, .1566, .1415, .1241, .1118, .1009, .0918, .2001, .1789,
2056                4 .1594, .1443, .1306, .1189, .1095, .1013, .2040, .1826, .1657, .1494, .1338
2057                5 , .1177, .1081, .0980, .2034, .1854, .1683, .1497, .1322, .1169, .1051, .094
2058                6 6, .1969, .1855, .1685, .1487, .1304, .1149, .1024, .0916, .1899, .1837, .16
2059                7 77, .1475, .1287, .1126, .1002, .0900, .1841, .1817, .1667, .1464, .1272, .1
2060                8 109, .0983, .0888, .1800, .1800, .1659, .1455, .1262, .1097, .0965, .0878, .
2061                9 1779, .1787, .1657, .1450, .1254, .1087, .0949, .0868, .1773, .1778, .1656,
2062                $ .1447, .1250, .1080, .0939, .0859, .1783, .1778, .1658, .1448, .1248, .1076
2063                $ , .0933, .0851/
2064          18.    data g4 /.1808, .1781, .1667, .1451, .1248, .1074, .0930, .0843, .2134, .20
2065                1 40, .1978, .1782, .1565, .1368, .1206, .1074, .2210, .2072, .1957, .1739, .1
2066                2 516, .1312, .1137, .1000, .2245, .2109, .1989, .1772, .1563, .1390, .1247, .
2067                3 1133, .2299, .2132, .2017, .1795, .1579, .1384, .1221, .1090, .2350, .2157,
2068                4 .2023, .1798, .1575, .1370, .1197, .1057, .2397, .2194, .2034, .1796, .1572
2069                5 , .1372, .1205, .1070, .2452, .2227, .2050, .1805, .1576, .1379, .1236, .111
2070                6 8, .2510, .2256, .2069, .1814, .1581, .1383, .1231, .1103, .2560, .2282, .20
2071                7 91, .1822, .1585, .1385, .1226, .1083, .2605, .2312, .2111, .1829, .1588, .1
2072                8 386, .1222, .1070, .2677, .2358, .2129, .1836, .1592, .1386, .1218, .1071, .
2073                9 2759, .2403, .2145, .1857, .1598, .1389, .1219, .1078, .2834, .2445, .2160,
2074                $ .1878, .1603, .1394, .1223, .1084, .2905, .2484, .2175, .1898, .1613, .1399
2075                $ , .1226, .1090/
2076          19.    data g5 /.2963, .2531, .2199, .1918, .1625, .1407, .1230, .1096, .4323, .35
2077                1 82, .3109, .2889, .2803, .2706, .2410, .2224, .4610, .4026, .3624, .3212, .2
2078                2 926, .2551, .2375, .2015, .4199, .3837, .3401, .2979, .2623, .2318, .2108, .
2079                3 1854, .3924, .3642, .3194, .2760, .2427, .2157, .1902, .1721, .3794, .3479,
2080                4 .3025, .2673, .2311, .2019, .1842, .1613, .3674, .3448, .2961, .2593, .2255
2081                5 , .1994, .1785, .1594, .3573, .3443, .2910, .2517, .2293, .2006, .1843, .167
2082                6 9, .3661, .3438, .2935, .2597, .2336, .2225, .2143, .2116, .3674, .3435, .30
2083                7 80, .2728, .2606, .2577, .2573, .2573, .3685, .3453, .3210, .3014, .2942, .2
2084                8 933, .2932, .2932, .3814, .3612, .3341, .3276, .3257, .3253, .3252, .3252, .
2085                9 3903, .3752, .3570, .3522, .3513, .3510, .3506, .3496, .4012, .3899, .3782,
2086                $ .3751, .3743, .3741, .3734, .3713, .4155, .4057, .3956, .3930, .3920, .3913
2087                $ , .3907, .3890/
2088          20.    data g6 /.4290, .4205, .4118, .4092, .4077, .4065, .4059, .4047, .5411, .53
2089                1 85, .5359, .5353, .5351, .5350, .5350, .5350, .5823, .5812, .5801, .5797, .5
2090                2 796, .5797, .5797, .5797, .6096, .6090, .6085, .6082, .6082, .6083, .6083, .
2091                3 6083, .6308, .6306, .6305, .6303, .6303, .6305, .6305, .6305, .6481, .6483,
2092                4 .6485, .6483, .6484, .6486, .6487, .6487, .6627, .6632, .6637, .6636, .6637
2093                5 , .6640, .6640, .6640, .6754, .6761, .6769, .6768, .6770, .6773, .6773, .677
2094                6 3, .6866, .6875, .6885, .6884, .6886, .6890, .6890, .6890, .6966, .6977, .69
2095                7 89, .6989, .6991, .6995, .6995, .6995, .7056, .7070, .7083, .7083, .7085, .7
2096                8 090, .7090, .7090, .7139, .7154, .7169, .7169, .7172, .7176, .7177, .7177, .
2097                9 7214, .7231, .7248, .7248, .7251, .7256, .7256, .7256, .7285, .7303, .7321,
2098                $ .7321, .7325, .7330, .7330, .7330, .7350, .7370, .7390, .7390, .7393, .7398
2099                $ , .7399, .7399/
2100          21.    data g7 / .7411, .7432, .7453, .7454, .7457, .7463, .7463, .7463, .8069, .81
2101                1 03, .8138, .8139, .8145, .8152, .8153, .8153, .8454, .8496, .8538, .8540, .8
2102                2 547, .8556, .8557, .8557, .8727, .8774, .8822, .8825, .8832, .8842, .8843, .
2103                3 8843, .8938, .8990, .9042, .9046, .9054, .9064, .9065, .9065, .9111, .9166,
2104                4 .9222, .9226, .9235, .9246, .9247, .9247, .9258, .9316, .9374, .9379, .9387
2105                5 , .9399, .9400, .9400, .9384, .9445, .9506, .9511, .9520, .9532, .9533, .953
2106                6 3, .9496, .9559, .9622, .9627, .9637, .9649, .9650, .9650, .9596, .9661, .97
2107                7 27, .9731, .9741, .9754, .9755, .9755, .9686, .9753, .9821, .9826, .9836, .9
2108                8 849, .9850, .9850, .9769, .9837, .9906, .9912, .9922, .9936, .9937, .9937, .
2109                9 9845, .9915, .9986, .9991,4* .9999, .9915, .9987,6* .9999, .9981,7* .9999/
2110          c
```

