

MICROCOPY RESOLUTION TEST CHART
BUREAU OF STANDARDS-1963-A

AD-A193 152

DTIC FILE COPY

4

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987



Science Applications International Corporation

DTIC
ELECTE
MAR 24 1988
S D

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

00 2 05 008

4

NUMERICAL MODELING OF AIRBLAST

1ST YEAR FINAL REPORT

SAIC 87/1701

JUNE 1987



Science Applications International Corporation

DTIC
ELECTE
S MAR 24 1988 D

D CP

DISTRIBUTION STATEMENT F
Approved for public release
Distribution Unlimited

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**



Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By <i>per lti</i>	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
<i>A-1</i>	

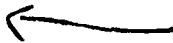
SCIENCE APPLICATIONS INTERNATIONAL CORPORATION

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

-A-

TABLE OF CONTENTS:

Introduction	1
Task 2.1	2
Task 2.2	3
Task 2.4	3
Task 2.5	4
Task 2.6	5
Task 2.7	5
Task 2.8	5
Task 2.9	5
Appendix A: Source listing of the 2-D Axisymmetric and Cartesian Nuclear and Non-Nuclear Modelling code;	
Appendix B: Abstract for DNA Dust Symposium;	
Appendix C: Chemical Explosive Charge Study using 3-D FCT code; and	
Appendix D: Compilation Listing of FLIK 3D Graphics code	



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 axisymmetric 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 nuclear 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 possibility of using distributed chemical charges in array geometries. Numerous chemical energy release models were 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**


```

1      program sp2r2d
2
3      a 2d fct code for high explosive and nuclear blasts in air.
4      coded for cartesian (x-y) and cylindrical (r-z)
5      coordinate systems.
6
7      parameter (mxx=150, mny=200, mxy=202)
8      parameter (mxxs=1, mnyy=1)
9      parameter (nsta=1)
10     parameter (nop=1)
11     parameter (mxx=1)
12
13     integer alpha
14     logical lmagrid, lstat, lnuc, thermal, lgrav, leos
15     real rha(mxx,mny), rhb(mxx,mny), rhc(mxx,mny)
16     real tem(mxx,mny), scf(mxx,mny), rho(mxx,mny)
17     real rvx(mxx,mny), rvy(mxx,mny), erg(mxx,mny)
18     common nx, ny, npx, nyp, nspec, idump, idlag, lmagrid
19     common lstat, lnuc, thermal, lgrav, leos
20     common alpha, gamma, gm1, dt, dx, dy, rhomin
21     common tem, scf, rho, rvx, rvy, erg, rha, rhb, rhc
22
23     real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
24     common /grids/ xcor, xcoro, ycor, ycoro, unit
25
26     real xband(5), dyr(4), yband(3), dyr(2)
27     common /grdcon/ ndx, nsmth, mdy, alt, nalted, xfine, hx, hy,
28     xband, yband, dxr, dyr, xleft
29
30     real rhoa(mxy), pres(mxy), vel(mxy), uh(mxy)
31     real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
32     real rho0(mxy), rh00(mxy), rhco(mxy), vh(mxy)
33     real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
34     real rhoH(mxy), rvxH(mxy), rvyH(mxy), ergh(mxy)
35     real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
36     real rmin(mxy), rmax(mxy), rvr(mxy)
37     common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
38     rho0, rh00, rhco, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
39     . ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
40
41     logical part
42     real xp(nop), yp(nop)
43     common /part/ part, xp, yp, nopp
44
45     real rfs(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta), tme(mxx)
46     real prs(mxx,nsta), xs(nsta), ysnsta)
47     common /stat/ rfs, vls, gms, prs, tme, xs, ys, frach, eblast,
48     txx
49
50     common /tstep/ dtmin, dtmax
51     common /jw/ rocj, pcj, dcj, eocj, gcj, acj, baos, ccj, ricj,
52     r2cj, wcj
53
54     real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
55     parameter (jtime=1)
56     real fsky(mxy), rhtl(mxx,1tme)
57     common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
58     . jtherm
59
60     logical lscan, right, left, ltop, botm
61     common /scan/ lscan, right, left, ltop, botm
62     common /active/ lxr, jxr, lxl, jxl, lyl, jyl, lrb, jrb, ltr,
63     jtr, ladd, jadd, factor
64

```

```

65 integer itape, otape, ifile, ofile
66 logical rezone, lrest, lsave, lgrid, dust, ldump, lmgold
67      , dustn
68 character *8 prntf, restf, dumpf, readf
69 character *25 path
70 character *40 glab
71 character *40 label
72 common /inout/ itape, otape, ifile, ofile, lsave, npath, nlab
73 c
74 parameter (ntdump=57)
75 real dmpim(ntdump)
76 logical ltdump
77 data ldump /.true./
78 c
79 data readf /'megdat'/, prntf /'megprnt'/, dumpf /'sp2r'/
80 data minstp, maxstp /1,11/
81 data itape, otape, ifile, ofile /14,15,0,1/
82 c
83 data ndx, nsmth, mdy /75,25,37/
84 data hx, hy /1.0,1.0/
85 data xfine, alt, xleft /0.0,0.0,0.0/
86 data naxed /30/
87 data lxx /0/
88 c
89 data dtmin, dtmax, dtstrt /1.0e-6,2.0e-5,1.e-5/
90 data jtherm /0/, nbod /0/
91 c
92 namelist /control/ minstp,maxstp,ldiag,ldump,lmgrid,leas,lmuc
93 .itape,otape,ifile,ofile,lstat,part,therm,lgrav,lrest,lsave
94 .ltdump,dmptim
95 namelist /mesh/ nx,ny,alpha,dx,dy,ndx,nsmth,mdy,xfine,alt
96 .naxed,hx,hy,ladd,jadd,factor,lscan,rigt,left,ltop,botm,lgrid
97 .xleft
98 namelist /band/ bcr,bcl,bct,ccb
99 namelist /physdat/ gamma,rhomin,dtmin,dtmax,nspec,jtherm,jdirt
100 .dstrt,dust,nbod,dustn
101 namelist /jwlist/ r0cj,pcj,dcj,e0cj,gcj,acj,beos,ccj,r1cj,r2cj
102 .wcj
103 namelist /stats/ xs,ys
104 namelist /newgrd/ lxr,lxl,jyt,jyb
105 c
106 c initialize the code control parameters.
107 idlag=10000
108 ldump=10000
109 lmgrid=.false.
110 lstat=.false.
111 part=.false.
112 leos=.false.
113 lmuc=.false.
114 lgrav=.false.
115 therm=.false.
116 rezone=.false.
117 lscan=.false.
118 rigt=.false.
119 left=.false.
120 ltop=.false.
121 botm=.false.
122 lsave=.false.
123 lgrid=.false.
124 dust=.false.
125 dustn=.false.
126 ltdump=.false.
127 c
128 c initialize grid parameters.

```

```

129 nx=mnx
130 ny=many
131 nxp=mx+1
132 nyp=my+1
133 alpha=1
134 dx=1.0
135 dy=1.0
136 kx=mnx
137 kx=1
138 ixl=1
139 jxl=1
140 iyt=1
141 jyt=many
142 iyb=1
143 jyb=1
144 ilr=mnx
145 jbr=many
146 ladd=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 gml=gamma-1.0
159 nspec=1
160 C
161 time=0.0
162 tme(1)=time
163 radi=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 continue
178 rpath=0
179 do 30 i=1,25
180 if (path(i:i).eq.' ') go to 40
181 rpath=rpath+i
182 continue
183 C
184 read (5,control)
185 if (.not.lrest) then
186 do 50 i=1,1.-1
187 prntf(i:i)=dumpf(i:i)
188 if (prntf(i:i).ne.' ') and (prntf(i:i).eq.' ') prntf(i:i)=i
189 }=0
190 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,control)
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.thermal) jtherm=0
207      if (.not.lstat) go to 70
208      read (5, stats)
209      write (6, stats)
210      do 60 ksta=1, nsta
211      ys(ksta)=ys(ksta)+alt
212      if (ldump.le.mxx) go to 70
213      write (6,220) ldump,mxx
214      go to 160
215      continue
216      do 80 k=1, mxy
217      unit(k)=1.0
218      c
219      c
220      c
221      c
222      c
223      c
224      c
225      c
226      c
227      c
228      c
229      c
230      c
231      c
232      c
233      c
234      c
235      c
236      c
237      c
238      c
239      c
240      c
241      c
242      c
243      c
244      c
245      c
246      c
247      c
248      c
249      c
250      c
251      c
252      c
253      c
254      c
255      c
256      c

      determine bluff body profiles.
      if (nbod.ne.0) call bluff (nbod)

      initialize variables to ambient conditions.
      if (.not.lrest) then
      call initial (bcl,bcr,bcb,bct)
      if (dust) call dust0
      if (dustn) call dust01
      if (thermal) call tlayer (rho,rvx,rvy,erg,rhomin,ensum)
      if (lnuc) call dposit (time)
      call burst (xcor,ycor,erg)
      call dposit (time)
      else
      call jump (gamma,leos,er00(2))
      call dpshok (rho,rvx,rvy,erg)
      endif
      call dumper (0,time,dumpf,label,path,glab)
      call diagno (lstep,time)
      go to 160
      endif

      restart if lrest=.true.
      lmgrid=lmgrid
      call restrt (minstp,time,restf,glab)
      lmgrid=lmgrid
      if (.not.lnuc) call jump (gamma,leos,er00(2))

      determine dump time number.
      ltdump=1
      if (ltdump) then
      do 90 ktcdump=ntcdump,1,-1
      if (time.ge.dmp1m(ktcdump)) go to 100
      continue
      go to 110
      ltdump=ktcdump+1
      do 100 ltdump=1tdump,le.ntcdump
      continue
      go to 110

```

```

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          lir=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('ix, ratios',ipe12.5,' rad loss fblast')
271      if (lnuc) call fblast (time,dt,rad,eblast,ratio)
272      if (lscan) call xyg
273      lactiv=min0(ladd,jadd)
274 c
275 c      begin the loop over time steps.
276      do 150 lstep=minstp,maxstp
277          kstep=lstep-minstp
278 c
279 c      determine active grid boundaries.
280      if (lscan) then
281          if (mod(kstep,lactiv).eq.0) call xygrid (arg,scrh)
282      endif
283 c
284 c      the time and the timestep are updated.
285      call dtset (lstep,time,radi,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      continue
293 c
294 c      integrate along rows and columns.
295      call bndrx (bcr,bcl)
296      call integx
297      call bndry (bct,ccb)
298      call integy
299      go to 140
300 c
301 c      integration for shock on bluff bodies.
302      call blufxy (bcl,bcr,ccb,bct)
303      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,lstep,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(lstep,time)
316      if (dustn) call dustm(lstep,time)
317 c
318 c      dump at the appropriate step.
319      if (ldump) then
320          ldump=(lstep.eq.maxstp).or.(time.ge.dmp1imf(tchump))

```

```

321 else
322   ldump=(lstep.eq.maxstp).or.(ldump.ne.0.and.mod(lstep,ldump).eq
323   .0)
324   endif
325   ldump=cvmg(.false.,ldump,lstep.eq.minstp)
326   if (ldump) then
327     call flush
328     call dumper (lstep,time,dumpf,label,path,glab)
329     call flush
330     ixx=0
331     if (ldump) then
332       if (time.ge.dmpim(ntdump)) go to 160
333       ltdump=ltdump+1
334       ltdump=ltdump.le.ntdump
335     endif
336   endif
337 c
338 c perform all required diagnostics
339 c if (ldiag.ne.0.and.mod(lstep,ldiag).eq.0) call diagno (lstep
340 c ,time)
341 c if (time.gt.25.) go to 170
342 c continue
343 c
344 c save print file.
345 c write (6,210) ptime
346 c if (.not.isave) go to 170
347 c write (glab,230) path(:npath),prntf
348 c ierr=1000
349 c call mass ('save',\waiton\,0,glab,ierr)
350 c write (6,240) prntf,ierr
351 c
352 c close files.
353 c close (unit=5,status='keep')
354 c close (unit=6,status='keep')
355 c stop
356 c
357 c 180 format (a)
358 c 190 format ('1fast2d hydrodynamics ',5x,a)
359 c 200 format (15x,a)
360 c 210 format ('Optree = ',e13.5)
361 c 220 format (1x,'ldump,mxx=',215/1x,'ldump should be .le. mxx')
362 c 230 format ('6',a,a,'6')
363 c 240 format (1x,'ierr for ',a,' =',14/)
364 c end
365 c
366 c-----
367 c-----
368 c-----
369 c parameter (mxx=150, mny=200, mxy=202)
370 c real erg(mnx,mny), scri(mxy), scr2(mxy)
371 c
372 c real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
373 c parameter (jthm=1)
374 c real fsky(mxy), rhtl(mnx,jthm)
375 c common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
376 c , jthm
377 c
378 c logical lscan, right, ltop, left, botm
379 c common /scan/ lscan, right, left, ltop, botm
380 c common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, iir,
381 c jbt, ladd, jadd, factor
382 c
383 c save old grid boundaries.
384 c ixro=ixr

```

```

385 ixlo=ixl
386 jyto=jyt
387 jybo=jyb
388 c
389 c right.
390 if (.not.right) go to 30
391 ixr=i
392 do 20 j=jmin,jmax
393 do 10 i=iimin,imax
394 scri(i)=cvmgt(float(i),0.0,erg(i,j),ge.scr2(j))
395 klim=imax((imax-jmin+1),scri(jmin,i))+jmin-1
396 ixr=max0(klim,ixr)
397 jxr=cvmgt(j,jxr,ixr.eq.klim)
398 continue
399 ixr=min0((ixr+1add),mx)
400 imax=ixr
401 right=cvmgt(.false...true...ixr.eq.mnx)
402 c
403 c top.
404 if (.not.top) go to 60
405 jyt=i
406 do 50 j=jmin,imax
407 do 40 i=iimin,imax
408 scri(j)=cvmgt(float(j),0.0,erg(i,j),ge.scr2(j))
409 klim=imax((jmax-jmin+1),scri(jmin,i))+jmin-1
410 jyt=max0(klim,jyt)
411 jyt=cvmgt(i,jyt,jyt.eq.klim)
412 continue
413 jyt=min0((jyt+1add),mny)
414 jmax=jyt
415 jtop=cvmgt(.false...true...jyt.eq.mny)
416 c
417 c left.
418 if (.not.left) go to 90
419 ixl=mx
420 do 80 j=jmin,jmax
421 do 70 i=iimin,imax
422 scri(j)=cvmgt(float(i),float(mnx),erg(i,j),ge.scr2(j))
423 klim=ismin((imax-jmin+1),scri(jmin,i))+jmin-1
424 ixl=cvmgt(min0(klim,ixl),ixl, klim.ne.imin)
425 jxl=cvmgt(j,jxl,ixl.eq.klim)
426 continue
427 ixl=max0((ixl-1add),1)
428 imin=ixl
429 left=cvmgt(.false...true...ixl.eq.1)
430 c
431 c bottom.
432 if (.not.botm) go to 120
433 jyb=ny
434 do 110 i=iimin,imax
435 do 100 j=jmin,jmax
436 scri(j)=cvmgt(float(j),float(mny),erg(i,j),ge.scr2(j))
437 klim=ismin((jmax-jmin+1),scri(jmin,i))+jmin-1
438 jyb=cvmgt(min0(klim,jyb),jyb, klim.ne.jmin)
439 iyb=cvmgt(i,iyb,jyb.eq.klim)
440 continue
441 iyb=max0((iyb-1add),1)
442 jmin=max0((jyb-jtherm+1)
443 botm=cvmgt(.false...true...jyb.eq.1)
444 ixl=cvmgt(imax-0((ixr,ixro),ixr,ixr.ne.mnx)
445 ixl=cvmgt(min0(ixl,ixlo),ixl,ixlo.ne.1)
446 jyt=cvmgt(imax-0((jyt,jyto),jyt,jyto.ne.mny)
447 jyb=cvmgt(min0((jyb,jybo),jyb,jybo.ne.1)
448 iir=ixr-ixl+1

```

```

449 jbt=jyt-jyb+1
450 iscan=cvmgt(.false...true...ixl.eq.1.and.ixr.eq.mnx.and.jyb.eq.
451 1.and.jyt.eq.mny)
452 return
453 c
454 entry xyg
455 imin=ixl
456 imax=ixr
457 jmin=mxo(jyb,jtherm+1)
458 jmax=jyt
459 do 130 j=1,mny
460 scr2(j)=factor*er00(j+1)
461 return
462 end
463
464 c-----
465 c integrates along rows.
466 c-----
467 parameter (mnx=150, mny=200, mxy=202)
468 parameter (mxxx=1, myyy=1)
469 integer alpha
470 logical lgrid, lstat, lnuc, ltherm, lgrav, leos
471 real rha(mnx,mny), rhb(mnxx,mny), rhc(mnxx,mny)
472 real tem(mnxx,mny), scl(mnxx,mny), rho(mnx,mny)
473 real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
474 common nx, ny, npx, nyp, nspec, ldump, ldiag, lgrid
475 common lstat, lnuc, ltherm, lgrav, leos
476 common alpha, gamma, gai, dt, dx, dy, rhomin
477 common tem, scl, 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), srvk(mxy), srvy(mxy), serg(mxy)
484 real rho0(mxy), rho(mxy), rhc0(mxy), vhf(mxy)
485 real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
486 real rhoh(mxy), rvah(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, srvk, srvy, serg,
490 rho0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvah, rvyh
491 , 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,srvk), (pscr,srvy), (gscr,serg)
495 equivalence (scr1,scrh), (scr2,rmin)
496 real gams(mxy), rha(mxy), rhoh(mxy), rhch(mxy), rgrv(mxy)
497 equivalence (gams,sgam), (rgrv,dely)
498 equivalence (rha,rmax), (rhch,rvr), (rhch,dely)
499 c
500 real pr00(mxy), rho0(mxy), er00(mxy), gp00(mxy), gravity(mxy)
501 parameter (jthe=1)
502 real fsky(mxy), rhtl(mnx,jthe)
503 common /grav/ pr00, rho0, er00, gp00, gravity, fsky, rhtl
504 , jtherm
505 c
506 common /active/ ixr, jxr, ixl, jxl, iyt, jyt, lyb, jyb, ilr.
507 jbt, iadd, jadd, factor
508 c
509 common /lbc/ rho1bc, rvx1bc, rvy1bc, erg1bc
510 common /rbc/ rho1bc, rvx1bc, rvy1bc, erg1bc
511 common /bbc/ rho1bc, rvx1bc, rvy1bc, erg1bc
512 common /lbc/ rho1bc, rvx1bc, rvy1bc, erg1bc

```



```

513 c      call ngridd (xcor0(ixl),llr+1,alpha)
514 c      call ogridu (llr+1)
515 c      call ngridd (xcor(ixl),llr+1,alpha)
516 c
517 c      loop over rows to be integrated.
518 c
519 c      do 250 j=lyb,jyt
520 c      rho=cvmgt(rhx0(j+1),rhor,abcr,eq.1.0)
521 c      rho1=cvmgt(rhx0(j+1),rho,abcl,eq.1.0)
522 c      erg=cvmgt(erg0(j+1),erg,abcr,eq.1.0)
523 c      erg1=cvmgt(erg0(j+1),erg1,abcl,eq.1.0)
524 c      rhcr=rhor
525 c      rhcl=rhol
526 c      do 10 i=ixl,ixr
527 c      rho0(i)=rho(i,j)
528 c      rho1(i)=rho1(i,j)
529 c      rho0(i)=0.0
530 c      rho1(i)=0.0
531 c      rvx0(i)=rvx(i,j)
532 c      rvy0(i)=rvy(i,j)
533 c      erg0(i)=erg(i,j)
534 c      rhoh(i)=rho0(i)
535 c      rvsh(i)=rvx0(i)
536 c      rvyh(i)=rvy0(i)
537 c      erg1(i)=erg0(i)
538 c
539 c      go to (60,40,20), nspec
540 c      do 30 i=ixl,ixr
541 c      rhb0(i)=rhc(i,j)
542 c      rhoh(i)=rho0(i)
543 c      rhc0(i)=rhc0(i)-rhc0(i)
544 c      rhcr=rhcr-rhcr
545 c      rhcl=rhcl-rhcl
546 c      do 50 i=ixl,ixr
547 c      rho0(i)=rho(i,j)
548 c      rho1(i)=rho1(i)
549 c      rhc0(i)=amax1(rhc0(i)-rho0(i),0.0)
550 c      rhch(i)=rhc0(i)
551 c      rhcr=rhcr-rhcr
552 c      rhcl=rhcl-rhcl
553 c      continue
554 c
555 c      define boundary velocities by averaging values in interior and
556 c      exterior boundary cells.
557 c      rvxlef=rvx1+rvx(ixl,j)+rvx1
558 c      rho1ef=rho1+rho(ixl,j)+rho1
559 c      uhl(ixl)=rvx(ixl,j)/rho(ixl,j)
560 c      uhl(ixl)=0.5*(uhl(ixl)+rvx1ef/rho1ef)
561 c
562 c      rvxrig=rvx1+rvx(ixr,j)+rvx1
563 c      rho1rig=rho1+rho(ixr,j)+rho1
564 c      uhr(ixr)=rvx(ixr,j)/rho(ixr,j)
565 c      uhr(ixr)=0.5*(uhr(ixr+1)+rvx1rig/rho1rig)
566 c
567 c      advance total density, energy and x and y momentum for
568 c      half and then whole step.
569 c      do 190 lpass=1,2
570 c      dsub=dt/float(3-lpass)
571 c      calculate transport velocities.
572 c      do 70 i=ixl,ixr
573 c      rhom(i)=1.0/rho0(i)
574 c      val(i)=rvsh(i)+rhom(i)
575 c      ergk(i)=amax1(0.0,erg1(i)-0.5*(rvmh(i)+rvyh(i)+rvyh(i)
576 c

```

```

577   rvyh(i)=rhh(i)*gp00(j+1)
578   do 80 i=1,ixl,ixr
579     wh(i)=0.5*(vel(i)+vel(i-1))
580     call veloc(wh(ixl),(i+1),dtsub,ilr)
581 c
582 c   calculate pressure.
583   if (leus) then
584     call eos(ilr,nispec,rho(i),ergk(i),gms(i),pres(i),pres(i),
585 c   1   ,scr(ixl),paci(ixl),gscr(ixl),scr(ixl),scr2(ixl),rhah(ixl)
586 c   2   ,rho(ixl))
587     call eospl(rho(i),ergk(i),ilr,gms(i),pres(i))
588   else
589     do 90 i=1,ixr
590       pres(i)=gmi*ergk(i)
591     endif
592 c
593 c   calculate source terms.
594   do 100 i=1,ixr
595     srvi(i)=-pres(i)
596     sergi(i)=srvi(i)*vel(i)
597     srvi=-pres(i)
598     srvi=-pres(i)
599     sergi=-pres(i)*uh(i)
600     sergr=-pres(i)*uh(i+1)
601 c
602 c   advance total density, energy and x and y momentum.
603   if (abs(abcl).eq.1.0) call setbd(i,1)
604   call fryfct(rho0(i),rho(i),ilr,frhor,rhor,frho1,rho1)
605   do 110 i=1,ixr
606     rho(i)=aex(rho(i),rho1)
607     call sourced(ilr,dtsub,2,unf(i),srvi(i),srvi(i),srvi(i))
608     call fryfct(rv0(i),rvh(i),ilr,frvvr,rvr,frv1,rv1)
609     call fryfct(rv0(i),rvh(i),ilr,frvvr,rvr,frv1,rv1)
610     call sourced(ilr,dtsub,1,unf(i),sergi(i),sergi(i),sergi(i))
611     call fryfct(erg0(i),erg(i),ilr,fergr,ergr,fergi,erg(i))
612 c
613 c   advance species.
614   do 120 i=1,ixr
615     scr1(i)=1.e-8*rho(i)
616     scr2(i)=0.0
617     go to (190,150,130), nispec
618     call fryfct(rho0(i),rho(i),ilr,frhbr,rhbr,frhb1,rhb1)
619     do 140 i=1,ixr
620       rho(i)=cvngt(0.0,rho(i),rho1(i),le.scr1(i))
621       scr2(i)=scr2(i)+rho(i)
622     call fryfct(rho0(i),rho(i),ilr,frhar,rhar,frha1,rha1)
623     do 160 i=1,ixr
624       rho(i)=cvngt(0.0,rho(i),rho1(i),le.scr1(i))
625       scr2(i)=scr2(i)+rho(i)
626     call fryfct(rho0(i),rho(i),ilr,frhcr,rhcr,frhc1,rhc1)
627     do 170 i=1,ixr
628       rho(i)=cvngt(0.0,rho(i),rho1(i),le.scr1(i))
629       scr2(i)=scr2(i)+rho(i)
630     do 180 i=1,ixr
631       scrh(i)=rho(i)/scr2(i)
632     continue
633 c
634   do 200 i=1,ixr
635     rho(i,j)=rho(i)
636     rvx(i,j)=rvx(i)
637     rvy(i,j)=rvy(i)
638     erg(i,j)=erg(i)
639     go to (250,230,210), nispec
640   do 220 i=1,ixr

```

```

641      rhh(1,j)=rhh(1)+scrh(1)
642      do 240 i=1,ixr
643      rha(1,j)=rha(1)+scrh(1)
644 c
645 c      loop back for the next row to be integrated.
646      continue
647      return
648 c
649 c      entry intgy
650 c
651 c      integrates along columns.
652 c
653      call ngridd (ycor(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=1,ixr
659      do 260 j=jyb,jyt
660      rho0(j)=rho(1,j)
661      rhc0(j)=rho(1,j)
662      rho0(j)=0.0
663      rhc0(j)=0.0
664      rvx0(j)=rvx(1,j)
665      rvy0(j)=rvy(1,j)
666      erg0(j)=erg(1,j)
667      rhh(j)=rhh0(j)
668      rvxh(j)=rvx0(j)
669      rvyh(j)=rvy0(j)
670      erg1(j)=erg0(j)
671 c
672      go to (310,290,270), nspec
673      do 280 j=jyb,jyt
674      rho0(j)=rho(1,j)
675      rhh(j)=rhh0(j)
676      rhc0(j)=rhc0(j)-rhh0(j)
677      rhct=rhct-rhh0
678      rhcb=rhcb-rhh0
679      do 300 j=jyb,jyt
680      rho0(j)=rho(1,j)
681      rhh(j)=rhh0(j)
682      rhc0(j)=amax1((rhc0(j)-rho0(j)),0.0)
683      rhch(j)=rhc0(j)
684      rhct=rhct-rhat
685      rhcb=rhcb-rhab
686      continue
687 c
688 c      set thermal layer densities.
689      do 312 j=1,axy
690      rgrv(j)=rho0(j)
691      if (thermal) then
692      do 314 j=1,jthe
693      rgrv(j+1)=rht(1,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(1,jyb)+rvyb
699      rhobot=frhocb+rhc(1,jyb)+rhob
700      vhl(jyb)=rvy(1,jyb)/rho(1,jyb)
701      vhl(jyb)=0.5*(vhl(jyb)+rvybot/rhobot)
702 c
703      rvytop=frvyt+rvy(1,jyt)+rvyt
704      rhotop=frhott+rhc(1,jyt)+rhot

```

```

705     vhl(jyt+1)=rvyl(jyt)/rho(jyt)
706     vhl(jyt+1)=0.5*(vhl(jyt+1)+rvyl(jyt)/rho(jyt))
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     rhoa(j)=1.0/rho(j)
716     vel(j)=rvyl(j)*rhoa(j)
717     ergk(j)=easx(0.0,ergk(j)-0.5*rhoa(j)*(rvxh(j)+rvxh(j)+rvyh(j)
320     +rvyh(j))-rhoa(j)*gp00(j+1))
718     +rvyh(j))-rhoa(j)*gp00(j+1))
719     do 330 j=jyb+1,jyt
720     vhl(j)=0.5*(vel(j)+vel(j-1))
721     call velocd (vhl(jyb),(jbt+1),dtsub,jbt)
722 c
723 c calculate pressure.
724     if (ileas) then
725     call eos (jbt,nspec,rhoa(jyb),ergk(jyb),gams(jyb),pres(jyb)
726     ,rsqr(jyb),pscr(jyb),gscr(jyb),scr1(jyb),scr2(jyb),rhoa(jyb)
727     ,rhoa(jyb))
728     call eospl (rhoa(jyb),ergk(jyb),jbt,gams(jyb),pres(jyb))
729     else
730     do 340 j=jyb,jyt
731     pres(j)=gam*ergk(j)
732     endif
733 c
734 c calculate source terms.
735     if (lgrav) then
736     do 350 j=jyb,jyt
737     srvt(j)=pr00(j+1)-pres(j)
738     sgrv(j)=-(rgrv(j+1)-rhoa(j))*gravity(j+1)
739     srvtb=pr00(jyb)-pres(jyb)
740     srvt-pr00(jyt+1)-pres(jyt)
741     sgrv(jyt+1)=-(rgrv(jyt+2)-rhoa(jyt))*gravity(jyt+2)
742     srvt(j)-pr00(j+1)-pres(j)-fsky(j)
743     sgrv(j)-rhoa(j)*gravity(j+1)
744     srvtb-pr00(jyb)-pres(jyb)-fsky(jyb)
745     srvt-pr00(jyt+1)-pres(jyt)-fsky(jyt+1)
746     sgrv(jyt+1)-rhoa(jyt)*gravity(jyt+2)
747     else
748     do 355 j=jyb,jyt
749     srvt(j)=pres(j)
750     srvtb=pres(jyb)
751     srvt-pr00(jyt)
752     endif
753     do 370 j=jyb,jyt
754     serg(j)=pres(j)*vel(j)
755     sergt=pres(jyt)*vhl(jyt+1)
756     sergb=pres(jyb)*vhl(jyb)
757     serg(j)=-(pres(j)+fsky(j+1)-pr00(j+1))*vel(j)
758     sergt=-(pres(jyt)+fsky(jyt+2)-pr00(jyt+2))*vhl(jyt+1)
759     sergb=-(pres(jyb)+fsky(jyb)-pr00(jyb))*vhl(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 (rhoa(jyb),rhoa(jyb),jbt,frhot,rhot,frhob,rhob)
764     do 380 j=jyb,jyt
765     rhoa(j)=easx(rhoa(j),rhoa(j))
766     call sourced (jbt,dtsub,2,unitt(jyb),srvt(jyb),srvtb,srvtb)
767     if (lgrav) then
768     call sourced (jbt,dtsub,3,unitt(jyb),sgrv(jyb),sgrv(jyt+1),0.0)

```

```

769      endif
770      call fryct (rvx0(jyb),rvyh(jyb),jbt,frvxt,rvyt,frvyb,rvyb)
771      call fryct (rvx0(jyb),rvyh(jyb),jbt,frvxt,rvyt,frvyb,rvyb)
772      call sourced (jbt,dtsub,1,unit(jyb),sergljyb),sergt,sergb)
773      call fryct (erg0(jyb),ergh(jyb),jbt,fergt,ergt,fergb,ergb)
774 C
775 C   advance species.
776      do 390 j=jyb,jyt
777         scri(j)=1.e-8*rhoh(j)
778         scr2(j)=0.0
779      go to (460,420,400), nspec
780      call fryct (rhc0(jyb),rhch(jyb),jbt,frhct,rhct,frhcb,rhcb)
781      do 410 j=jyb,jyt
782         rhch(j)=cvmgt(0.0,rhch(j),rhch(j)),le.scri(j)
783         scr2(j)=scr2(j)+rhch(j)
784      call fryct (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.scri(j)
787         scr2(j)=scr2(j)+rhah(j)
788      call fryct (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.scri(j)
791         scr2(j)=scr2(j)+rhch(j)
792      do 450 j=jyb,jyt
793         scrh(j)=rhoh(j)/scr2(j)
794         continue
795 C
796 C   replace variables in 2D arrays.
797      do 470 j=jyb,jyt
798         rho(i,j)=rhoh(j)
799         rvx(i,j)=rvxh(j)
800         rvy(i,j)=rvyh(j)
801         erg(i,j)=ergh(j)
802      go to (520,500,480), nspec
803      do 490 j=jyb,jyt
804         rhb(i,j)=rhbh(j)+scrh(j)
805         do 510 j=jyb,jyt
806            rha(i,j)=rhah(j)+scrh(j)
807 C
808 C   loop back for the next col to be integrated.
809         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     bc = 1   subsonic (ambient).
818 C     bc = -1  reflective.
819 C     bc = 2   prescribed.
820 C
821 C   check if active grid includes boundaries.
822     abcl=cvmgt(bcl,0.0,ixl,eq.1)
823     abcr=cvmgt(bcr,0.0,ixr,eq.mnx)
824 C
825 C   density
826     frhor=cvmgt(0.0,1.0,abcr,gn.1.0)
827     rhor=cvmgt(rhor,bc,0.0,ahcr,eq.2.0)
828     frhol=cvmgt(0.0,1.0,abcl,gn.1.0)
829     rhol=cvmgt(rhol,bc,0.0,ahcl,eq.2.0)
830 C   species density.
831     frhar=1.0
832     rhar=0.0

```

```

833 frhal=1.0
834 rhal=0.0
835 frhbr=1.0
836 rhbr=0.0
837 frhbl=1.0
838 rhbl=0.0
839 frhcr=frhor
840 rhcr=rhor
841 frhcl=frhol
842 rhcl=rhol
843 C
844 C
845 C
846 C
847 C
848 C
849 C
850 C
851 C
852 C
853 C
854 C
855 C
856 C
857 C
858 C
859 C
860 C
861 C
862 C
863 C
864 C
865 C
866 C
867 C
868 C
869 C
870 C
871 C
872 C
873 C
874 C
875 C
876 C
877 C
878 C
879 C
880 C
881 C
882 C
883 C
884 C
885 C
886 C
887 C
888 C
889 C
890 C
891 C
892 C
893 C
894 C
895 C
896 C

x - component of momentum.
frvxt=cvmgt(abcr,amax1(0.0,1.0-abcr),abcr,eq.-1.0)
rvxt=cvmgt(rvxtbc,0.0,abcr,eq.2.0)
frvxb=cvmgt(abcl,amax1(0.0,1.0-abcl),abcl,eq.-1.0)
rvxb=cvmgt(rvxtbc,0.0,abcl,eq.2.0)

y - component of momentum.
frvyt=cvmgt(1.0,amax1(0.0,1.0-abcr),abcr,eq.-1.0)
rvyt=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)

energy.
fergr=cvmgt(0.0,1.0,abcr,ge.1.0)
ergr=cvmgt(ergbc,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)

entry bndry(bct,ccb)
-----
sets up y boundary conditions for next call to Integy.
-----
check if active grid includes boundaries.
abcb=cvmgt(1.0,0.0,abcb,eq.1.0)
abct=cvmgt(bct,0.0,jyb,eq.1)

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)

species density.
frhat=1.0
rhat=0.0
frhab=1.0
rhah=0.0
frhbt=1.0
rhbt=0.0
frhbb=1.0
rhbb=0.0
frhct=frhot
rhct=rhot
frhcb=frhob
rhcb=rhob

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(rvxtbc,0.0,abcb,eq.2.0)

```

```

897 C      y - component of momentum.
898 C      frvy=cvmgt(abct,amaxt(0.0,1.0-abct),abct,eq.-1.0)
899      rvy=cvmgt(rvytbc,0.0,abct,eq.2.0)
900      frvyb=cvmgt(abcb,amaxt(0.0,1.0-abcb),abcb,eq.-1.0)
901      rvyb=cvmgt(rvybbc,0.0,abcb,eq.2.0)
902 C
903 C      energy.
904 C      fergt=cvmgt(0.0,1.0,abct,ge.1.0)
905      ergt=cvmgt(erg00(jyt+2),0.0,abct,eq.1.0)
906      ergt=cvmgt(ergtbc,ergt,abct,eq.2.0)
907      fergb=cvmgt(0.0,1.0,abcb,ge.1.0)
908      ergb=cvmgt(erg00(jyb),0.0,abcb,eq.1.0)
909      ergb=cvmgt(ergbcb,ergb,abcb,eq.2.0)
910      return
911
912      subroutine diagno (nstep,time)
913
914 C-----
915 C      performs general diagnostics.
916 C-----
917      parameter (mx=150, mny=200, mxy=202)
918      parameter (mxxx=1, mnyy=1)
919      parameter (mxx=1)
920      parameter (nop=1, nsta=1)
921      integer otape
922      integer alpha
923      logical lgrid, lstat, lnucl, lgrav, ltherm, lgrav, leos
924      real rha(mnx,mny), rhb(mnxx,mny), rhc(mnxx,mny)
925      real tem(mnxx,mny), scl(mnxx,mny), rho(mnx,mny)
926      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
927      common nx, ny, nxp, nyp, nspec, ldump, ldiag, lgrid
928      common lstat, lnucl, ltherm, lgrav, leos
929      common alpha, gamma, gmi, dt, dx, dy, rhomin
930      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
931 C
932      real pre(mnx,mny)
933      equivalence (pre,rvx)
934      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
935      common /grids/ xcor, xcoro, ycor, ycoro, unit
936 C
937      real xbnd(5), dxr(4), ybnd(3), dyr(2)
938      common /grdcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
939      xbnd, ybnd, dxr, dyr, xleft
940 C
941      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
942      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
943      real rho0(mxy), rho(mxy), rho0(mxy), vlt(mxy)
944      real rho0(mxy), rho0(mxy), rho0(mxy), erg0(mxy)
945      real rhoh(mxy), rvzh(mxy), rvyh(mxy), ergh(mxy)
946      real delx(mxy), dely(mxy), scrh(mxy), sgam(mxy)
947      real rain(mxy), rmax(mxy), rvr(mxy)
948      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
949      rho0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvzh, rvyh
950      , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
951 C
952      real rscr(mxy), pscr(mxy), gscr(mxy), scr1(mxy), scr2(mxy)
953      equivalence (rscr,vel), (pscr,rvr), (gscr,uh), (scr1,vh)
954      equivalence (scr2,scrh)
955 C
956      logical part
957      real xp(nop), yp(nop)
958      common /part/ part, xp, yp, nopp
959 C
960      real rhg(mxx,nsta), vls(mxx,nsta), qms(mxx,nsta), lme(mxx)