```
c     real air eos, table lookup on grimore data.  (no temp. model)
c     to avoid costly logrithmic functions the table "g" is stored in a
c     form so that the hexadecimal word structure of a 32 bit machine
c     may be exploited
c     this logic may be transfered to other machines by recalculating
c     the table "g" appropriate to the word architecture of that machine.
c     machine dependent functions and key numbers must also be changed.
c     ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
      22.           ist=0
      23.           nr=n
      24.    10     nst=min0(nr,m)
c
      25.           do 20 ire=1,nst
      26.           rho(ire)=rzero(ist+ire)
      27.           e(ire)=amax1(3.e8,eee(ist+ire)/rrr(ist+ire))
c
c     calculate mass density variation index "i".
      28.           tem(ire)=alog(rho(ire))*rl16e+500.0
      29.           i(ire)=tem(ire)
      30.           omp(ire)=tem(ire)-i(ire)
      31.           i(ire)=502-i(ire)
      32.           p(ire)=1.0-omp(ire)
      33.           i(ire)=max0(i(ire),1)
c
c     calculate internal energy variation index "j".
      34.           tem(ire)=alog(e(ire))*rl16e
      35.           jcy(ire)=ifix(tem(ire))
      36.           tem(ire)=tem(ire)-jcy(ire)
      37.           tem(ire)=exp(xl16e*tem(ire))
      38.           jcy(ire)=jcy(ire)-7
      39.           js(ire)=tem(ire)
      40.           q(ire)=tem(ire)-js(ire)
      41.           j(ire)=js(ire)+15*jcy(ire)
      42.           j(ire)=min0(j(ire),104)
      43.           j(ire)=i(ire)+8*j(ire)
      44.    20     i(ire)=j(ire)-8
      45.           do 30 ire=1,nst
      46.           t11(ire)=qf(i(ire))
      47.           t21(ire)=qf(i(ire)+1)
      48.           t12(ire)=qf(j(ire))
      49.    30     t22(ire)=qf(j(ire)+1)
c
c     calculate gamma by linear interpolation.
      50.           do 40 ire=1,nst
      51.           t12(ire)=t12(ire)-t11(ire)
      52.           t22(ire)=t22(ire)-t21(ire)
      53.    40     gamma(ist+ire)=1.0+omp(ire)*(t11(ire)+q(ire)*t12(ire))+p(ire)*
      54.       1   (t21(ire)+q(ire)*t22(ire))
c
      55.           nr=nr-nst
      56.           ist=ist+nst
                    if (nr.gt.0) go to 10
c
c     calculate pressure in units of eee
      57.           do 50 ire=1,n
      58.    50     ppp(ire)=amax1(0.0,eee(ire)*(gamma(ire)-1.0))
```