```

```

961 real prs(mxx,nsta), xs(nsta), ys(nsta), vs(nsta)
962 common /stat/ rns, vis, gms, prs, tme, xs, ys, fracn, eblast,
963 lxx
964 c
965 real pr00(mxy), rho0(mxy), er00(mxy), gp00(mxy), gravity(mxy)
966 parameter (jthm=1)
967 real fsky(mxy), rhtl(mnx,jthm)
968 common /grav/ pr00, rho0, er00, gp00, gravity, fsky, rhtl
969 , jthrm
970 c
971 logical lscan, rigr, left, ltop, botm
972 common /scan/ lscan, rigr, left, ltop, botm
973 common /active/ ixr, jxr, lxi, jxi, lyl, jyl, iyb, jyb, ilr,
974 jbt, iadd, jadd, factor
975 c
976 c compute the various conservation sums for the system.
977 call ngridd (xcor,nxp,alpha)
978 call ogridd (nxp)
979 call ngridd (xcor,nxp,alpha)
980 do 30 j=1,ny
981 call consrd (rho(1,j),nx,rho0(j))
982 call consrd (rvx(1,j),nx,rvx0(j))
983 call consrd (rvy(1,j),nx,rvy0(j))
984 call consrd (erg(1,j),nx,erg0(j))
985 go to (30,20,10), nspec
986 call consrd (rhb(1,j),nx,rhb0(j))
987 call consrd (rha(1,j),nx,rha0(j))
988 continue
989 call ngridd (ycor,nyp,1)
990 call ogridd (nyp)
991 call ngridd (ycor,nyp,1)
992 call consrd (rho0.ny,rhosum)
993 call consrd (rvx0.ny,rvxsum)
994 call consrd (rvy0.ny,rvysum)
995 call consrd (erg0.ny,ergsum)
996 go to (60,50,40), nspec
997 call consrd (rhb0.ny,rhbsum)
998 call consrd (rha0.ny,rhasum)
999 continue
1000 write (6,210) nstep,time,rhasum,rhbsum,rhosum,rvxsum,rvysum
1001 ,ergsum
1002 return
1003 c
1004 entry pressg
1005 c -----
1006 c calculates pressure in the entire grid and stores it in rvx,
1007 c which is stored temporarily on disk.
1008 c -----
1009 call scrdmp (rvx,1)
1010 do 90 j=1,mny
1011 do 70 i=1,mnx
1012 ergk(i)=amax1(0.0,erg(1,j)-0.5*(rvx(1,j)*rvx(1,j)+rvy(1,j)*rvy
1013 (1,j))/rho(1,j)-rho(1,j)*gp00(j+1))
1014 if (leos) then
1015 c call eos (mnx,nspec,rho(1,j),ergk,sgam,pre(1,j),rscr,pscr,gscr
1016 c ,scr1,scr2,rha(1,j),rhb(1,j))
1017 call eospl (rho(1,j),ergk,mnx,sgam,pre(1,j))
1018 else
1019 do 80 i=1,mnx
1020 pre(1,j)=gm1*ergk(i)
1021 endif
1022 continue
1023 return
1024 c

```



```

1025 entry datain(itape)
1026 read (itape) ix,ly,nspec,lmgrid,lstat,lnuc,lgrav,leos,part
1027 .therm)
1028 if (lx.eq.nx.and.ly.eq.ny) go to 100
1029 write (6,220) itape,ix,ly,nx,ny
1030 stop
1031 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) ((vis(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(i),i=1,nsta)
1039 read (itape) ((ys(i),i=1,nsta)
1040 read (itape) ((tme(i),i=1,mxx)
1041 read (itape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
1042 read (itape) lscan,rigt,left,ltop,botm,ixr,jxr,ixl,jxl,lyt,jyt
1043 .lyb,jyb)
1044 go to (140,130,120), nspec
1045 read (itape) ((rhh(i,j),i=1,nx),j=1,ny)
1046 read (itape) ((rha(i,j),i=1,nx),j=1,ny)
1047 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) (gravity(i),i=1,mxy)
1059 read (itape) (fsky(i),i=1,mxy)
1060 if (therm) read (itape) jtherm,((rhtl(i,j),i=1,mxx),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 continue
1067 read (itape) (xcoro(i),i=1,nxp),(ycoro(j),j=1,nyp)
1068 read (itape) (xbrd(i),i=1,5),(ybrd(j),j=1,3)
1069 read (itape) (dxr(i),i=1,4),(dyr(j),j=1,2)
1070 return
1071 c
1072 entry dataout(otape)
1073 write (otape) nx,ny,nspec,lmgrid,lstat,lnuc,lgrav,leos,part
1074 .therm)
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) ((vis(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(i),i=1,nsta)
1082 write (otape) ((ys(i),i=1,nsta)
1083 write (otape) ((tme(i),i=1,mxx)
1084 write (otape) (xcor(i),i=1,nxp),(ycor(j),j=1,nyp)
1085 write (otape) lscan,rigt,left,ltop,botm,ixr,jxr,ixl,jxl,lyt,jyt
1086 .jyt,lyb,jyb)
1087 go to (190,180,170), nspec
1088 write (otape) ((rhh(i,j),i=1,nx),j=1,ny)

```

```

1089 write (otape) ((rha(1,j),i=1,nx),j=1,ny)
1090 continue
1091 write (otape) ((rho(1,j),i=1,nx),j=1,ny)
1092 write (otape) ((pre(1,j),i=1,nx),j=1,ny)
1093 call scrdmp (rvx,2)
1094 write (otape) ((rvx(1,j),i=1,nx),j=1,ny)
1095 write (otape) ((rvy(1,j),i=1,nx),j=1,ny)
1096 write (otape) ((erg(1,j),i=1,nx),j=1,ny)
1097 if (lnuc) write (otape) radl,fracn,eblast
1098 write (otape) (p00(1),i=1,mxy)
1099 write (otape) (rho0(1),i=1,mxy)
1100 write (otape) (ei00(1),i=1,mxy)
1101 write (otape) (gp00(1),i=1,mxy)
1102 write (otape) (gravy(1),i=1,mxy)
1103 write (otape) (fsky(1),i=1,mxy)
1104 if (thermal) write (otape) jtherm,((rht(1,j),i=1,mnx),j=1,jthm
1105 )
1106 write (otape) dt
1107 if (.not.part) go to 200
1108 write (otape) nopp
1109 write (otape) (xp(1),i=1,nopp)
1110 write (otape) (yp(1),i=1,nopp)
1111 continue
1112 write (otape) (xcoro(1),i=1,nxp),(ycoro(j),j=1,nyp)
1113 write (otape) (xbnd(1),i=1,5),(yband(j),j=1,3)
1114 write (otape) (dxx(1),i=1,4),(dyr(j),j=1,2)
1115 return
1116 c
1117 210 format ('Oat step ',i4,' and time ',1pe12.4,' the conservation s
1118 ums are ...',/,,' sum of rha sum of rib sum of rho ',/,,' sum
1119 2 of rvx sum of rvy sum of erg ',/b14.6)
1120 220 format ('Odatain input error on itape =',i3,' the grid ',size o
1121 iff tape is ',i3,' by ',i3,' but fast2d ',expects ',i3,' by ',i3
1122 2)
1123 end
1124
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), 1012
1131 open (unit=10,file='store',status='new')
1132 write (10) ((arr(1,j),i=1,mnx),j=1,mny)
1133 return
1134 rewind (10)
1135 read (10) ((arr(1,j),i=1,mnx),j=1,mny)
1136 close (unit=10,status='delete')
1137 return
1138 end
1139
1140 c
1141 real rho0(na), rhan(na)
1142 c
1143 c
1144 c
1145 c frycct (rho0, rhan, na, rbc, rhoabc, lbc, rho1bc)
1146 c originator - fry and book code 4040, nr1 June 1982
1147 c does multicoefficient fct with a switch to choose either
1148 c of the form drho/dt = -div (rho*v) + sources in either
1149 c cartesian, cylindrical, or spherical geometry. the finite-differ-
1150 c ence grid can be eulerian, or spherical rezone, or lagrangian and can
1151 c be arbitrarily spaced. the algorithm used is a low-phase error fct
1152 c algorithm, vectorized and optimized for speed. a complete descrip-

```

tion appears in nrl memo report 3237, "flux-corrected transport
modules for solving generalized continuity equations".

arguments: in this routine the right boundary radh(np) is half a
cell beyond the last grid point n at rnc(n) and the left boundary
radh(l) is half a cell before the first grid point at rnc(1).
rho0 real array(na) grid point densities at start of step
rho1 real array(na) grid point densities at end of step
na integer number of cells in the system
rbc real right boundary condition factor
rhorbc real right guard cell value of rho
lbc real left boundary condition factor
rho1bc real left guard cell value of rho

language and limitations: the subroutine jpbfct is a multiple-
entry fortran routine in single precision (32 bits asc). the asc
parameter statement is used to set symbolically the internal array
dimensions. underflows are possible when the function being trans-
ported has many zeroes. the calculations generally misconserv by
one or two bits per cycle. the relative phase and amplitude errors
(for smooth functions) are typically several percent for charac-
teristic lengths of 1 - 2 cells (wavelengths of order 10 cells).
shocks are generally accurate to better than 1 percent. this sub-
routine must be compiled with the y option to force storage and
retention of internal variables. alternatively a common block can
be added to accomplish the same end.

entry points: jhgfct, dhfct, dthfct, ogridd, ngridd, velocd, sourced, and
consrd. the detailed documentation (or the listing below) gives
the explanation and use of the arguments to these other entries.

no auxiliary or library routines are called by jpbfct.

.....

parameter (npt=202)
logical lsource, lmom, logic(2)
real rbc, lbc, mask1, mask2
real mask3
real scrh(npt), rhot(npt), diff(npt), source(npt)
real flxh(npt), rulh(npt), mulh(npt), adugth(npt)
real epsb(npt), fsgn(npt), fabs(npt), lnrho(npt)
real temp(npt), term(npt), psgn(npt), lorhot(npt)
real int(npt), flxm(npt), flxp(npt), deltap(npt)
real rho(npt), lhr(npt), rh(npt), rln(npt), adwidth(npt)
real vtdodr(npt), gamma(npt)

common block /fctcom/ contains grid, geometry and cell information
to be communicated between subroutines, hydrodynamic utilities,
and the main program.

roh real array(na+1) old interface locations set by ogridd
rnh real array(na+1) new interface locations set by ngridd
real roh(npt), rnh(npt), roc(npt), rrc(npt)
real ah(npt), lo(npt), ln(npt), rln(npt)
common /fctcom/ roh, rnh, roc, rrc, ah, lo, ln, rln

real e(na), r(na)
real uhr(npa)
integer ind(nsh)
integer alpha
real radh(npa)
real c(na), a(na)
real rho(na)

1153 cd
1154 cd
1155 cd
1156 cd
1157 cd
1158 cd
1159 cd
1160 cd
1161 cd
1162 cd
1163 cd
1164 cd
1165 cd
1166 cd
1167 cd
1168 cd
1169 cd
1170 cd
1171 cd
1172 cd
1173 cd
1174 cd
1175 cd
1176 cd
1177 cd
1178 cd
1179 cd
1180 cd
1181 cd
1182 cd
1183 cd
1184 cd
1185 cd
1186 cd
1187 c
1188
1189 c
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200 c
1201 c
1202 c
1203 c
1204 cd
1205 cd
1206
1207
1208
1209 c
1210
1211
1212
1213
1214
1215
1216

```

1217 c      data fctone, roh, source, lsource, lmom /1.0.npt*1.,npt*0.,
1218      .false.,.false./
1219      data pi /3.14159265358979323846264338327950288419716939937511/
1220      data fpl /4.188790204786386084379046224057674407958984375/
1221      data c1, c2, c3, c4 /0.1666666666666666666666666666666666666666
1222      666666666666667, 0.33333333333333333333333333333333333333333333333
1223      333333333333333, 0.13333333333333333333333333333333333333333333333
1224      333333333333333, 0.03333333333333333333333333333333333333333333333
1225      333333333333333/
1226      equivalence (logic(1),lsource), (logic(2),lmom)
1227      equivalence (lnrho(1),lorhot(1))
1228      equivalence (flxh(1),scrh(1)), (term(1),source(1))
1229      equivalence (fabs(1),scrh(1)), (terp(1),source(1))
1230      equivalence (psgn(1),int(1))
1231
1232 c      lmom=.false.
1233      go to 10
1234
1235 c
1236 c
1237 c
1238 c
1239 c      entry jhgfct(rhoo,rhon,na,rbc,rhorbc,ibc,rholbc,e,r)
1240 cd
1241 cd
1242 cd      jhgfct (rhoo, rhon, na, rbc, rhorbc, ibc, 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      arguments: this entry uses the same arguments as jpbfcf with the
1248 cd      addition of arguments 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
1255      calculate the diffusive and convective fluxes.
1256      lmom=.true.
1257      n=na
1258      np=n+1
1259      do 20 i=2,n
1260      flxh(i)=0.5*aduth(i)*(rhoo(i)+rhoo(i-1))
1261      diff(i)=(rhoo(i)-rhoo(i-1))
1262      rhol=rhoo(i)*ibc+rholbc
1263      rhor=rhoo(n)*rbcr+rhorbc
1264      diff(n)=(rhoo(n)-rhoo(1))
1265      flxh(np)=(rhor-rhoo(n))
1266      flxh(np)=0.5*aduth(np)*(rhoo(np)+rhoo(n))
1267 c
1268 c      calculate lambdaao*rhot, the transported mass elements.
1269      do 30 i=1,n
1270      lorhot(i)=1a(i)*rhoo(i)-(flxh(i+1)-flxh(i))
1271 c
1272 c      add in the source terms as appropriate.
1273      do 40 i=1,n
1274      lorhot(i)=lorhot(i)+source(i)
1275      continue
1276      go to 90
1277 c
1278 c
1279 c
1280 c

```

```

1281 c      entry d1bfct(rhoo,rhon,na,rbc,rhorbc,IBC,rho1bc)
1282 c
1283 cd
1284 cd
1285 cd
1286 cd      description: this entry performs nonconservative convection. the
1287 cd      equation       $d(\rho h)/dt = v \cdot \text{grad}(\rho h) + \text{source}$  is solved.
1288 cd
1289 cd      arguments:  this routine uses the same arguments as jpbfct.
1290 cd
1291 cd
1292 c
1293 c      calculate the diffusive and convective fluxes.
1294      lmom=.false.
1295      n=na
1296      np=n+1
1297      do 60 i=2,n
1298          diff(i)=rhoo(i)-rho0(i-1)
1299          flux(i)=vdtodr(i)*diff(i)
1300          diff(i)=nullh(i)*diff(i)
1301          rho1=rhoo(i)+lbc+rho1bc
1302          rhor=rhoo(n)+rbc+rhorbc
1303          diff(np)=nullh(np)*(rho0(i)-rho1)
1304          diff(np)=nullh(np)*(rhor-rhoo(n))
1305          flux(np)=vdtodr(np)*(rho0(i)-rho1)
1306          flux(np)=vdtodr(np)*(rhor-rhoo(n))
1307 c
1308 c      calculate lambdao*rhot. the transported mass elements.
1309      do 70 i=1,n
1310          lorhot(i)=ln(i)*(rho0(i)-0.5*(flux(i+1)+flux(i)))
1311 c
1312 c      add in the source terms as appropriate.
1313      if (.not.lsourc) go to 90
1314      do 80 i=1,n
1315          lorhot(i)=lorhot(i)+source(i)
1316 c
1317 c      calculate the phenetical antidiffusive fluxes here.
1318      continue
1319      do 100 i=1,n
1320          lnrhot(i)=lorhot(i)*gamma(i+1)*diff(i+1)-gamma(i)*diff(i)
1321      do 110 i=1,n
1322          rhot(i)=lorhot(i)*rln(i)
1323          rhotr=rbc*rhot(n)+rhorbc
1324          rho1=lbc*rhot(i)+rho1bc
1325 c
1326      do 120 i=2,n
1327          flux(i)=nullh(i)*(rhot(i)-rhot(i-1))
1328          flux(i)=nullh(i)*(rhot(i)-rhot1)
1329          flux(np)=nullh(np)*(rhotr-rhot(n))
1330 c
1331 c      diffuse the solution rhot using old fluxes.
1332      do 130 i=1,n
1333          lnrhot(i)=lorhot(i)*nullh(i+1)*diff(i+1)-nullh(i)*diff(i)
1334 c
1335 c      calculate the transported/diffused density and grid differences.
1336      do 140 i=1,n
1337          rhot(i)=lnrhot(i)*rln(i)
1338          rhotr=rbc*rhot(n)+rhorbc
1339          rho1=lbc*rhot(i)+rho1bc
1340      do 150 i=2,n
1341          diff(i)=rhot(i)-rhot(i-1)
1342          diff(i)=rhot(i)-rhot1
1343          diff(np)=rhotr-rhot(n)
1344 c

```

```

1345 c calculate the sign and magnitude of the antidiffusive flux.
1346 do 160 i=1,np
1347 fsgn(i)=sign(ftime,diff(i))
1348 do 170 i=1,np
1349 fabs(i)=abs(flxp(i))
1350 c
1351 c calculate the flux-limiting based on the pressure for the momentum
1352 if (.not.imeom) go to 280
1353 do 180 i=1,n
1354 int(i)=e(i)-0.5*rhot(i)*rhot(i)/r(i)
1355 do 190 i=2,n
1356 deltap(i)=int(i)-int(i-1)
1357 deltap(i)=int(i)-lbc*int(i)
1358 deltap(np)=rbc*int(n)-int(n)
1359 do 200 i=1,n
1360 flxp(i+1)=-rhot(i)*(1.-sqrt(amax1(0.,1.-2.*r(i)*deltap(i+1)/
1361 (rhot(i)*rhot(i))))))
1362 flxm(i)=rhot(i)*(1.-sqrt(amax1(0.,1.+2.*r(i)*deltap(i)/(rhot
1363 (i)*rhot(i))))))
1364 continue
1365 do 210 i=1,np
1366 deltap(i)=nuh(i)*deltap(i)
1367 do 220 i=1,n
1368 int(i)=(int(i)+ln(i)+deltap(i+1)-deltap(i))*rln(i)
1369 do 230 i=2,n
1370 deltap(i)=rhot(i)*(int(i)-int(i-1))
1371 deltap(i)=rhot(i)*(lbc*int(i)-int(i))
1372 deltap(np)=rhot(n)*(int(n)-rbc*int(n))
1373 do 240 i=1,np
1374 psgn(i)=sign(ftime,deltap(i))
1375 do 250 i=2,n
1376 term(i)=fsgn(i)*psgn(i)*ln(i)*amin1(psgn(i)*diff(i+1),psgn(i)
1377 *flxm(i-1),psgn(i)*flxp(i+1))
1378 term(i)=fsgn(i)*psgn(i)*ln(i)*amin1(psgn(i)*diff(2),psgn(i)
1379 *flxp(2))
1380 term(np)=1.0e+75
1381 do 260 i=1,np
1382 fabs(i)=amin1(term(i),fabs(i))
1383 do 270 i=2,n
1384 term(i)=psgn(i)*fsgn(i)*ln(i-1)*amin1(psgn(i)*diff(i-1),psgn(i)
1385 *flxp(i+1),psgn(i)*flxm(i-1))
1386 term(np)=psgn(np)*fsgn(np)*ln(n)*amin1(psgn(np)*diff(n),psgn
1387 (np)*flxm(n))
1388 term(i)=1.0e+75
1389 go to 320
1390 c
1391 c calculate the flux-limiting changes on the right and the left.
1392 do 290 i=1,n
1393 term(i)=fsgn(i)*ln(i)*diff(i+1)
1394 term(np)=1.0e+75
1395 rhotrr=rbc*rhot(n-1)+rhorbc
1396 term(np)=fsgn(np)*ln(n)*(rhotrr*rhotr)
1397 do 300 i=1,np
1398 fabs(i)=amin1(term(i),fabs(i))
1399 do 310 i=2,np
1400 term(i)=fsgn(i)*ln(i-1)*diff(i-1)
1401 term(i)=1.0e+75
1402 c
1403 c term and term may also be estimated off the ends of the system.
1404 rhotl=lbc*rhot(2)+rholbc
1405 term(1)=fsgn(1)*ln(1)*(rhotl-rhot(1))
1406 c
1407 c correct the fluxes completely now.
1408 do 330 i=1,np

```

7 6x8

```

1409 330  flxh(i)=fsgn(i)*amax1(0.0,amin1(fabs(i),term(i)))
1410 c
1411 c  calculate the new flux-corrected densities.
1412 do 340 i=1,n
1413 340  rhon(i)=rln(i)*(lnrho(i)-(flxh(i+1)-flxh(i)))
1414 1source=.false.
1415 do 350 i=1,np
1416 350  source(i)=0.0
1417 return
1418 c
1419 c
1420 c
1421 c  entry velocd(uh,npa,dt,ncol)
1422 c
1423 cd
1424 cd
1425 cd
1426 cd  velocd (uh, npa, dt)
1427 cd  description: this entry calculates all velocity-dependant coeffs.
1428 cd  for multicoefficient fct
1429 cd
1430 cd  arguments:
1431 cd  uh  real array(npa)  flow velocity at cell interfaces
1432 cd  npa  integer  number of cell interfaces = n + 1
1433 cd  dt  real  stepsize for the time integration
1434 cd  ncol  integer that specifies the number of n values
1435 cd
1436 c
1437 c  calculate the interface area x velocity differential x dt.
1438 np=npa
1439 n=np-1
1440 do 360 i=1,np
1441 360  adudth(i)=ah(i)*dt*uh(i)-adugth(i)
1442 c
1443 c  calculate the half-cell epsilon (v*dt/dx)
1444 do 370 i=1,np
1445 370  epsih(i)=adudth(i)*r1h(i)
1446 c
1447 c  next calculate the diffusion and antidiffusion coefficients.
1448 c  variation with epsilon means fourth-order accurate phases.
1449 do 380 i=1,np
1450 380  gamma(i)=0.2+0.2*epsh(i)*epsh(i)
1451 380  nu1h(i)=-c4+c3*epsh(i)**2
1452 380  nu1h(i)=c1-c1*epsh(i)**2
1453 380  if (ncol.eq.0) go to 420
1454 do 390 i=1,ncol
1455 390  nu1h(i)=c1+c2*epsh(i)*epsh(i)
1456 390  nu1h(i)=0.25-0.5*nu1h(i)
1457 390  gamma(i)=0.0
1458 go to 420
1459 continue
1460 do 410 i=1,np
1461 410  nu1h(i)=c1+c2*epsh(i)*epsh(i)
1462 410  nu1h(i)=0.25-0.5*nu1h(i)
1463 410  gamma(i)=0.0
1464 continue
1465 do 430 i=1,np
1466 430  rdt=1.0/dt
1467 430  diff(i)=uh(i)*rdt*(rho1(i)-rho2(i))
1468 430  nu1h(i)=1h(i)*nu1h(i)
1469 430  gamma(i)=1h(i)*gamma(i)
1470 430  nu1h(i)=1h(i)*nu1h(i)
1471 430  gamma(i)=1h(i)*gamma(i)
1472 430  nu1h(i)=1h(i)*nu1h(i)

```

```

1473 C
1474 C now calculate vdtodr for convct.
1475 dt2=2.0*dt
1476 dt4=4.0*dt
1477 do 440 i=2,n
1478 vtdodr(i)=dt4*diff(i)/(rnh(i+1)-rnh(i-1)+roh(i+1)-roh(i-1))
1479 vtdodr(i)=dt2*diff(i)/(rnh(2)-rnh(1)+roh(2)-roh(1))
1480 vtdodr(np)=dt2*diff(np)/(rnh(np)-rnh(n)+roh(np)-roh(n))
1481 return
-----
1482 C
1483 C
1484 C
1485 C
1486 C
1487 cd
1488 cd
1489 cd
1490 cd
1491 cd
1492 cd
1493 cd
1494 cd
1495 cd
1496 cd
1497 cd
1498 cd
1499 cd
1500 C
1501 C
1502 do 450 is=1,nsh
1503 i=ind(is)
1504 adudth(i)=0.0
1505 nuh(i)=0.0
1506 muh(i)=0.0
1507 gammat(i)=0.0
1508 continue
1509 C
1510
1511 C
1512 C
1513 C
1514 C
1515 C
1516 cd
1517 cd
1518 cd
1519 cd
1520 cd
1521 cd
1522 cd
1523 cd
1524 cd
1525 cd
1526 cd
1527 cd
1528 cd
1529 cd
1530 C
1531 C
1532
1533
1534
1535
1536 460

```

entry setbd(ind,nsh)

setbd (ind, nsh)
description: this entry sets some velocity dependent parameters to zero at the various material boundaries to inhibit diffusion of material across the boundaries.

arguments:
ind integer array(nsh) end boundary indices for material interfaces.
nsh integer number of material interfaces.

loop over interfaces.
do 450 is=1,nsh
i=ind(is)
adudth(i)=0.0
nuh(i)=0.0
muh(i)=0.0
gammat(i)=0.0
continue
return

entry ngridd(radh,npa,alpha)

ngride (radh, npa, alpha)
description: this entry sets new geometry variables and coeffs.

arguments:
radh real array(npa) new cell interface positions
npa integer number of cell interfaces = n + 1
alpha integer = 1 for cartesian geometry
= 2 for cylindrical geometry
= 3 for spherical geometry
= 4 for nozzle geometry

calculate the new half-cell positions and grid changes.
np=npa
n=np-1
malpha=abs(alpha)
do 460 i=1,np
rnh(i)=radh(i)


```

1537      do 470 i=1,n
1538      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      do 490 i=1,np
1545      ah(i)=1.0
1546      if (alpha.gt.0) go to 510
1547      do 500 i=1,n
1548      ln(i)=rnh(2)-rnh(i)
1549      go to 600
1550      do 520 i=1,n
1551      ln(i)=rnh(i+1)-rnh(i)
1552      go to 600
1553 c
1554 c      cylindrical coordinates.
1555      do 540 i=1,np
1556      diff(i)=rnh(i)*rnh(i)
1557      ah(i)=pi*(roh(i)+rnh(i))
1558      do 550 i=1,n
1559      ln(i)=pi*(diff(i+1)-diff(i))
1560      go to 600
1561 c
1562 c      spherical coordinates.
1563      do 570 i=1,np
1564      diff(i)=rnh(i)*rnh(i)*roh(i)
1565      scrh(i)=(roh(i)+rnh(i))*roh(i)
1566      do 580 i=1,np
1567      ah(i)=ftpi*(scrh(i)+rnh(i))*2
1568      do 590 i=1,n
1569      ln(i)=ftpi*(diff(i+1)-diff(i))
1570 c
1571 c      now the geometric variables which are system independent.
1572      do 610 i=2,n
1573      ln(i)=amin(ln(i),ln(i-1))
1574      ln(i)=amin(ln(i),ln(2))
1575      ln(np)=amin(ln(n),ln(n-1))
1576      do 620 i=1,n
1577      rin(i)=1.0/ln(i)
1578      if (alpha.gt.0) go to 640
1579      do 630 i=1,np
1580      adugth(i)=ah(i)*(rnh(i)-roh(i))
1581      go to 660
1582      do 650 i=1,np
1583      adugth(i)=ah(i)*(rnh(i)-roh(i))
1584      do 670 i=2,n
1585      rih(i)=0.5*(rin(i)+rin(i+1))
1586      rin(i)=rin(i)
1587      rin(np)=rin(n)
1588      return
1589 c
1590 c
1591 c
1592      entry sourced(na,dt,modes,c,d,dr,dl)
1593 c
1594 cd
1595 cd
1596 cd
1597 cd
1598 cd
1599 cd
1600 cd

```

.....

entry sourced(na,dt,modes,c,d,dr,dl)

.....

sourced (na, dt, modes, c, d, dr, dl)

description: this entry accumulates different source terms.

arguments

na integer number of cells in the system (= n)

```

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=np+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(i)=dth*ah(i)*(d(i)+d(i-1))
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 source=.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(i)=dtq*(d(i)+d(i-1))
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 source=.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)+d*(i)*d(i)
1646 source=.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)=dth*ah(i)*d(i)
1652 do 790 i=1,n
1653 790 source(i)=source(i)+(scrh(i+1)-scrh(i))
1654 source=.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+1)*ah(i+1))
1664 source=.true.

```

```

1665 c      return
1666 c      a dummy for future source terms.
1667 c      . . . .
1668 c      . . . .
1669 c      . . . .
1670 c      840 continue
1671 c      return
1672 c
1673 c
1674 c
1675 c
1676 c
1677 cd
1678 cd
1679 cd
1680 cd
1681 cd
1682 cd
1683 cd
1684 cd
1685 cd
1686 cd
1687 c
1688 c
1689 c
1690 c
1691 c
1692 c
1693 c
1694 c
1695 c
1696 c
1697 c
1698 c
1699 c
1700 c
1701 c
1702 c
1703 c
1704 cd
1705 cd
1706 cd
1707 cd
1708 cd
1709 cd
1710 cd
1711 cd
1712 cd
1713 cd
1714 cd
1715 cd
1716 c
1717 c
1718 c
1719 c
1720 c
1721 c
1722 c
1723 c
1724 c
1725 c
1726 c
1727 c
1728 c

```

description: this entry copies old grid and geometry variables into arrays designated to hold them when new values are defined.

arguments:

 npa integer number of interfaces in the system

copy the previously new grid values to be used as the old grid.

 n=npa-1

 np=npa

 do 850 i=1,n

 lo(i)=ln(i)

 roc(i)=rnc(i)

 rho(i)=rln(i)

 roh(i)=rlo(i)

 do 860 i=1,np

 roh(i)=rnh(i)

 return

entry consrd(rho,na,csum)

consrd (rho, na, csum)

 description: this entry computes the ostensibly conserved sum.

arguments:

 rho real array(na) cell values for physical variable (rho)

 na integer number of cells in the system

 csum real value of the conservation sum of rho

compute the ostensibly conserved total mass (beware your b.c.)

 n=na

 csum=0.0

 do 870 i=1,n

 csum=csum+ln(i)*rho(i)

 return

end

 subroutine dtset (istep,time,radi,ensum)

description: calculates time step as one-half the courant time step to insure positivity for fci.

parameter (mx=150, my=200, mx2=202)

```

1729 parameter (mixx=1, mixy=1)
1730 integer lmgrid
1731 logical lmgstat, lstat, linc, ltherm, lgrav, leos
1732 real rha(mix,mix), rhb(mix,mix), rhc(mix,mix), rho(mix,mix)
1733 real tem(mix,mix), scf(mix,mix), rho(mix,mix)
1734 real rvx(mix,mix), rvy(mix,mix), erg(mix,mix)
1735 common nx, ny, npx, nyp, nspec, idump, ldiag, lmggrid
1736 common lstat, linc, ltherm, lgrav, leos
1737 common alpha, gamma, gml, dt, dx, dy, rhomin
1738 common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
1739 c
1740 real xcor(mix), ycor(mix), ycoro(mix), unit(mix)
1741 common grids/ xcor, ycor, ycoro, unit
1742 c
1743 real rhom(mix), pres(mix), vel(mix), ufl(mix)
1744 real ergk(mix), srvx(mix), srvy(mix), serg(mix)
1745 real rho0(mix), rho0(mix), rho0(mix), vfl(mix)
1746 real rho0(mix), rvx0(mix), rvy0(mix), erg0(mix)
1747 real rhoh(mix), rvxh(mix), rvyh(mix), ergh(mix)
1748 real delx(mix), dely(mix), scrh(mix), sgrv(mix), sgam(mix)
1749 real rmin(mix), rmax(mix), rvr(mix)
1750 common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
1751 rho0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
1752 ., ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
1753 c
1754 common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, iir,
1755 jbt, iadd, jadd, factor
1756 c
1757 real pr00(mix), rh00(mix), er00(mix), gp00(mix), gravity(mix)
1758 parameter (jthm=1)
1759 real fsky(mix), rhtl(mix,jthm)
1760 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
1761 ., jtherm
1762 c
1763 real rscr(mix), pscr(mix), gscr(mix), scr1(mix), scr2(mix)
1764 equivalence (rscr,rmin), (pscr,rmax), (gscr,rvr), (scr1,sgrv)
1765 equivalence (scr2,scrh)
1766 real gams(mix), vels(mix), deltat(mix)
1767 equivalence (sgam,gams), (srvx,vels), (deltat,srvy)
1768 c
1769 common /tstep/ dtmin, dtmax
1770 c
1771 dtold=dt
1772 vfluid=0.0
1773 vsound=0.0
1774 tmstep=dtmax
1775 it=0
1776 jt=0
1777 c
1778 c get delta x and delta y.
1779 do 10 i=ixl,ixr
1780 delx(i)=xcor(i+1)-xcor(i)
1781 do 20 j=iyb,jyt
1782 dely(j)=ycor(j+1)-ycor(j)
1783 c
1784 do 70 j=iyb,jyt
1785 c
1786 c maximum fluid velocity and sound speed calculation.
1787 do 30 i=ixl,ixr
1788 rhom(i)=1.0/rho(i,j)
1789 vel(i)=(rvx(i,j)+rvx(i,j)+rvy(i,j)+rvy(i,j))*rhom(i)
1790 ergk(i)=amax(0.0,erg(i,j))*0.5*vel(i)-rho(i,j)*gp00(j+1)
1791 vel(i)=sqrt(vel(i)*rhom(i))
1792 if (leos) then

```

```

1793 C      call eos (lir,nspec,rho(ixl,j),ergk(ixl,j),gams(ixl),pres(ixl)
1794 C      ,rscr(ixl),pscr(ixl),gscr(ixl),scr(ixl),scr2(ixl),rho0(ixl)
1795 C      ,rho0(ixl))
1796 C      call eospl (rho(ixl,j),ergk(ixl,j),lir,gams(ixl),pres(ixl))
1797 C      do 40 i=ixl,ixr
1798 C      vels(i)=sqrt(gams(i)*pres(i)*rhom(i))
1799 C      else
1800 C      do 50 i=ixl,ixr
1801 C      pres(i)=gm1*ergk(i)
1802 C      vels(i)=sqrt(gamma*pres(i)*rhom(i))
1803 C      endif
1804 C      do 60 i=ixl,ixr
1805 C      deltat(i)=amin(dely(i),dely(j))/(vel(i)+vels(i))
1806 C      it=imin(lir,deltat(ixl),1)+ixl-1
1807 C      if (deltat(it).gt.timestep) go to 70
1808 C      timestep=deltat(it)
1809 C      vfluid=vel(it)
1810 C      vsound=vels(it)
1811 C      jt=j
1812 C      continue
1813 C
1814 C      determine the time step
1815 C      dt=amax1(dtmn,amin1(1,1-dtold,amin1(dtmx,0.5*timestep)))
1816 C      dt=cvmgt(0.1*(float(istep)*dt+float(10-istep)*dtmin),dt,istep
1817 C      ,it,10)
1818 C      time=time+dt
1819 C      write (6,80) istep,time,dt,vfluid,vsound,it,jt,radi,ixr,jxr
1820 C      ,ixl,jxl,lyt,lyb,jyb
1821 C      return
1822 C
1823 C      80 format ('s',15,'x','t=',1pe10.3,'dt=',1pe10.3,'vfs=',2(1pe10.3),',at
1824 C      1',214,'r',1pe10.3,'lrltb',814)
1825 C      end
1826 C      subroutine restrt (istep,time,restf,label)
1827 C      -----
1828 C      reads data from dump file to restart calculation.
1829 C      -----
1830 C      parameter (mx=150, mny=200, mxy=202)
1831 C      character *8 restf, dumpf, dfile
1832 C      character *25 path
1833 C      character *40 glab
1834 C      character *40 label
1835 C      character *40 labl
1836 C      logical isave
1837 C      integer otape, ofile
1838 C      common /inout/ itape, otape, ifile, ofile, isave, rpath, nlab
1839 C
1840 C      open (unit=itape,file=restf,status='old')
1841 C      read (itape) nfile,dfile,nstep,time,label
1842 C      write (6,60)
1843 C      write (6,50) ifile,nstep,time,nfile,dfile,label
1844 C      if (ifile.ne.nfile) go to 30
1845 C      if (istep.ne.nstep+1) go to 40
1846 C      call datain (itape)
1847 C      close (unit=itape,status='keep')
1848 C      write (6,60)
1849 C      return
1850 C
1851 C      entry dumper(istep,time,dumpf,label,path,glab)
1852 C      -----
1853 C      this entry dumps data at the specified time step.
1854 C      -----
1855 C      write (dfile,100) ofile
1856 C      do 10 i=5,7

```

```

1857 if (dfile(1:1).eq.'') dfile(1:1)='0'
1858 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.isave) go to 20
1869 write (glab,80) path(1:npath),dfile
1870 ler=1000
1871 call mass ('save','\wait=on',0,glab,terr)
1872 write (6,70) dfile,terr
1873 ofile=ofile+1
1874 write (6,60)
1875 return
1876 c
1877 c error exits follow
1878 write (6,110) ifile,nfile
1879 stop
1880 write (6,120) lstep,nstep
1881 stop
1882 c
1883 format ('. . . restart off file ',i2,' at step no.',i6,' and time
1884 1 =',lpe12.4,/,/, reading file no.',i3,' which is labeled ',a,
1885 2 =',a,/)
1886 format ('1. . . . .',)
1887 format (1x,'terr for ',a,' =',l4/)
1888 format ('5',a,a,'8')
1889 format ('. . . . . dumped onto file ',i2,' at step no.',i6,' and time
1890 1 =',lpe12.4,/,/, this file is labeled ',a,/,/,ix,a,/)
1891 format (18)
1892 format ('Of file required ('',i2,') does not equal file found', ('
1893 1',i2,')',/)
1894 format (1x,'minstp should be last dump step + 1//ix,'minstp, last
1895 1dump step=',2i5/ix,'run aborted')
1896 end
1897 i subroutine eos (np,nspec,rho,ein,gam,pre,rscr,pscr,gscr,scr1
1898 ,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 scr1(jk)=1.0/rscr(jk)
1911 go to (60,40,20), nspec
1912 c
1913 c species 3.
1914 call eos3 (rho,ein,np,gscr,pscr)
1915 do 30 jk=1,np
1916 scrh(jk)=rhb(jk)*scr1(jk)
1917 rscr(jk)=cvmgt(0.0,scrh(jk),scrh(jk),1t,0.1)
1918 pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
1919 gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
1920 rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rhb(jk),scrh(jk) eq 0.0)