```
   2167          c
   2168    59.         return
   2169    60.       end
 FOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        25, P=    4143c
 FOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        50, P=    4267c
 FOSP1               VECTOR LOOP BEGINS AT SEQ. NO.        57, P=    4322d
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE PROGRAM REFERENCES	
10	4143c	56U	24I
20	UNDEF**	44I	25E
	20A4162b	25L	
	20B4252b	25I	
30	UNDEF**	49L	45E
	30A4257d	45I	
	30B4267c	45I	
40	UNDEF**	53I	50E
	40A4277a	50I	
	40B4322d	50I	
50	UNDEF**	58L	57E
	50A4340c	57I	
	50B4351b	57L	

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES								
	ALOG	R	EXTERNAL		34U	28U							
	AMAX1	R	INLINE FUNCT.		58U	27U							
2564	E	R	1DIM ARRAY		34P	27S	4D						
2	EEE	R	1DIM ARRAY	DUM ARG.	58P	27P	3P	1D					
4134	EOSP1		ENTRY		1D/2								
	EXP	R	EXTERNAL		37U								
4	G1	*R	1D EQ. ARRAY		9D	15S	7D						
174	G2	*R	1D EQ. ARRAY		9D	16S	7D						
364	G3	*R	1D EQ. ARRAY		9D	17S	7D						
554	G4	*R	1D EQ. ARRAY		10D	18S	7D						
744	G5	*R	1D EQ. ARRAY		10D	19S	7D						
1134	G6	*R	1D EQ. ARRAY		10D	20S	7D						
1324	G7	*R	1D EQ. ARRAY		11D	21S	8D						
4	GAMMA	R	1DIM ARRAY	DUM.ARG.	58U	53S	3P	1D					
4	GF	R	UNDEF EQUIV		49U	48U	47U	46U	11D	8D			
3344	I	I	1DIM ARRAY		47U	46U	44S	43U	33P	33S	31U	31S	30U
					29S	5D							
	IFIX	I	INLINE FUNCT.		35U								
4132	IRE	I	VARIABLE		58U/3	57I	53U/9	52U/3	51U/3	50I	49U/2	48U/2	47U/2
					46U/2	45I	44U/2	43U/3	42U/2	41U/3	40U/3	39U/2	38U/2
					37U/2	36U/3	35U/2	34U/2	33U/2	32U/2	31U/2	30U/3	29U/2
					28U/2	27U/3	26U/2	25I					
4127	IST	I	VARIABLE		55U	55S	53U	27U/2	26U	22S			
3500	J	I	1DIM ARRAY		49U	48U	44U	43U	43S	42P	42S	41S	5D
3634	JCY	I	1DIM ARRAY		41U	38U	38S	36U	35S	6D			
3770	JS	I	1DIM ARRAY		41U	40U	39S	6D					
	M	I	PARAMETER		24P	6P/2	5P/6	4P/7	2S				
	MAIN.		ENTRY										

	MAXO	I	INLINE FUNCT		33U							
	MINO	I	INLINE FUNCT.		42U	24U						
3	N	I	VARIABLE	DUM.ARG.	57N	23U	3P/4	1D				
4130	NR	I	VARIABLE		56U	54U	54S	24P	23S			
4131	NST	I	VARIABLE		55U	54U	50N	45N	25N	24S		
3054	OMP	R	1DIM ARRAY		53U	32U	30S	5D				
2720	P	R	1DIM ARRAY		53U	32S	5D					
5	PPP	*R	1DIM ARRAY	DUM.ARG.	58S	3P	1D					
3210	Q	R	1DIM ARRAY		53U/2	40S	5D					
2430	RUO	R	1DIM ARRAY		28P	26S	4D					
4126	RLIGE	R	VARIABLE		34U	28U	14S					
1	RRR	R	1DIM ARRAY	DUM.ARG.	27U	26U	3D	1D				
4124	RZERO1	R	VARIABLE		26U	12S						
1650	T11	R	1DIM ARRAY		53U	51U	46S	4D				
2004	T12	R	1DIM ARRAY		53U	51U	51S	48S	4D			
2140	T21	R	1DIM ARRAY		53U	52U	47S	4D				
2274	T22	R	1DIM ARRAY		53U	52U	52S	49S	4D			
1514	TEM	R	1D EQ. ARRAY		40U	39U	37U	37S	36U	36S	35P	34S	30U
					29U	28S	11D	5D				
4125	XLIGE	R	VARIABLE		37P	13S						