```

```

1921 C
1922 C
1923 C
1924 C
1925 C
1926 C
1927 C
1928 C
1929 C
1930 C
1931 C
1932 C
1933 C
1934 C
1935 C
1936 C
1937 C
1938 C
1939 C
1940 C
1941 C
1942 C
1943 C
1944 C
1945 C
1946 C
1947 C
1948 C
1949 C
1950 C
1951 C
1952 C
1953 C
1954 C
1955 C
1956 C
1957 C
1958 C
1959 C
1960 C
1961 C
1962 C
1963 C
1964 C
1965 C
1966 C
1967 C
1968 C
1969 C
1970 C
1971 C
1972 C
1973 C
1974 C
1975 C
1976 C
1977 C
1978 C
1979 C
1980 C
1981 C
1982 C
1983 C
1984 C

species 2.
40 call eos2 (rho,ein,np,gscr,pscr)
do 50 jk=1,np
scrh(jk)=rha(jk)*scri(jk)
scrh(jk)=cvmgt(0.0,scrh(jk),scrh(jk),l(t.o.1)
pre(jk)=pre(jk)+pscr(jk)*scrh(jk)
gam(jk)=gam(jk)+(gscr(jk)-1.0)*scrh(jk)
rscr(jk)=cvmgt(rscr(jk),rscr(jk)-rha(jk),scrh(jk),eq.0.0)

species 1.
60 call eos1 (rho,ein,np,gscr,pscr)
do 70 jk=1,np
scri(jk)=rscr(jk)*scri(jk)
pre(jk)=amax1(0.0,pre(jk)+pscr(jk)*scri(jk))
gam(jk)=gam(jk)+(gscr(jk)-1.0)*scri(jk)
return

subroutine eospl (rrr,eee,n,gamma,ppp)
equation of state routine for air.

input variable definitions.
rrr = mass density
eee = internal energy per unit volume
      (converted for internal use to energy per unit mass)
n = number of entries in arrays rrr & eee

this routine calculates gamma and ppp as follows:
gamma = 1.0 + p/(rho*e)
ppp = pressure(units of eee)

parameter (m=64)

dimension rrr(n), eee(n), gamma(n), ppp(n)
dimension t11(m), t12(m), t21(m), t22(m), rho(m), e(m), int(m)
dimension p(m), omp(m), q(m), f(m), j(m), tem(m)
dimension g(8,105), jcy(m), js(m)
dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
dimension g7(120), g8(840)

equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
equivalence (g7(1),g(1,91)), (int(1),tem(1)), (gf(1),g(1,1))

data rzero1 /774.413/
data x116e /2.7725887222397744835689081810414791107177734375/
data r116e /0.360673760222409577734651975333690643310546875/

g = gamma - 1.0 is stored for 32 bit word machines in powers of
16 across for mass density variation and intermediate values
l = 16 for powers of 16 vertically which represent the internal
energy variation.

16**(2) .ge. rho .gr. 16**(6)
16**(8) .le. e .le. 16**(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/
data g2 /3918.3918.3918.3918.3918.3918.3918.3918.3918.3723.37
115.3707.3699.3690.3680.3663.3637.3555.3538.3522.3502.3
2476.3430.3344.3238.3370.3370.3364.3347.3277.3099
32885.3227.3227.3201.3134.3062.3014.2884.2591.3166.3110.
43061.2946.2831.2787.2677.2358.3111.3006.2940.2787.2635

```

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 9 179, 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, 1589, 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, 3651, 3438, 2935, 2597, 2336, 2225, 2143, 2116, 3674, 3435, 30
 2023 7 80, 2728, 2608, 2577, 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, 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, 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, 7085, 7
 2036 8 090, 7090, 7090, 7139, 7154, 7169, 7169, 7172, 7177, 7177,
 2037 9 7214, 7231, 7248, 7248, 7251, 7256, 7256, 7256, 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, 8558, 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, 9599, 9622, 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, 9817, 9906, 9912, 9922, 9936, 9937, 9937


```

2049 9 9845,.9915,.9986,.9991,4*.9999,.9915,.9987,6*.9999,.9981,7*.9999/
2050 c
2051 c real air eos, table lookup on gilmore data. (no temp. model)
2052 c to avoid costly logarithmic functions the table "g" is stored in a
2053 c form so that the hexadecimal word structure of a 32 bit machine
2054 c may be exploited.
2055 c this logic may be transferred to other machines by recalculating
2056 c the table "g" appropriate to the word architecture of that machine.
2057 c machine dependent functions and key numbers must also be charged.
2058 c
2059 -----
2059 c
2060 nr=n
2061 nst=mn0(nr,m)
2062 c
2063 do 20 ire=1,nst
2064 rho(ire)=zero1*rrr(1st+ire)
2065 e(ire)=amax1(3.e8,eee(1st+ire)/rrr(1st+ire))
2066 c
2067 c calculate mass density variation index "i".
2068 tem(ire)=alog(rho(ire))*r116a+500 0
2069 i(ire)=tem(ire)
2070 omp(ire)=tem(ire)-i(ire)
2071 i(ire)=502-i(ire)
2072 p(ire)=1.0-omp(ire)
2073 i(ire)=max0(i(ire),1)
2074 c
2075 c calculate internal energy variation index "j".
2076 tem(ire)=alog(e(ire))*r116a
2077 jcy(ire)=ifix(tem(ire))
2078 tem(ire)=tem(ire)-jcy(ire)
2079 tem(ire)=exp(x116a*tem(ire))
2080 jcy(ire)=jcy(ire)-7
2081 js(ire)=tem(ire)
2082 q(ire)=tem(ire)-js(ire)
2083 j(ire)=js(ire)+15*jcy(ire)
2084 j(ire)=mno(j(ire),104)
2085 i(ire)=i(ire)+8*j(ire)
2086 i(ire)=j(ire)-8
2087 do 30 ire=1,nst
2088 t11(ire)=gf(i(ire))
2089 t21(ire)=gf(i(ire)+1)
2090 t12(ire)=gf(j(ire))
2091 t22(ire)=gf(j(ire)+1)
2092 c
2093 c calculate gamma by linear interpolation.
2094 do 40 ire=1,nst
2095 t12(ire)=t12(ire)-t11(ire)
2096 t22(ire)=t22(ire)-t21(ire)
2097 gamma(1st+ire)=1.0+omp(ire)*(t11(ire)+q(ire)+t12(ire))+p(ire)+
2098 (t21(ire)+q(ire)+t22(ire))
2099 c
2100 nr=nr-nst
2101 1st=1st+nst
2102 if (nr.gt.0) go to 10
2103 c
2104 c calculate presura in units of ena
2105 do 50 ire=1,n
2106 ppp(ire)=amax1(0.0,eee(ire)*(gamma(ire)-1.0))
2107 c
2108 return
2109 and
2110 subroutine eos2 (rrr,eee,n,gamma,ppp)
2111 c
2112 c equation of state for dust.

```

```

2113 c      real rrr(n), eee(n), gamma(n), ppp(n)
2114 c
2115 c      do 10 i=1,n
2116 c      gamma(i)=1.0
2117 c      ppp(i)=0.0
2118 c      10
2119 c      return
2120 c
2121 c      end
2122 c      subroutine radlos (t,xcor,ycor,radl,fracn)
2123 c
2124 c      subroutines radlos and fbloss perform calculations to account
2125 c      for the radiative energy loss from the fireball to the
2126 c      surroundings.
2127 c
2128 c      the calculations involve the following steps:
2129 c      1. call subroutine fbloss to compute the radiative energy loss and
2130 c      2. subtract this energy proportionately from the entire grid.
2131 c
2132 c      arguments:
2133 c      t = time.
2134 c      dt = time step.
2135 c      xcor = x coordinates of grid points.
2136 c      ycor = y coordinates of grid points.
2137 c      rho = fluid density.
2138 c      rvx = x momentum.
2139 c      rvy = y momentum.
2140 c      erg = energy per unit volume and
2141 c      radl = cumulative radiation energy loss.
2142 c      fracn= fraction of yield deposited in grid.
2143 c
2144 c      parameter (mnx=150, mny=200, mxy=202)
2145 c      parameter (mxxx=1, mnyy=1)
2146 c      integer alpha
2147 c      logical lmgrid, lstat, lnuc, ltherm, lgrav, leos
2148 c      real rha(mnx,mny), rhb(mnx,mny), rhc(mnx,mny)
2149 c      real tem(mnx,mny), scl(mnx,mny), rho(mnx,mny)
2150 c      real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
2151 c      common rix, riy, rixp, rixy, nspec, indump, ldiag, lmgrid
2152 c      common lstat, lnuc, ltherm, lgrav, leos
2153 c      common alpha, gamma, gmi, dt, dx, dy, rhomin
2154 c      common tem, scl, rho, rvx, rvy, erg, rha, rhb, rhc
2155 c      real xcor(mxy), ycor(mxy)
2156 c
2157 c      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
2158 c      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
2159 c      real rha0(mxy), rhb0(mxy), rhc0(mxy), vh(mxy)
2160 c      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
2161 c      real rhoh(mxy), rvah(mxy), rvyh(mxy), ergh(mxy)
2162 c      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
2163 c      real rmin(mxy), rmax(mxy), rvr(mxy)
2164 c      common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
2165 c      rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvah, rvyh
2166 c      , ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
2167 c
2168 c      real pr00(mxy), rho0(mxy), er00(mxy), gp00(mxy), gravity(mxy)
2169 c      parameter (jthm=1)
2170 c      real fsky(mxy), rhtl(mnx,jthm)
2171 c      common /grav/ pr00, rho0, er00, gp00, gravity, fsky, rhtl
2172 c      , jtherm
2173 c      common /active/ lxr, jxr, lxl, jyl, jyt, lyb, lyb, llr.
2174 c      jbt, ladd, jadd, factor
2175 c
2176 c      compute the energy radiated by the fireball in dt seconds.
2177 c      call fbloss (t,dt,f)

```

```

2177 f=f*fracn
2178 radi=radi+f
2179 if (f.eq.0.0) return
2180 c subtract this energy proportionately from the entire grid.
2181 tsum=0.0
2182 do 20 j=jyb.jyt
2183 do 10 i=ixl.ixr
2184 rhom(i)=1.0/rho(i,j)
2185 ergk(i)=rhom(i)*amax1(0.0,erg(i,j)-0.5*rhom(i)*(rvx(i,j)*rvx(i
2187 .j)+rvy(i,j)*rvy(i,j))-rho(i,j)*gp00(j+1))
2188 ergk(i)=ergk(i)+ergk(i)*ergk(i)
2189 tsum=tsum+sum(ixr-ixl+1,ergk,i)
2190 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 (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 .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)
2201 .ergk(i)/uh(i),erg(i,j),1,t,1.1*er00(j+1))
2202 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 c 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*log(sigma)-3.72132e-06*log
2227 (sigma))*2-1.51933e-07*log(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=1.2586*rhoa**(0.17827)
2238 bnl=10.0
2239 if (rhoa.lt.4.0e 06) bnr=5771.13*rhoa**(0.51428)
2240 bt ime=0.91981*tmax*rhoa**(-0.072126)

```

```

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

```

```

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 jbt, iadd, jadd, factor
2310 c
2311 real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
2312 parameter (jthm=1)
2313 real fsky(mxy), rhtl(mnx, jthm)
2314 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
2315 , jtherm
2316 c
2317 real rhs(mxx, nsta), vis(mxx, nsta), gms(mxx, nsta), tme(mxx)
2318 real prs(mxx, nsta), xs(nsta), ys(nsta)
2319 common /stat/ rhs, vis, gms, prs, tme, xs, ys, fracn, eblast,
2320 ixx
2321 c
2322 ixx=ixx+1
2323 tme(ixx)=time
2324 imin=cvmgt(ixl, i, nstep, ne, i)
2325 imax=cvmgt(ixr, mx, nstep, ne, i)
2326 jmin=cvmgt(jyb, j, nstep, ne, i)
2327 jmax=cvmgt(jyt, my, nstep, ne, i)
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 l=imin, imax
2332 if (xs(j).le.xcor(l).or.xs(j).gt.xcor(l+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 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 m=1
2344 l=l-1
2345 if (m.eq.1) l=1
2346 go to 50
2347 m=1
2348 go to 50
2349 m=l+1
2350 continue
2351 fay=(ya(j)-ycor(l))/(ycor(2)-ycor(l))
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=(ya(j)-yb)/(ya-yb)
2356 c
2357 c must get the x fraction of the cell
2358 xfrac=(xs(j)-xcor(l))/(xcor(l+1)-xcor(l))
2359 xfrac=xfrac-0.5
2360 if (xfrac) 70, 110, 150
2361 c
2362 c lower part of the cell
2363 if (l.eq.1) go to 110
2364 xl=(xcor(l)+xcor(l-1))*0.5
2365 xr=(xcor(l)+xcor(l+1))*0.5
2366 fac=(xs(j)-xl)/(xr-xl)
2367 pl=erg(l-1, l)/(erg(l, l)-erg(l-1, m))*fac
2368 p2=erg(l-1, m)/(erg(l, m)-erg(l-1, m))*fac

```

```

2369   ergs=pi+(p2-p1)*fay
2370   r1=rho(i-1,1)*(rho(i,1)-rho(i-1,1))*fac
2371   r2=rho(i-1,m)*(rho(i,m)-rho(i-1,m))*fac
2372   rhas=r1+(r2-r1)*fay
2373   rhas=0.0
2374   rhbs=0.0
2375   rhcs=rhos
2376   go to (100,90,80), nspec
2377   r1=rhb(i-1,1)*(rhb(i,1)-rhb(i-1,1))*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   r1=rha(i-1,1)*(rha(i,1)-rha(i-1,1))*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   continue
2386   u1=rvx(i-1,1)*(rvx(i,1)-rvx(i-1,1))*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-1,1)*(rvy(i,1)-rvy(i-1,1))*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   pi=erg(i,1)
2398   p2=erg(i,m)
2399   ergs=pi+(p2-p1)*fay
2400   r1=rho(i,1)
2401   r2=rho(i,m)
2402   rhas=r1+(r2-r1)*fay
2403   rhas=0.0
2404   rhbs=0.0
2405   rhcs=rhos
2406   go to (140,130,120), nspec
2407   r1=rhb(i,1)
2408   r2=rhb(i,m)
2409   rhbs=r1+(r2-r1)*fay
2410   rhcs=rhcs-rhbs
2411   r1=rha(i,1)
2412   r2=rha(i,m)
2413   rhas=r1+(r2-r1)*fay
2414   rhcs=amax1((rhcs-rhas),0.0)
2415   continue
2416   u1=rvx(i,1)
2417   u2=rvx(i,m)
2418   rvxs=u1+(u2-u1)*fay
2419   v1=rvy(i,1)
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   x1=(xcor(i)+xcor(i+1))*0.5
2426   xr=(xcor(i+1)+xcor(i+2))*0.5
2427   fac=(xs(j)-x1)/(xr-x1)
2428   pi=erg(i,1)+fac*(erg(i+1,1)-erg(i,1))*fac
2429   p2=erg(i,m)+fac*(erg(i+1,m)-erg(i,m))*fac
2430   ergs=pi+(p2-p1)*fay
2431   r1=rho(i,1)+fac*(rho(i+1,1)-rho(i,1))*fac
2432   r2=rho(i,m)+fac*(rho(i+1,m)-rho(i,m))*fac

```

```

2433 rhos=r1+(r2-r1)*fay
2434 rhos=0.0
2435 rhbs=0.0
2436 rhcs=rhos
2437 go to (180,170,160), nspec
2438 r1=rhb(1,1)*(rhb(1+1,1)-rhb(1,1))*fac
2439 r2=rhb(1,m)*(rhb(1+1,m)-rhb(1,m))*fac
2440 rhbs=r1+(r2-r1)*fay
2441 rhcs=rhos-rhbs
2442 r1=rha(1,1)*(rha(1+1,1)-rha(1,1))*fac
2443 r2=rhal(1,m)*(rha(1+1,m)-rha(1,m))*fac
2444 rhbs=r1+(r2-r1)*fay
2445 rhcs=amax1((rhcs-rhas),0.0)
2446 continue
2447 u1=rvx(1,1)+(rvx(1+1,1)-rvx(1,1))*fac
2448 u2=rvx(1,m)+(rvx(1+1,m)-rvx(1,m))*fac
2449 rvxs=u1+(u2-u1)*fay
2450 v1=rvy(1,1)+(rvy(1+1,1)-rvy(1,1))*fac
2451 v2=rvy(1,m)+(rvy(1+1,m)-rvy(1,m))*fac
2452 rvys=v1+(v2-v1)*fay
2453 continue
2454 rhm(k,j)=rhos
2455 rhom=1.0/rhos
2456 vis(k,j)=0.5*(rvxs+rvxs+rvys+rvys)*rhom
2457 eint=amax1(0.0,ergs-vis(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 ,scr1,scr2,rhas,rhbs)
2461 call eospl (rhos,eint,1,gms(k,j),prs(k,j))
2462 else
2463 prs(k,j)=gmi*eint
2464 gms(k,j)=gamma
2465 endif
2466 vis(k,j)=sqrt(2.0*vis(k,j)*rhom)
2467 continue
2468 return
2469
2470 end
2471 c -----
2472 c hydro advance for grid containing bluff bodies.
2473 c -----
2474 parameter (mx=150, mny=200, mxy=202)
2475 parameter (nmbd=2, nbpl=nmbd+1)
2476 c
2477 logical lscan, rgt, ltop, left, botm
2478 common /scan/ lscan, rgt, left, ltop, botm
2479 common /active/ ixr, jxr, ixl, jxl, iyt, jyt, iyb, jyb, ilr,
2480 jbt, iadd, jadd, factor
2481 c
2482 real xcor(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
2483 common /grids/ xcor, ycoro, ycor, ycoro, unit
2484 c
2485 real rhom(mxy), pres(mxy), vel(mxy), uhi(mxy)
2486 real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
2487 real rho0(mxy), rho0(mxy), rho0(mxy), rho0(mxy), vhi(mxy)
2488 real rho0(mxy), rho0(mxy), rho0(mxy), rho0(mxy), erg0(mxy)
2489 real rho0(mxy), rho0(mxy), rho0(mxy), rho0(mxy), erg0(mxy)
2490 real delx(mxy), dely(mxy), rmax(mxy), scrh(mxy), sgam(mxy)
2491 real rmin(mxy), rmax(mxy), rvr(mxy)
2492 common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
2493 rho0, rho0, rho0, vhi, rho0, rho0, rho0, rho0, rho0, rho0, rho0, rho0,
2494 , ergk, dely, dely, scrh, sgam, rmin, rmax, rvr
2495 c
2496 integer nleff(mxy,nbpl), nrlq(mxy,nbpl), lmbf(nbpl), lmx(nbpl)

```

```

2497 integer nbot(mix,nbpt), ntop(mnx,nbpt), jmn(nbnd), jmx(nbnd)
2498 common /bodint/ nief, nrig, lmn, lmx, nbot, ntop, jmn, jmx
2499 c
2500 c
2501 when active grid does not include bodies.
2502 if (ixr.lt.lmin+1.or.jyt.lt.jmin+1.or.ixl.gt.lmax+1.or.jyb.gt
2503 .jmax+1) then
2504 call bndrx (bcr,bcl)
2505 call integx
2506 call bndry (bct,ccb)
2507 call integy
2508 return
2509 endif
2510 c
2511 adjust active grid boundaries if necessary.
2512 if (lscan) then
2513 ixl=cvmgt(max0(lmin-jadd,1),ixl,lmin.le.ixl.and.lmax.ge.ixl)
2514 ixr=cvmgt(min0(lmax+jadd,mnx),ixr,lmin.le.ixr.and.lmax.ge.ixr)
2515 jyb=cvmgt(max0(jmin-jadd,1),jyb,jmin.le.jyb.and.jmax.ge.jyb)
2516 jyt=cvmgt(min0(jmax+jadd,mny),jyt,jmin.le.jyt.and.jmax.ge.jyt)
2517 c
2518 reset boundary conditions.
2519 abcl=cvmgt(bcl,0.0,ixl.eq.1)
2520 abcr=cvmgt(bcr,0.0,ixr.eq.mnx)
2521 abcb=cvmgt(ccb,0.0,jyb.eq.1)
2522 abct=cvmgt(bct,0.0,jyt.eq.mny)
2523 endif
2524 c
2525 save grid boundaries.
2526 isavl=ixl
2527 isavr=ixr
2528 jsavb=jyb
2529 jsavt=jyt
2530 c
2531 row integration.
2532 c
2533 region below bodies.
2534 if (jmin.gt.jsavb) then
2535 jyt=jmin-1
2536 jbt=jyt-jyb+1
2537 call bndrx (abcr,abcl)
2538 call integx
2539 endif
2540 c
2541 region including the bodies.
2542 do 10 j=jmin,jmax
2543 jyb=j
2544 jyt=j
2545 jbt=1
2546 do 10 n=1,nbodpt
2547 if (nief(j,n).ne.0) then
2548 ixl=nief(j,n)
2549 ixr=nrig(j,n)
2550 ijr=ixr-ixl+1
2551 aabcl=cvmgt(abcl,-1.0,ixl.eq.isavl)
2552 aabcr=cvmgt(abcr,-1.0,ixr.eq.isavr)
2553 call bndrx (aabcr,aabcl)
2554 call integx
2555 endif
2556 continue
2557 c
2558 region above bodies.
2559 if (jmax.lt.jsavt) then
2560 ixl=isavl
2561 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         xl=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         xl=i
2586         ixr=i
2587         ilr=i
2588         do 20 n=1,nbodi
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,1.lt.imn(i))
2595               aabct=cvmgt(abct,-1.0,jyt.eq.jsavt)
2596               call bndry (aabct,aabcb)
2597               call integy
2598               endif
2599            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,abch)
2610         call integy
2611         endif
2612 C
2613 C      reset active grid boundaries.
2614         ixr=isavr
2615         xl=isavl
2616         jyb=jsavb
2617         jyt=jsavt
2618         ilr=ixr-ixl+1
2619         jbt=jyt-jyb+1
2620         return
2621 C
2622         entry bluf(nbot)
2623 C
2624 C      initialize bluff body parameters.

```

```

2625 C      nbotp1=nbot+1
2626       imin=1
2627       imax=mnx
2628       jmin=1
2629       jmax=my
2630
2631 C
2632 C      body boundaries.
2633       imn(1)=18
2634       imx(1)=24
2635       jmn(1)=1
2636       jmx(1)=105
2637       imn(2)=18
2638       imx(2)=52
2639       jmn(2)=158
2640       jmx(2)=164
2641 C
2642 C      initialize body integers to zero.
2643       do 50 n=1,nbotp1
2644         do 30 j=1,my
2645           nlef(j,n)=0
2646           nrig(j,n)=0
2647         do 40 i=1,mnx
2648           nbot(i,n)=0
2649         ntop(i,n)=0
2650       continue
2651 C
2652 C      body integers.
2653 C      left, segment 1.
2654       do 60 j=1,my
2655         nlef(j,1)=1
2656 C
2657 C      left and right, segment 2.
2658       do 70 n=1,nbot
2659         do 70 j=jmn(n),jmx(n)
2660           nlef(j,2)=imx(n)+1
2661         do 70 nrig(j,2)=mnx
2662 C
2663 C      right, segment 1.
2664       do 80 n=1,nbot
2665         do 80 j=jmn(n),jmx(n)
2666           nrig(j,1)=imn(n)-1
2667         do 90 j=jmx(1)+1,jmn(2)-1
2668           nrig(j,1)=mnx
2669         do 100 j=jmx(2)+1,my
2670           nrig(j,1)=mnx
2671 C
2672 C      bottom, segment 1.
2673       do 110 i=1,imn(1)-1
2674         nbot(i,1)=1
2675         do 120 i=imn(1),imx(1)
2676           nbot(i,1)=jmx(1)+1
2677         do 130 i=imx(1)+1,mnx
2678           nbot(i,1)=1
2679 C
2680 C      bottom and top, segment 2.
2681       do 140 i=imn(2),imx(2)
2682         nbot(i,2)=jmx(2)+1
2683         ntop(i,2)=my
2684 C
2685 C      top, segment 1.
2686       do 150 i=1,imn(1)-1
2687         ntop(i,1)=my
2688         do 160 i=imn(2),imx(2)

```

```

2689      160      ntop(1,1)=jmn(2)-1
2690      do 170 i=lmx(2)+1,lmx
2691      ntop(1,1)=mny
2692      return
2693      end
2694      subroutine dpshok (rho,rvx,rvy,erg)
2695      c-----
2696      c deposits shock in shock tube
2697      c-----
2698      parameter (mny=150, mny=200, mxy=202)
2699      parameter (nrbd=2, nbp1=nrbd+1)
2700      real rho(mnx,mny), erg(mnx,mny)
2701      real rvx(mnx,mny), rvy(mnx,mny), mach
2702      c-----
2703      integer nlef(mny,nbp1), nrig(mny,nbp1), lmn(nrbd), lmx(nrbd)
2704      integer rbot(mnx,nbp1), ntop(mnx,nbp1), jmn(nrbd), jmx(nrbd)
2705      common /botint/ nlef, nrig, lmn, lmx, nbot, ntop, jmn, jmx
2706      data pi, p2, r1 /1.013e6,3.42e6,1.19e-3/
2707      logical leos
2708      common /bbc/, rhobbc, rvxbbc, rvybbc, ergbbc
2709      c-----
2710      c set up shock
2711      do 10 j=1,jmx(1)
2712      do 10 i=1,lmn(1)-1
2713      rho(1,j)=rhobbc
2714      rvx(1,j)=rvxbbc
2715      rvy(1,j)=rvybbc
2716      erg(1,j)=ergbbc
2717      return
2718      c-----
2719      c entry jump(gamma,leos,ergamb)
2720      c-----
2721      c calculate conditions behind shock
2722      c-----
2723      c compute internal energy, mach number and density
2724      p2opi=p2/pi
2725      ergbbc=p2/(gamma-1.0)
2726      mach=sqrt(((gamma+1.0)*p2opi+(gamma-1.0))/(2.0*gamma))
2727      rhobbc=r1*(gamma+1.0)*mach*mach/(gamma-1.0)*mach*mach+2.0)
2728      c-----
2729      c iterate to get better gamma and energy
2730      gam=gamma
2731      pre=pi
2732      if (.not.leos) go to 30
2733      kount=0
2734      if (kount.eq.50) then
2735      write (6,40)
2736      stop
2737      endif
2738      call eospi (rhobbc,ergbbc,1,gam,pre)
2739      kount=kount+1
2740      ergbbc=p2/(gam-1.0)
2741      mach=sqrt(((gam+1.0)*p2opi+(gam-1.0))/(2.0*gam))
2742      rhobbc=r1*(gam+1.0)*mach*mach/(gamma-1.0)*mach*mach+2.0)
2743      if (abs(pre-p2).gt.1.e-6*p2) go to 20
2744      c-----
2745      c determine ambient sound speed
2746      call eospi (r1,ergamb,1,gam,pre)
2747      camb=sqrt(gam*pre/r1)
2748      c-----
2749      c determining conditions behind shock
2750      rvxbbc=0.0
2751      rvybbc=mach*camb*1.1*rhobbc, r1)
2752      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 (mny=150, mny=200, mxy=202)
2761 parameter (mmax=1, mny=1)
2762 integer alpha
2763 logical lgrid, lstat, lnuc, therm1, lgrav, leos
2764 real rha(mnx,mny), rhb(mnx,mny), rhc(mnx,mny)
2765 real tem(mnx,mny), sc1(mnx,mny), rho(mnx,mny)
2766 real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
2767 common nx, ny, npx, nyp, nspec, idump, idlag, lgrid
2768 common lstat, lnuc, therm1, lgrav, leos
2769 common alpha, gamma, gmi, dt, dx, dy, rhomin
2770 common tem, sci, 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), prest(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 rha0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
2784 ., ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
2785 c
2786 real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
2787 parameter (jthm=1)
2788 real fsky(mxy), rhtl(mnx,jthm)
2789 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
2790 ., 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,npx)
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 nyp=nyp+1
2807 do 10 j=1,mny
2808 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 j1m=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)=amax(rh00(j),rhomin)

```

```

2817 pr00(j)=rho0(j)*8.3144e7*tam/amass
2818 continue
2819 c
2820 c get gravitational acceleration and potential.
2821 gr2=cvmgt(g0*r0*r0,0.0,igrav)
2822 do 30 j=1,nypp
2823 ygp(j)=ygp(j)*1.e5
2824 do 40 j=1,jlim
2825 gravi(j)=-gr2/(r0+ygp(j))*2
2826 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 er00(j)=pr00(j)/(gamma-1.0)
2831 if (.not.leas) go to 90
2832 kount=0
2833 if (kount.eq.50) then
2834 write (6,260)
2835 stop
2836 endif
2837 call eospl (rho0,er00,jlim,gams,ptest)
2838 kount=kount+1
2839 do 70 j=1,jlim
2840 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 continue
2844 do 100 j=1,jlim
2845 er00(j)=er00(j)+rho0(j)*gp00(j)
2846 c
2847 c set values in remaining cells.
2848 if (lcb.eq.-1.0.or..not.igrav) then
2849 jlim=cvmgt(1,nypp,igrav)
2850 do 110 j=1,jlim
2851 rho0(j)=rho0(2)
2852 pr00(j)=pr00(2)
2853 gravi(j)=gravy(2)
2854 er00(j)=er00(2)
2855 endif
2856 c
2857 c compute "falling sky" term.
2858 if (igrav) then
2859 fsky(1)=pr00(1)
2860 fsky(2)=pr00(2)
2861 do 120 j=2,nypp-1
2862 fsky(j+1)=fsky(j)+rho0(j)*gravy(j)*(ygp(j+1)-ygp(j)-1)
2863 else
2864 do 130 j=1,nypp
2865 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),rho0(jj),pr00(jj),er00(jj),gp00(jj)
2873 .gravy(jj),fsky(jj)
2874 continue
2875 c
2876 c fill fluid variable arrays
2877 do 150 j=1,mny
2878 do 150 k=1,mny
2879 rho(k,j)=rho0(j),
2880 rva(k,j)=0

```

```

2881 rvv(1,j)=0.0
2882 erg(1,j)=er00(j+1)
2883 go to (200,180,160), nspec
2884 do 170 j=1,mny
2885 do 170 i=1,mnx
2886 rrb(1,j)=0.0
2887 do 190 j=1,mny
2888 do 190 i=1,mnx
2889 rha(1,j)=rho(1,j)
2890 continue
2891 c
2892 return
2893 c
2894 210 format (1p10e13.4)
2895 220 format ('Ocor(1) - the x coordinate grid locations ',/.3x)
2896 230 format ('Ocor(j) - the y coordinate grid locations ',/.3x)
2897 240 format (3x,'j',7x,'y',17x,'rho(gm/cc)',3x,'preldynes/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,13.7(5x,e12.5))
2901 260 format (1x,'failure to converge in subroutine initial')
2902 end
2903 -----
2904 subroutine burst (xcor,ycor,erg)
2905 c
2906 c deposits burst energy in 3 zones.
2907 -----
2908 parameter (mnx=150, mny=200, mxy=202)
2909 real erg(mnx,mny), xcor(mxy), ycor(mxy)
2910 c
2911 parameter (mxx=1, nsta=1)
2912 real rhs(mxx,nsta), vis(mxx,nsta), gms(mxx,nsta), tme(mxx)
2913 real prs(mxx,nsta), xs(nsta), ys(nsta)
2914 common /stat/ rhs, vis, gms, prs, tme, xs, ys, fracn, eblast,
2915 1,xx
2916 data pi /3.1415926/, engin/C.0/
2917 namelist /blast/ burstn,eblast,nicel,njct,njcc2
2918 c
2919 read namelist data.
2920 read (5,blast)
2921 write (6,blast)
2922 c
2923 determine energy per cell.
2924 vol=pi*(xcor(nicel+1)**2*(ycor(njcc2+1)-ycor
2925 njct))
2926 ez=eblast*4.2e19*burstn/vol
2927 fracn=1.0
2928 c
2929 distribute burst energy.
2930 do 10 j=njct,njcc2
2931 delty=ycor(j+1)-ycor(j)
2932 do 10 i=1,nicel
2933 vol=pi*(xcor(i+1)+xcor(i))*(xcor(i+1)-xcor(i))*delty
2934 engin=engin+ ez*vol
2935 erg(1,j)=erg(1,j)+ez
2936 write(6,99) engin
2937 format(1x,'energy deposited=',1pe12.5,' ergs'/)
2938 return
2939 end
2940 -----
2941 subroutine cav61 (alt,r,t,m)
2942 c calculate mass density, temperature and mean molecular weight
2943 based on the average model in the "Cira 1961" tables.
2944 c data is stored in the following manner.

```

```

2945 C 0 - 120 km in intervals of 5 km.
2946 C 120 - 300 km in intervals of 10 km and
2947 C 300 - 800 km in intervals of 50km.
2948 C -----
2949 real rh(53), tn(53), am(53), m
2950 data rh /1.23e-03,7.48e-04,4.19e-04,2.02e-04,8.89e-05,4.06e-05
2951 ,1.84e-05,8.69e-06,4.07e-06,2.02e-06,1.02e-06,5.54e-07,3.04e-0
2952 7,1.67e-07,8.84e-08,4.37e-08,1.94e-08,8.12e-09,3.12e-09,1.23e-
2953 09,4.78e-10,2.07e-10,9.49e-11,4.77e-11,2.44e-11,6.95e-12,3.07e
2954 -12,1.69e-12,1.11e-12,8.26e-13,6.59e-13,4.73e-13,3.61e-13,2.77
2955 e-13,2.14e-13,1.67e-13,1.30e-13,1.03e-13,8.11e-14,6.45e-14,5.1
2956 5e-14,4.14e-14,3.34e-14,1.23e-14,5.09e-15,2.33e-15,1.17e-15,6.
2957 15e-16,3.45e-16,2.00e-16,1.19e-16,7.22e-17,4.60e-17/
2958 oata tn /289.3,256.7,222.4,214.1,217.1,222.5,228.8,236.4,247.9
2959 ,260.6,269.6,268.9,257.8,238.6,217.0,198.1,184.6,178.9,181.1,1
2960 92.5,212.1,237.6,265.0,297.6,343.0,569.4,799.1,1015.1,1155.1,17
2961 7.1193,1211.1,1227.1,1243.1,1259.1,1274.1,1288.1,1302.1,1314.1,1326.
2962 4.1338.1,1348.1,1359.1,1401.1,1436.1,1466.1,1474.1/
2963 data am /19.28,97.28,91.28,85.28,78.28,72.28,67.28,60.28,43.28
2964 1.25,28.09,27.90,27.70,27.47,27.25,27.00,26.75,26.48,26.18,25.8
2965 7,25.55,25.21,24.85,24.50,24.11,23.74,21.80,20.00,18.55,17.48.
2966 3 16.31,15.70,15.10,14.62,14.02,13.88/
2967 C
2968 C calculate appropriate index for table look-up.
2969 p1=20*amin1(alt,120.0001)
2970 p2=10*amin1(amax1(alt-120.0,0.0),180.0001)
2971 p3= 02*amax1(alt-300.0,0.0)
2972 i1=p1
2973 i2=p2
2974 i3=p3
2975 i=11+12+i3+1
2976 i=max0(i,1)
2977 i=min0(i,52)
2978 C
2979 C calculate atmospheric quantities by linear interpolation.
2980 p=(p1-i1)+(p2-i2)+(p3-i3)
2981 omp=f.0-p
2982 r=rh(i)*omp+rh(i+1)*p
2983 t=tn(i)*omp+tn(i+1)*p
2984 m=am(i)*omp+am(i+1)*p
2985 return
2986 end
2987 subroutine dposit (time)
2988 C
2989 C sets up a nuclear blast at (x0,y0,t0) or a HE
2990 C blast at (x0,y0,0,0) in a 2D cartesian or cylindrical grid.
2991 C -----
2992 parameter (mxx=150, mny=200, mxy=202)
2993 parameter (mxxx=1, mnyy=1)
2994 parameter (nop=1)
2995 parameter (mxx=1, nsta=1)
2996 C
2997 integer alpha
2998 logical lmgrid, lstat, lnucl, lnucl, ltherml, lgrav, leos
2999 real rha(mxx,mny), rnb(mxxx,mny), rhc(mxxx,mny),
3000 real tea(mxxx,mny), sc1(mxxx,mny), rho(mxx,mny)
3001 real rva(mxx,mny), rvy(mnx,mny), ergl(mxx,mny)
3002 common lstat, lnucl, lnucl, ltherml, lgrav, leos
3003 common alpha, gamma, qml, dt, dy, dz, rhomin
3004 common tem, sc1, rho, rva, rvy, erg, rha, rhb, rhc
3005 C
3006 C real rhom(mxy), pres(mxy), vel(mxy), ubl(mxy)
3007 C real ergp(mxy), svx(mxy), svy(mxy), serq(mxy)
3008 C

```

```

3009 real rho0(mxy), rhb0(mxy), rhc0(mxy), vhl(mxy)
3010 real rho0(mxy), rvc0(mxy), rvy0(mxy), erg0(mxy)
3011 real rhoh(mxy), vxh(mxy), vyh(mxy), ergh(mxy)
3012 real delx(mxy), dely(mxy), scri(mxy), sgrv(mxy), sgam(mxy)
3013 real rmini(mxy), rmax(mxy), rvr(mxy)
3014 common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
3015 rho0, rhb0, rhc0, vh, rhoo, rvx0, rvy0, rhoh, rvxh, rvyh
3016 . ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3017 c
3018 real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unitt(mxy)
3019 common /grids/ xcor, xcoro, ycor, ycoro, unitt
3020 c
3021 real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
3022 parameter (jthm=1)
3023 real fsky(mxy), rhtl(mnx,jthm)
3024 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
3025 . jtherm
3026 c
3027 real rhs(mxx,nsta), vis(mxx,nsta), gms(mxx,nsta), tme(mxx)
3028 real prs(mxx,nsta), xs(nsta), ys(nsta)
3029 common /stat/ rhs, vis, gms, prs, tme, xs, ys, fracn, eblast,
3030 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,vic), (ebl,ein)
3038 c
3039 real pav(mxy), rav(mxy), vav(mxy), gav(mxy), eav(mxy)
3040 real gams(mxy)
3041 integer imin(mxy), imax(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), yf(nop)
3047 common /part/ part, xp, yp, nopp
3048 c
3049 logical lani
3050 namelist /depo/ x0,y0,t0,r0,eblast,lani
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 comparison of the final energy, and thus what the energy is in the
3057 c explosion.
3058 c -----
3059 call ngridd (xcor,xsp,alpha)
3060 call ogridd (nyp)
3061 call ngridd (xcor,xsp,alpha)
3062 do 10 j=1,myy
3063 call consrd (erg(1,j),nx,erg0(j))
3064 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.lmuc) go to 30
3070 if (abs(eblast-1000.)gt.10.) lani=.false.
3071 if (lani) go to 50
3072 c

```



```

3073 c calculate a scale factor for the radius based on the blast energy
3074 c measured in kt. then evaluate ktnuc at radial intervals
3075 c of 1/4 the smallest cell dimension. scale r0 and dr
3076 c according to blast energy.
3077 c -----
3078 rscale=(1.0/eblast)**(1.0/3.0)
3079 dr=0.25*amin1(dx,dy)*rscale
3080 nr=(1.2*rscale*r0/dr)+1
3081 nrp=nr+1
3082 radd=0.0
3083 do 20 i=1,nrp
3084 call ktnuc1 (t0,radd,ppp,dens,eee,vvv,gamma)
3085 pbl(i)=ppp
3086 rbl(i)=dens
3087 vbl(i)=vvv
3088 gbl(i)=gamma
3089 radd=radd+dr
3090 go to 50
3091 c -----
3092 c deposit HE blast energy in the grid using CJDET at
3093 c 1/4 the smallest cell dimension.
3094 c -----
3095 rscale=1.0
3096 dr=0.25*amin1(dx,dy)
3097 nr=(1.1*r0/dr)+1
3098 nrp=nr+1
3099 x=-0.5*dr
3100 r0inv=1.0/r0
3101 do 40 i=1,nrp
3102 x=x+dr
3103 xpass=x*r0inv
3104 call cjdet (xpass,dens,vvv,eee,ppp,gamma)
3105 pbl(i)=ppp
3106 ebl(i)=eee
3107 rbl(i)=dens
3108 vbl(i)=vvv
3109 gbl(i)=gamma
3110 continue
3111 c -----
3112 c calculate cell boundaries relative to (x0, y0). place in delx
3113 c and dely.
3114 do 60 i=1,nrp
3115 delx(i)=xcor(i) *x0
3116 do 70 j=1,nrp
3117 dely(j)=ycor(j)-y0
3118 c -----
3119 c compute lengths of position vectors (rad11) to cell corners.
3120 do 80 i=1,nrp
3121 rha0(i)=delx(i)*delx(i)
3122 do 90 j=1,nrp
3123 rha0(j)=dely(j)*dely(j)
3124 c -----
3125 c now calculate cell centers relative to (x0, y0). place in delx and
3126 c dely.
3127 do 100 i=1,mnx
3128 wh(i)=0.5*(delx(i)+delx(i+1))
3129 do 110 i=1,mix
3130 delx(i)=wh(i)
3131 do 120 j=1,mny
3132 vh(j)=0.5*(dely(j)+dely(j+1))
3133 do 130 j=1,mny
3134 dely(j)=vh(j)
3135 c -----
3136 c compute lengths of position vectors of the cell centers