TABLE OF EXTERNAL NAMES

ALOG	R	EXTERNAL	34U	28U	
EXP	R	EXTERNAL	37U		

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

M = 92

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
20	IRE	25	44	4162b	70
30	IRE	45	49	4257d	10
40	IRE	50	53	4277a	23
50	IRE	57	58	4340c	11

BLOCK NAMES AND LENGTHS IN OCTAL

4354 FOSP1 240 #IB 3 #CL

STATIC SPACE (IN OCTAL)

B SAVE:	33
T SAVE:	5
CONSTANTS:	2
VARIABLES:	4131
TEMPORARIES:	203
CODE:	221
TOTAL:	4617

```
      1.           subroutine eosp2 (rrr,eee,n,gamma,ppp)
          c ---------------------------------------------------------
          c     equation of state for dust.
          c ---------------------------------------------------------
      2.           real rrr(n), eee(n), gamma(n), ppp(n)
      3            data gama,gamam1 /1.4,0.4/
          c
      4            do 10 i=1,n
      5.           gamma(i)=gama
      6.     10    ppp(i)=eee(i)*gamam1
      7.           return
      8            end
EOSP2          VECTOR LOOP BEGINS AT SEQ. NO.        4, P=       13b
```

TABLE OF STATEMENT NUMBERS (ALL ADDRESSES IN TABLES ARE IN OCTAL)

NUMBER	USE	SOURCE PROGRAM REFERENCES	
10	UNDEF**	6L	4L
10A27d		4L	
10B37a		4L	

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
(FN=FORMAT NUMBER, UNDEF*=UNDEFINED STATEMENT NUMBER)

TABLE OF NAMES ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS	NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES		
2	FEE	R	1DIM ARRAY	DUM.ARG.	6U	2P	1D
7	EOSP2		ENTRY		1D/2		
3	GAMA	R	VARIABLE		5U	3S	
4	GAMAM1	R	VARIABLE		6U	3S	
4	GAMMA	*R	1DIM ARRAY	DUM.ARG.	5S	2P	1D
5	I	I	VARIABLE		6U/2	5U	4L
	MAIN.		ENTRY				
3	N	I	VARIABLE	DUM.ARG.	4N	2P/4	1D
5	PPP	*R	1DIM ARRAY	DUM.ARG.	6S	2P	1D

TABLE OF EXTERNAL NAMES

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
10	I	4	6	27d	10

BLOCK NAMES AND LENGTHS IN OCTAL

42-EOSP2	27 #IB	0 #CL

STATIC SPACE (IN OCTAL)