```

```

3137 do 140 i=1,mny
3138   scrh(i)=delc(i)+delc(i+1)
3139 do 150 j=1,mny
3140   sgrv(j)=dely(j)+dely(j+1)
3141   if (.not.(rnuc) 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 ktrnc 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 ktrnc values to the ambient values set up in Initial.
3148 c -----
3149   pkt=1.01325e6
3150   rhokt=1.225e-3
3151   r00=r0*scale/dr+1
3152 do 290 j=1,mny
3153   do 170 i=1,mnx
3154     rvr(i)=scrh(i)+sgrv(j)
3155 c pick up variables from 2D arrays.
3156 c do 180 i=1,mnx
3157   rvx0(i)=rvx(i,j)
3158   rvy0(i)=rvy(i,j)
3159   rho0(i)=rho(i,j)
3160   rhom(i)=1.0/rho0(i)
3161   erg0(i)=erg(i,j)-rho0(i)*gp00(j+1)
3162   180 -----
3163 c interpolate the blast values onto the grid.
3164 c -----
3165 c find maximum and minimum radius vectors to carriers of each cell.
3166 c scale according to blast energy.
3167 c do 190 i=1,mnx
3168   vel(i)=rho0(i)+rhub0(j)
3169   wh(i)=rho0(i+1)+rhub0(j)
3170   vh(i)=rho0(i)+rhub0(j+1)
3171   rmin(i)=amin(vel(i),wh(i),vh(i),vh(i))
3172   rmax(i)=amax(vel(i),wh(i),vh(i),vh(i))
3173   vel(i)=rho0(i+1)+rhub0(j+1)
3174   rmin(i)=amin(vel(i),rmin(i))
3175   rmax(i)=amax(vel(i),rmax(i))
3176   rmin(i)=sqrt(rmin(i))*rscale
3177   rmax(i)=sqrt(rmax(i))*rscale
3178   190 -----
3179 c interpolate onto grid by averaging between rmin and rmax.
3180 c do 200 i=1,mnx
3181   rmax(i)=rmax(i)/dr+1.0
3182   rmin(i)=rmin(i)/dr+1.0
3183   imax(i)=int(rmax(i))
3184   imin(i)=int(rmin(i))
3185   200 -----
3186 c do 220 i=1,mnx
3187   kmIn=imin(i)
3188   kmInp=kmin+1
3189   kmax=imax(i)
3190   kmaxp=kmax+1
3191   eav(i)=0.0
3192   pav(i)=0.0
3193   rav(i)=0.0
3194   vav(i)=0.0
3195   gav(i)=0.0
3196   220 -----
3197 c if (rmin(i).gt.r00) go to 220
3198 c do 210 k=kminp,kmax
3199 c
3200

```

```

3201 eav(i)=eav(i)+ebi(k)
3202 pav(i)=pav(i)+pbi(k)
3203 rav(i)=rav(i)+rbi(k)
3204 vav(i)=vav(i)+vbi(k)
3205 gav(i)=gav(i)+gbi(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)+ebi(kmin)+flow+ebi(kmax*p)+fhigh
3211 pav(i)=pav(i)+pbi(kmin)+flow+pbi(kmax*p)+fhigh
3212 rav(i)=rav(i)+rbi(kmin)+flow+rbi(kmax*p)+fhigh
3213 vav(i)=vav(i)+vbi(kmin)+flow+vbi(kmax*p)+fhigh
3214 gav(i)=gav(i)+gbi(kmin)+flow+gbi(kmax*p)+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 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   ergk(i)=amax1(0.0,erg0(i)-0.5*rhom(i))*(rvx0(i)+rvy0(i)+rvy0(i))
3229   rvy0(i))
3230   call eospl (rho0,ergk,mx,gams,pres)
3231   do 240 i=1,mnx
3232     if (rmin(i).gt.r00) go to 240
3233     rho0(i)=(rav(i)/rho0t)*rho0(i)
3234     rho0(i)=amax1(rho0(i),rhom(i))
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(i)/uh(i)
3239     pres(i)=(pav(i)/pk0)*pres(i)
3240     gams(i)=gav(i)
3241 C
3242     rhom(i)=1.0/rho0(i)
3243     ergk(i)=(rvx0(i)+rvy0(i)+rvy0(i)+rvy0(i))*0.5*rhom(i)
3244     erg0(i)=pres(i)/(gams(i)-1.0)*ergk(i)
3245     continue
3246   go to 270
3247   do 260 i=1,mnx
3248     if (rmin(i).gt.r00) go to 260
3249     rho0(i)=amax1(rav(i),rhom(i))
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(i)/uh(i)
3254     erg0(i)=eav(i)
3255     continue
3256 C
3257 C refill fluid variable arrays.
3258   do 280 i=1,mnx
3259     rho(i,j)=rho0(i)
3260     rvx(i,j)=rvx0(i)
3261     rvy(i,j)=rvy0(i)
3262     erg(i,j)=erg0(i)+rho0(i)*gp00(i)
3263     continue
3264   go to (220,100,100), nspoc

```

```

3265 do 310 j=1,mny
3266 do 310 i=1,mix
3267 rha(1,j)=cvmgtrho(1,j),0.0,rho(1,j).gt.rha(1,j)
3268 continue
3269 continue
3270 c
3271 if (part) call partin (xcor,ycor,nx,ny,y0)
3272 c
3273 set time at start of calculation.
3274 time=0.0
3275 if (lnuc) time=t0/rscale
3276 go to 480
3277 c
3278 read lanl data for a 1 megaton blast.
3279 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),vic(k),ein(k),den(k),pre(k)
3287 radd=rad(k)*rscale
3288 call ktncu1 (tkt,radd,pre(k),den(k),ein(k),vic(k),gma(k))
3289 ein(k)=pre(k)/(gma(k)-1.0)*den(k)
3290 continue
3291 close (unit=7,status='keep')
3292 c
3293 recalculate gamma and energy using equation of state for air.
3294 do 350 k=1,nrp
3295 ein(k)=ein(k)*den(k)
3296 kount=0
3297 if (kount.gt.40) stop 8888
3298 call eospl (den,ein,nrp,gma,ps)
3299 do 370 k=1,nrp
3300 ein(k)=pre(k)/(gma(k)-1.0)
3301 do 380 k=1,nrp
3302 if (abs(pre(k)-ps(k)).gt.1.e-8*pre(k)) go to 360
3303 c
3304 store data in terms of momentum and total energy.
3305 do 390 k=1,nrp
3306 ein(k)=ein(k)+0.5*vic(k)*vic(k)*den(k)
3307 vic(k)=den(k)*vic(k)
3308 do 400 k=2,nrp
3309 pre(k)=1.0/(rad(k)-rad(k-1))
3310 c
3311 do 470 j=1,mny
3312 if (abs(dely(j)).gt.rad(nrp)) go to 470
3313 do 410 i=1,mnx
3314 rvr(i)=sqrt(scrh(i)+sgrv(j))
3315 c
3316 pick up variables from 2D arrays.
3317 do 420 i=1,mnx
3318 rvx0(i)=rvx(i,j)
3319 rvy0(i)=rvy(i,j)
3320 rho0(i)=rho(i,j)
3321 erg0(i)=erg(i,j).rho0(i)*gp00(j+1)
3322 c
3323 interpolate the blast values onto the grid.
3324 do 450 i=1,mnx
3325 if (rvr(i).gt.rad(nrp)) go to 450
3326 kp=1
3327
3328

```

```

3329 do 430 k=2,nrp
3330 if (rvr(1).lt.rad(k-1).or.rvr(1).gt.rad(k)) go to 430
3331 kp=k
3332 go to 440
3333 continue
3334 fact=pre(kp)*(rvr(1)-rad(kp-1))
3335 rho(1)=den(kp-1)+(den(kp)-den(kp-1))*fact
3336 rvx(1)=v1c(kp-1)+(v1c(kp)-v1c(kp-1))*fact
3337 rvy(1)=rvx0(1)+dely(j)/rvr(1)
3338 rvx(1)=rvx0(1)+delx(1)/rvr(1)
3339 ergo(1)=eln(kp-1)+(eln(kp)-eln(kp-1))*fact
3340 continue
3341
3342 C
3343 C
3344 C
3345 C
3346 C
3347 C
3348 C
3349 C
3350 C
3351 C
3352 C
3353 C
3354 C
3355 C
3356 C
3357 C
3358 C
3359 C
3360 C
3361 C
3362 C
3363 C
3364 C
3365 C
3366 C
3367 C
3368 C
3369 C
3370 C
3371 C
3372 C
3373 C
3374 C
3375 C
3376 C
3377 C
3378 C
3379 C
3380 C
3381 C
3382 C
3383 C
3384 C
3385 C
3386 C
3387 C
3388 C
3389 C
3390 C
3391 C
3392 C

```

do 430 k=2,nrp
if (rvr(1).lt.rad(k-1).or.rvr(1).gt.rad(k)) go to 430
kp=k
go to 440
continue
fact=pre(kp)*(rvr(1)-rad(kp-1))
rho(1)=den(kp-1)+(den(kp)-den(kp-1))*fact
rvx(1)=v1c(kp-1)+(v1c(kp)-v1c(kp-1))*fact
rvy(1)=rvx0(1)+dely(j)/rvr(1)
rvx(1)=rvx0(1)+delx(1)/rvr(1)
ergo(1)=eln(kp-1)+(eln(kp)-eln(kp-1))*fact
continue

refill 2D arrays.
do 460 i=1,mnx
rho(i,j)=rho0(i)
rvx(i,j)=rvx0(i)
rvy(i,j)=rvy0(i)
erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
continue

calculate fraction of eblast deposited in the grid.
continue
call ngridd (xcor,nxp,alpha)
call ogridd (nxp)
call ngridd (xcor,nxp,alpha)
do 490 j=1,mny
call consd (erg(i,j),nx,erg0(j))
continue
call ngridd (ycor,nyp,1)
call ogridd (nyp)
call ngridd (ycor,nyp,1)
call consd (erg0,ny,dpsum)
fracn=(dpsum-ambsum)/(eblast*4.2e19)
write (6,520) ambsum,dpsum,fracn
return

500 format (e12.6,16)
510 format (5(1x,e14.8))
520 format (1x,'energy in ambient mesh=',e12.5,' ergs'/1x,'energy afte
r deposition',e12.5,' ergs'/1x,'fraction of eblast deposited (nuc
2lear case only)',e12.5//)
end

subroutine cjdet (x,rho,u,e,p,gamma)

evaluation of the cj detonation flow field (spherical)
x = r/rc (radius/charge radius)
rho = density (g/cc)
u = velocity (cm/sec)
e = internal energy (ergs/g)
p = pressure (dynes/cm**2)

common /jw1/ r0, pcj, e0cj, gcj, a, beos, c, r1, r2, w
dimension xx(56), rhorn(56), uun(56), een(56)
data xx /0.00,0.46,0.47,0.48,0.49,0.50,0.51,0.52,0.53,0.54,0.55
,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.
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,
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
3,0.94,0.95,0.96,0.97,0.98,0.99,1.00/
data rhorn /0.58336908,0.58329052,0.58333359,0.58547372,0.5877
1509,0.59020287,0.59297010,0.59578818,0.59884679,0.60206354,0.6
0542828,0.60893458,0.61257368,0.61633992,0.62022847,0.62423760,
0.62836248,0.63260019,0.63675151,0.64141273,0.64598429,0.650666

```

3393 4 24.0.65545791.0.66036117.0.66537666.0.67050570.0.67575240.0.681
3394 5 11676.0.68660545.0.69221991.0.69796503.0.70384878.0.70987409.0.
3395 6 71605092.0.72238619.0.72888911.0.73557264.0.74244702.0.74952847
3396 7 0.75683498.0.76438427.0.77220094.0.78031456.0.78875953.0.79757
3397 8 631.0.80681324.0.81654400.0.82684541.0.83783180.0.84965658.0.86
3398 9 254716.0.87683612.0.89311200.0.91249108.0.93788856.1.00000000/
3399 data uun / 0.0000000.0.0000000.0.00482083.0.01035851.0.0166617
3400 1 4.0.02356024.0.03094239.0.03874840.0.04691900.0.05541681.0.0642
3401 2 1171.0.07328302.0.08260766.0.09216960.0.10195599.0.11195995.0.1
3402 3 2217087.0.13258135.0.14319155.0.15399441.0.16499040.0.17617987.
3403 4 0.18756266.0.19914278.0.21092318.0.22290809.0.23510681.0.247521
3404 5 95.0.26016891.0.27305350.0.28618804.0.29959211.0.31327447.0.327
3405 6 26002.0.34156698.0.35621879.0.37124780.0.38668105.0.40255913.0.
3406 7 41892713.0.43583086.0.45333302.0.47150749.0.49044237.0.51024151
3407 8 0.53102863.0.55299151.0.57632965.0.60133791.0.62841576.0.65815
3408 9 610.0.69143391.0.72980273.0.77623236.0.83853370.1.00000000/
3409 data een / 0.39426151.0.39416990.0.39445356.0.39672121.0.3993476
3410 1 9.0.40227160.0.40545160.0.40886858.0.41250038.0.41633454.0.4203
3411 2 6077.0.42457420.0.42896569.0.43353057.0.43826503.0.44316903.0.4
3412 3 4823870.0.45347190.0.45887235.0.46443692.0.47016850.0.47606906.
3413 4 0.48213992.0.48838571.0.49480942.0.50141495.0.50821042.0.515197
3414 5 40.0.52238870.0.52978772.0.53740394.0.54525214.0.55333781.0.561
3415 6 67936.0.57028919.0.57918406.0.58838654.0.59791517.0.60779816.0.
3416 7 61808703.0.62875247.0.63989770.0.65155333.0.66377920.0.67664534
3417 8 0.69023508.0.70467544.0.72009885.0.73670238.0.75475061.0.77467
3418 9 543.0.79692936.0.82264954.0.85371912.0.89516944.1.00000000/
3419 c
3420 c r0 = ambient charge density (g/cc)
3421 c pcj = cj pressure (megabars)
3422 c dcj = cj detonation velocity (cm/microsecond)
3423 c e0cj = cj energy (mbars-cc/cc)
3424 c gcj.a.beos.c.r1.r2.w= jwl eos prameters
3425 c -----
3426 c calculate cj jump conditions
3427 g=gcj
3428 wn=dcj*.1.0e+06
3429 q=e0cj/r0*.1.0e+12
3430 rhon=r0*(g+1.0)/g
3431 pn=r0*wn*wn/(g+1.0)
3432 en=q+.5*wn*wn/(g+1.0)**2)
3433 un=wn/(g+1.0)
3434 if (x.ge.0.0.and.x.le.1.0) go to 10
3435 rho=1.229e-3
3436 u=0
3437 e=7.542e6
3438 go to 20
3439 c
3440 c evaluate self-similar cj detonation flow field
3441 rho=rhon*terpl(x,xx,rhorhn,56.0,0.0,0.3,rho,0)
3442 u=un*terpl(x,xx,uun,56.0,0.0,0.3hvel,0)
3443 e=en*terpl(x,xx,een,56.0,0.0,0.6henergy,0)
3444 call eos2 (rho,e,1,gamma,p)
3445 return
3446 end
3447 function terpl (x,a,b,n,cl,c2,z,nj)
3448 c -----
3449 c linear interpolation subprogram
3450 c input
3451 c x abscissa input
3452 c a abscissa breakpoints, ascending order
3453 c b ordinate breakpoints
3454 c n no. a,b data points
3455 c cl abscissa data increments (mode 3)
3456 c c2 abscissa data starting pt (mode 3)

```

```

3457 C      z      data overflow message. blank gives no message.
3458 C      n)      interpolation integer for mode 2 operation
3459 C      output
3460 C      terpl linear interpolation result
3461 C      three modes of operation
3462 C      1 normal. x.a.b known. nj unknown. do loop search reqd.
3463 C      2 x.a.b.nj known. no search reqd.
3464 C      3 quick interpolation scheme for equally spaced abscissa data
3465 C      calls none
3466 C-----
3467      dimension a(n), b(n)
3468      nm1=n-1
3469      j=c1*x+c2
3470      j=j*nj
3471      if (j.gt.nm1) j=nm1
3472      if (j.gt.0) go to 20
3473      do 10 i=1,nm1
3474      j=1
3475      if (x.le.a(i+1)) go to 20
3476      continue
3477      print 40, z,x
3478      terpl=a(j+1)-a(j)
3479      if (terpl.eq.0.) go to 30
3480      terpl=b(j)+b(j+1)-b(j)*(x-a(j))/terpl
3481      return
3482      print 50, z
3483      terpl=alog(0.)
3484      stop
3485 C
3486 C      40 format (40h range exceeded during interpolation of .a6,12h data.
3487 C      1x = .e15.7)
3488 C      50 format (10x,12h)terpl error .a6,24h data. zero denominator./10x,30h
3489 C      error trace by alog(0.) below.)
3490 C      end
3491 C      subroutine tlayer (rho,rvx,rvy,erg,rhomin,sum)
3492 C-----
3493 C      thermal layer model using sound speed data.
3494 C-----
3495 C      parameter (mny=150, mny=200, mxy=202)
3496 C      parameter (jthm=1)
3497 C
3498 C      real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
3499 C      common /grids/ xcor, xcoro, ycor, ycoro, unit
3500 C
3501 C      real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
3502 C      real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
3503 C      real rho0(mxy), rhb0(mxy), rhc0(mxy), vhm(mxy)
3504 C      real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3505 C      real rhoh(mxy), rvzh(mxy), rvyh(mxy), ergv(mxy)
3506 C      real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
3507 C      real rmin(mxy), rmax(mxy), rvr(mxy)
3508 C      common /scrh/ rhom, pres, vel, uh, ergk, srvx, srvy, serg,
3509 C      rho0, rhb0, rhc0, vh, rhoh, rvx0, rvy0, erg0, rhoh, rvzh, rvyh
3510 C      , ergv, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3511 C
3512 C      real gam0(mxy), gamn(mxy), eint(mxy), cst(mxy)
3513 C      real xkt(mxy), ratio(mxy), pscr(mxy)
3514 C      equivalence (gam0,dela), (gamn,dely)
3515 C      equivalence (eint,vel), (cst,vh), (pscr,erght)
3516 C      equivalence (xkt,rhb0), (ratio,rhb0)
3517 C
3518 C      real csamb(jthm)
3519 C      real rho(mnx,mny), rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
3520 C