B SAVE:	25
I SAVE:	2
CONSTANTS:	1
VARIABLES:	5
TEMPORARIES:	0
CODE:	34
TOTAL:	71

```
      c                   ,rscr(jk)
      c          go to (60,40,20), nspec
10.          species 3
11.    20       call eosp3 (rho,ein,np,gscr,pscr)
12.             do 30 jk=1,np
13.             scr3(jk)=rhb(jk)*scr1(jk)
14.             scr3(jk)=cvmgt(0.0,scr3(jk),scr3(jk).lt.0.1)
15.             pre(jk)=pre(jk)+pscr(jk)*scr3(jk)
16.             gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr3(jk)
       30       rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rhb(jk),scr3(jk).lt.0.1)
      c
      c
17.          species 2
18.    40       call eosp2 (rho,ein,np,gscr,pscr)
19.             do 50 jk=1,np
20.             scr2(jk)=rha(jk)*scr1(jk)
21.             scr2(jk)=cvmgt(0.0,scr2(jk),scr2(jk).lt.0.1)
22.             pre(jk)=pre(jk)+pscr(jk)*scr2(jk)
23.             gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr2(jk)
       50       rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rha(jk),scr2(jk).lt.0.1)
      c
      c
24.          species 1.
25.    60       call eosp1 (rho,ein,np,gscr,pscr)
26.             do 70 jk=1,np
27.             scr1(jk)=amax1(0.0,rscr(jk)*scr1(jk))
28.             pre(jk)=amax1(0.0,pre(jk)+pscr(jk)*scr1(jk))
29.    70       gam(jk)=gam(jk)+(gscr(jk)-1.0)*scr1(jk)
30.             return
              end
EOS
EOS
EOS
EOS
VECTOR LOOP BEGINS AT SEQ. NO.   4, P=
VECTOR LOOP BEGINS AT SEQ. NO.  11, P=   13b
VECTOR LOOP BEGINS AT SEQ. NO.  18, P=   52a
VECTOR LOOP BEGINS AT SEQ. NO.  25, P=  120d
                                         167c
```

```
 2 NSPEC    I   VARIABLE     
 6 PRE      R   1DIM ARRAY   DUM.ARG.
10 PSCR     R   1DIM ARRAY   DUM.ARG.   21U
15 RHA      R   1DIM ARRAY   DUM.ARG.   23U    19U
16 RHB      R   1DIM ARRAY   DUM.ARG.   16U    12U    3P
 3 RHO      R   1DIM ARRAY   DUM.ARG.   24P    17P    10P    5U     2U
 7 RSCR     R   1DIM ARRAY   DUM.ARG.   26P    23P/2  23S    16P/2  16S   8U
12 SCR1     R   1DIM ARRAY   DUM.ARG.   28U    27U    26U    26S    19U   12U   8S   3P
13 SCR2     R   1DIM ARRAY   DUM.ARG.   23P    22U    21U    20P/2  20S   19S   3P   1D
14 SCR3     R   1DIM ARRAY   DUM.ARG.   16P    15U    14U    13P/2  13S   12S   3P   1D
```

TABLE OF EXTERNAL NAMES

EOSP1	EXTERNAL	24U
EOSP2	EXTERNAL	17U
EOSP3	EXTERNAL	10U

ABBREVIATIONS USED ABOVE (THESE ARE KEYED TO THE SOURCE LISTING LINE NUMBER)

A	USED IN FORTRAN ASSIGN STATEMENT	P	USED IN CALL/FUNC CALL OR ARRAY DEF
D	DEFINED IN DECLARATIVE STATEMENT	R	FORMAT USED IN A READ STATEMENT
E	STATEMENT NUMBER ENDING A DO LOOP	S	STORED SO CONTENTS MAY BE CHANGED
I	INDEX OF A DO OR IMPLIED DO LOOP	U	NAME USED IN EXECUTABLE STATEMENT
J	STATEMENT NUMBER USED IN TRANSFER	W	FORMAT USED IN A WRITE STATEMENT
L	SOURCE LINE OF A STATEMENT NUMBER	*	DEFINED OR DECLARED BUT NOT USED
N	NAME USED AS A DO LOOP PARAMETER	?	TEN OR MORE REFERENCES TO SYMBOL

TABLE OF LOOPS ENCOUNTERED

LABEL	INDEX	FROM	TO	ADDRESS	LENGTH
10	JK	4	8	30d	13
30	JK	11	16	74c	24
50	JK	18	23	143b	24
70	JK	25	28	207d	22

BLOCK NAMES AND LENGTHS IN OCTAL

234 EOS 60 #TB 22 #CL

STATIC SPACE (IN OCTAL)

B SAVE:	57
T SAVE:	1
CONSTANTS:	2
VARIABLES:	3
TEMPORARIES:	22
CODE:	227
TOTAL:	336

INITIAL PAGES OF PROGRAM UNITS

ADJUST	88
COLCON	79
CONINT	91
CONV3D	13
DISCOL	82
DISCON	43
DISPRF	37
DISTPL	40
EOS	109
FOSP1	99
FOSP2	106
FILE10	61
FLIK3D	1
MYCON	85
PLTARR	65
PROF3D	46
RESTRI	72
WSIZE	96

*** COMPILATION SUMMARY ***

EXECUTE LINE - t-nplot3d on=adix
CPU SECONDS - 3.1893
I/O SECONDS - 1.0630
SYS SECONDS - 0.0472
SIZE(OCTAL) - 601000

END

DATE

FILMED

DTIC

JULY 88