```

```

3521 real pr00(mxy), rho0(mxy), er00(mxy), gp00(mxy), gavy(mxy)
3522 real fsky(mxy), rhtl(mnx,jthm)
3523 common /grav/ pr00, rho0, er00, gp00, gavy, fsky, rhtl
3524 i, jthrm
3525 c
3526 c calculate sound speed in ambient atmosphere.
3527 mnx=mnx-1
3528 do 10 j=1,jthrm
3529 rho=rho(mnx,j)
3530 rvx=rvx(mnx,j)
3531 rvy=rvy(mnx,j)
3532 erg=erg(mnx,j)
3533 c
3534 eint=erg1-0.5*(rvx1+rvx1+rvy1+rvy1)/rho1-rho1*gp00(j+1)
3535 call eospl(rho1,eint,i,j,gam1,pres1)
3536 csamb(j)=sqrt(gam1*pres1/rho1)
3537 continue
3538 sum=0.0
3539 c
3540 c set up thermal layer.
3541 do 80 j=1,jthrm
3542 do 20 i=1,mnx
3543 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 eint(i)=erg0(i)-ergk(i)
3552 call eospl(rho0,eint,mnx,gamn,pres)
3553 icount=0
3554 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 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 continue
3564 do 70 i=1,mnx
3565 erg0(i)=eint(i)+ergk(i)
3566 sum=sum+(erg0(i)-erg0(i,j))
3567 rho(i,j)=amax1(rho0(i),rho0(i,n))
3568 rhtl(i,j)=rho(i,j)
3569 erg(i,j)=erg0(i)+rho0(i)*gp00(j+1)
3570 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 cs(i)=cvmgt(csthm,csamb(j),xkt(i)) je,xmin,and,xkt(i),le,xmax)
3584 return

```



```

3585      end
3586      subroutine griddx (nreg,nxp,xval,scrh)
3587      -----
3588      c calculates x - coordinate values for variable grid size.
3589      c -----
3590      parameter (mxy=202)
3591      c
3592      real xband(5), dxr(4), yband(5), dyr(2)
3593      common /gradcon/ ndx, nsmth, mdy, alt, nahed, xfine, hx, hy,
3594      xband, yband, dxr, dyr, xleft
3595      c
3596      real lband(11), xval(nxp), scrh(mxy)
3597      c
3598      c set up the region boundary indices.
3599      nx=nxp-1
3600      lband(1)=1
3601      nreg=nreg+1
3602      do 10 ireg=1,nreg
3603      lband(ireg+1)=lband(ireg)+(xband(ireg+1)-xband(ireg))/dxr(ireg)
3604      lband(nreg+1)=float(nxp)
3605      c
3606      c now the values of xval are interpolated in the region tables.
3607      ireg=1
3608      do 30 i=1,nxp
3609      do 20 ireg=1,nreg
3610      scrh(ireg)=cvmgf(float(ireg),float(nreg),float(i),le,lband
3611      (ireg+1))
3612      ireg=int(scrh(lsmn(nreg),scrh,i)))
3613      c
3614      c 1 is in the region lband(ireg) to lband(ireg+1).
3615      xval(i)=xband(ireg)+(xband(ireg+1)-xband(ireg))*((float(i)-lband
3616      (ireg))/(lband(ireg+1)-lband(ireg)+1.0e 5))
3617      30 continue
3618      c
3619      c smooth the discontinuities in grid size.
3620      do 60 ismth=1,nsmth
3621      do 40 i=2,nx
3622      scrh(i)=0.25*(xval(i+1)+2.0*xval(i)+xval(i-1))
3623      do 50 i=2,nx
3624      xval(i)=scrh(i)
3625      60 continue
3626      c
3627      return
3628      end
3629      subroutine xgrid (jtherm)
3630      c -----
3631      c calculates interface locations for moving grid problems.
3632      c -----
3633      parameter (mxx=150, mny=200, mxy=202)
3634      parameter (mxxx=1, mnyy=1)
3635      integer alpha
3636      logical lmgrid, lstat, lnucl, ltherm, lgrav, leos
3637      real rha(mxx,mny), rhb(mxxx,mny), rho(mxxx,mnyy)
3638      real tem(mxxx,mnyy), scf(mxxx,mny), rho(mxx,mny)
3639      real rvx(mxx,mny), rvy(mxx,mny), erg(mxx,mny)
3640      common nx, ny, nxp, nyp, nspec, ldump, ldiag, lmgrid
3641      common lstat, lnucl, ltherm, lgrav, leos
3642      common alpha, gamma, gml, dt, dx, dy, rhomin
3643      common tem, scf, rho, rvx, rvy, erg, rha, rhb, rhc
3644      c
3645      real xcorf(mxy), ycorf(mxy), ycor(mxy), ycorf(mxy), unit(mxy)
3646      common /grids/ xcor, ycor, ycorf, ycorf, unit
3647      c
3648      real xband(5), dxr(4), yband(5), dyr(2)

```

```

3649 common /grdcon/ ndx, nsmth, mdy, all, nashed, xfine, hx, hy,
3650 xbnd, ybnd, dxr, dyr, xleft
3651 c
3652 real rhom(mxy), pres(mxy), vel(mxy), uh(mxy)
3653 real ergk(mxy), srvx(mxy), srvy(mxy), seig(mxy)
3654 real rho0(mxy), rhb0(mxy), rhc0(mxy), vhl(mxy)
3655 real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3656 real rho0(mxy), rvxh(mxy), rvyh(mxy), erg1(mxy)
3657 real delx(mxy), dely(mxy), scrh(mxy), sgrv(mxy), sgam(mxy)
3658 real rmin(mxy), rmax(mxy), rvr(mxy)
3659 common /scrch/ rhom, pres, vel, uh, ergk, srvx, srvy, seig,
3660 rho0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
3661 c
3662 . ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3663 c
3664 save old x - coordinate values.
3665 do 10 i=1,nxp
3666 xcoro(i)=xcor(i)
3667 c
3668 determine shock-ambient interface.
3669 ihld=1
3670 jhld=0
3671 do 30 j=itherm*2,mny
3672 do 20 i=1,mnx
3673 scrh(i)=cvmgt(float(i),0.0,erg(i,j),gt.1,1+erg(mnx,j))
3674 klim=ismax(mnx,scrh,1)
3675 ihld=max0(ihld,klim)
3676 jhld=cvmgt(j,jhld,ihld,eq.klim)
3677 continue
3678 xhld=xcor(ihld)+20.*dx
3679 xshk0=xbnd(3)-float(nashed)*dx
3680 if (xhld.lt.xshk0) return
3681 xtry=amini(xshk0+dx,xhld)
3682 xbnd(3)=amini(xtry+float(nashed)*dx,xbnd(5))
3683 xbnd(4)=amini(xbnd(3)+float(nx-ndx)*hx,xbnd(5))
3684 xbnd(2)=amax1(xbnd(3)-float(ndx)*dx,xbnd(1))
3685 c
3686 compute the new grid as it slides along with the shock.
3687 call griddx (4,nxp,xcor,scrh)
3688 if (therm1) call rhlyr
3689 return
3690 c
3691 entry ygrid
3692 do 40 j=1,nyp
3693 ycoro(j)=ycoro(j)
3694 c
3695 entry xgrid0
3696 xcor(1)=0.0
3697 if (lmgrid) go to 60
3698 do 50 i=2,nxp
3699 xcor(i)=xcor(i-1)+dx
3700 go to 70
3701 c
3702 standard moving grid for hob problems. shock edge falls within
3703 fine grid.
3704 c
3705 continue
3706 nhx=int((xfine-float(ndx)*dx)/hx)+1
3707 nhx=max0(0,nhx)
3708 xbnd(1)=xleft
3709 xbnd(2)=xbnd(1)+float(nhx)*hx
3710 xbnd(3)=xbnd(2)+float(ndx)*dx
3711 xbnd(4)=xbnd(1)+float(ndx)*dx+float(nx-ndx)*hx
3712 xbnd(5)=xbnd(4)
3713 dxr(1)=hx

```

```

3713 dxr(2)=dx
3714 dxr(3)=hx
3715 dxr(4)=dx
3716 call gridax (4,nxp,xcor,scrh)
3717 do 80 i=1,nxp
3718   xcoro(i)=xcor(i)
3719   return
3720 C
3721   entry ygrido
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     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     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   ycor(j)=ycor(j-1)+dylrg
3745   130   do 140 j=1,nyp
3746     ycoro(j)=ycor(j)
3747   return
3748
3749   end
3750 C   subroutine distm(lstep,time)
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 (mxx=150, mny=200, mxy=202)
3756   parameter (mxxx=1, mnyy=1)
3757 C
3758   integer lmgrid, lstat, lnuc, therm1, lgrav, leos
3759   real rho(mxx,mny), rho(mxxx,mny), rho(mnxx,mny), rho(mnxx,mny)
3760   real temp(mnxx,mny), sci(mnxx,mny), rho(mnxx,mny)
3761   real rvx(mnxx,mny), rvy(mnxx,mny), erg(mnxx,mny)
3762   common nx, ny, nyp, nyp, nspec, idump, idtag, lmgrid
3763   common lstat, lnuc, therm1, lgrav, leos
3764   common alpha, gamma, gmi, dt, dx, dy, rhomin
3765   common tem, sci, rho, rvx, rvy, erg, rha, rhb, rhc
3766
3767 C   real xcor(mxy), xcoro(mxy), ycor(mxy), ycoro(mxy), unit(mxy)
3768   common /grids/ xcor, xcoro, ycor, ycoro, unit
3769 C
3770 C   real rhom(mxy), pres(mxy), vel(mxy), ub(mxy)
3771   real ergk(mxy), srvx(mxy), srvy(mxy), serg(mxy)
3772   real rho0(mxy), rho0(mxy), rho0(mxy), vhi(mxy)
3773   real rho0(mxy), rvx0(mxy), rvy0(mxy), erg0(mxy)
3774   real dnew(mxy)
3775   real rhob(mxy), rvxb(mxy), rvyb(mxy), erqb(mxy)
3776

```

```

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, ub, ergk, srvx, srvy, serg,
3780 rho0, rhb0, rhc0, vh, rho0, rvx0, rvy0, erg0, rhob, rvxb, rvyb
3781 ., ergh, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3782 C
3783 real pr00(mxy), rho0(mxy), er00(mxy), gp00(mxy), gravity(mxy)
3784 parameter (jthm=1)
3785 real fsky(mxy), rhtl(mnx,jthm)
3786 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
3787 ., jthrm
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 rdot(mxy), tauw(mxy), visrat(mxy), phil(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   scrh(i)=cvmgt(float(i),0.0,rho(i),1).gt.1.2*rho(mnx,1))
3802 C   ishkr=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 C   inject=mnx
3807 C   if (inject.eq.1) return
3808 C
3809 C mine's 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)=B*cft/cfo
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   cfo = 2tauw/rhoa/uair**2
3824 C   next tau 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 C do 20 i=1,mnx
3832 C   rhom(i)=1.0/rho(i,1)
3833 C   rhoth(i)=rho(i,1)
3834 C   rair(i)=rho(i,1)/rha(i,1)
3835 C   vel(i) = rvx(i,1)*rhom(i)
3836 C   qnew(i)=(deltas(i)-delta2(i))*rair(i)*vel(i)
3837 C   ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))*rhom(i)
3838 C   eint(i)=erg(i,1)*ergk(i)-rhoth(i,1)*gp00(2)
3839 C   call eospl (rhom,eint,inject,sgam,pres)
3840 C

```

```

3841 C      obtain the ratio of viscosities,  $\mu_{\text{min}}/\mu_{\text{ua}}$ 
3842 C
3843 C      do 25 i=1,mnx
3844 C          pres(i)=amax1(pres(i),1.e5)
3845 C          mech(i)=vel(i)/sqrt(sgam(i))*pres(i)/rair(i)
3846 C          visrat(i)=0.524*abs(mach(i))
3847 C      continue
3848 C
3849 C      begin the painful process of integration
3850 C
3851 C          call ngridd(xcoro,nx+1,2)
3852 C          call ogridd(nx+1)
3853 C          call ngridd(xcor,nx+1,2)
3854 C
3855 C      obtain the velocities at cell interfaces
3856 C
3857 C          do 26 l=2,nx
3858 C              uh(l)=(vel(l)+vel(l-1))*0.5
3859 C              uh(l)=0.0
3860 C              uh(nx+1)=uh(nx)
3861 C              call velocd(uh,nx+1,dt,nx+1)
3862 C
3863 C      next do the source terms
3864 C
3865 C          do 1 l=1,mnx
3866 C              rho0(l)=-qnew(l)
3867 C              call sourced(mnx,dt,5,rho0,vel,0.0,0.0)
3868 C              call sourced(mnx,dt,5,delta2,pres,pres(mnx),pres(l))
3869 C              do 2 l=1,mnx
3870 C                  rho0(l)=-theta(l)*vel(l)**2*rair(l)
3871 C                  call sourced(mnx,dt,1,rho0,rho0,0.0,0.0)
3872 C              do 3 l=1,mnx
3873 C                  rho0(l)=qnew(l)*vel(l)
3874 C                  call sourced(mnx,dt,1,rho0,rho0,0.0,0.0)
3875 C              do 4 l=1,mnx
3876 C                  rho0(l)=tauw(l)
3877 C                  call sourced(mnx,dt,3,rho0,rho0,0.0,0.0)
3878 C              call frycct(qnew,rhae,mnx,1.0,0.0,1.0,0.0)
3879 C
3880 C      update the qnew array and compute the boundary layer
3881 C      thicknesses, deltas and delta.
3882 C
3883 C          delty=ycor(2)-ycor(1)
3884 C          do 5 l=1,mnx
3885 C              qnew(l)=rhae(l)
3886 C              if(vel(l).eq.0.0) vel(l)=1.e-30
3887 C              deltas(l)=qnew(l)/rair(l)*vel(l) + delta2(l)
3888 C              delta(l)=amax1(.02*delty,8.*deltas(l))
3889 C          do 6 l=1,mnx
3890 C              mdot(l)=0.0
3891 C              if(vel(l).lt.1.e-4) go to 6
3892 C              te=con3*pres(l)/rair(l)
3893 C              rhor=0.5*(tw/ta+1.0) + 0.11*(sgam(l)-1.0)*mach(l)**2
3894 C              phi(l)=(1.0/rhor)**0.75*visrat(l)**0.25
3895 C              if(delta(l).lt.1.e-4) then
3896 C                  tauw(l)=0.0
3897 C              else
3898 C                  tauw(l)=0.0225*(amu/rair(l)*vel(l)*delta(l)**0.25)*
3899 C                      phi(l)*rair(l)*vel(l)**2
3900 C              endif
3901 C          cfo=2.*tauw(l)/rair(l)/vel(l)**2
3902 C          cft=2.*beta*sign(q-delta/rair(l))*vel(l)**2
3903 C
3904 C

```

```

3905 c solve for B, ln(1+B)=B*cft/cfo
3906 raat=cfo/cft
3907 if(raat.le.1.0) go to 6
3908 B=raat*.15
3909 do 7 k=1,25
3910 a=B/(alog(1.0+B))
3911 B=raat-a*B
3912 continue
3913 mdot(1)=abs(cft*.5*B*(rair(1)*vel(1)))
3914 continue
3915 c
3916 c deposit dust in the flow field behind shock front.
3917 do 30 i=1,inject
3918 if(delta(i).lt.1.e-4) then
3919 dirt(1)=0.0
3920 else
3921 dirt(1)=mdot(1)*ff*dt/delty
3922 endif
3923 dnew(1)=rha(1,1)+dirt(1)
3924 rha(1,1)=cvmgt(dnew(1),rha(1,1),ergk(1).gt.edp0)
3925 rho(1,1)=rair(1)+rha(1,1)
3926 ergo(1)=amax(0.0,eint(1)+1.e7*dirt(1)*con3*pres(1)/rho(1,1))
3927 erg(1,1)=cvmgt(ergo(1)+rho(1,1)*gp00(2)+ergk(1)/rho(1,1)*rho(1,1),erg(1,1),ergk(1).gt.edp0)
3928 continue
3929
3930 if(mod(istep-1,100).eq.0) then
3931 write(6,57) istep, time
3932 format(1x,'dirt for step',15,' and time ',1p12.5)
3933 write(6,56) (dirt(i),i=1,30)
3934 else
3935 endif
3936 format(1x,1p8e12.5)
3937 return
3938 c
3939 entry dust01
3940 c
3941 c initialize dust routine by setting dust density to zero.
3942 do 40 j=1,mny
3943 do 40 i=1,mnx
3944 rha(i,j)=0.0
3945 do 50 l=1,mnx
3946 delta(l)=0.02*(ycor(2)-ycor(1))
3947 deltas(l)=delta(l)/B.0
3948 delta2(l)=0.0
3949 theta(l)=delta(l)*.7/.32
3950 mdot(l)=0.0
3951 tauw(l)=0.0
3952 continue
3953 50
3954 return
3955 enn
3956
3957 c -----
3958 c adds dust so as to model the dust pickup behind the shock front.
3959 c the 50UR model used is based upon the 5*3 model. the dust mass
3960 c added at each step is .08*area*rho(air).
3961 c -----
3962 parameter (mnx=150, mny=200, mxy=202)
3963 parameter (mxxx=1, mnyy=1)
3964 c
3965 integer lmgrid, lstat, linc, therm1, lgrav, leos
3966 real rha(mnx,mny), rhb(mnx,mny), rho(mnx,mny)
3967 real tem(mnx,mny), scf(mnx,mny), rho(mnx,mny)

```

```

3969 real rvx(mnx,mny), rvy(mnx,mny), erg(mnx,mny)
3970 common nx, ny, npx, nyp, nspec, idump, idtag, lmgrid
3971 common lstat, lnuc, therm1, lgrav, leos
3972 common alpha, gamma, gmi, dt, dx, dy, rhomin
3973 common tem, sci, rho, rvx, rvy, erg, rha, rha, rha, rha, rha, rha
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), serng(mxy)
3980 real rha0(mxy), rh00(mxy), rhc0(mxy), vhf(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, serng,
3986 rha0, rhb0, rhc0, vh, rh00, rvx0, rvy0, erg0, rhoh, rvxh, rvyh
3987 ., ergk, delx, dely, scrh, sgrv, sgam, rmin, rmax, rvr
3988 c
3989 real pr00(mxy), rh00(mxy), er00(mxy), gp00(mxy), gravity(mxy)
3990 parameter (jthm=1)
3991 real fsky(mxy), rhtl(mnx,jthm)
3992 common /grav/ pr00, rh00, er00, gp00, gravity, fsky, rhtl
3993 ., jthrm
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   scrh(i)=vmgt(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 c   inject=mnx
4009 c   if (inject.eq.1) return
4010 c
4011 c calculate kinetic energy
4012 c do 20 i=1,inject
4013 c   rhom(i)=1.0/rho(i,1)
4014 c   rair(i)=rho(i,1)-rha(i,1)
4015 c   ergk(i)=0.5*(rvx(i,1)*rvx(i,1)+rvy(i,1)*rvy(i,1))+rhom(i)
4016 c   eint(i)=erg(i,1)-ergk(i,1)*gp00(i,1)
4017 c
4018 c get pressure
4019 c call eospl (rho,eint,inject,sgam,pres)
4020 c
4021 c deposit dust in the flow field behind shock front.
4022 c dy12=(ycor(2)-ycor(1))
4023 c do 30 i=1,inject
4024 c   dirt(i)=0.08*rair(i)*dt*ff*rhom(i)*abs(rvx(i,1))/dy12
4025 c   dnew(i)=rhai(i)*dirt(i)
4026 c   rha(i,1)=vmgt(dnew(i),rha(i,1),ergk(i),gt,edp0)
4027 c   rho(i,1)=rair(i)+rha(i,1)
4028 c   ergd(i)=amax0(0,eint(i,1)+dirt(i,1)*con3*pres(i,1)+rhai(i,1)
4029 c   ergk(i,1)=vmgt(ergd(i),rho(i,1)*gp00(i,1)+ergk(i,1)+rhai(i,1)
4030 c   .1),erg(i,1),ergk(i,1),gt,edp0)
4031 c   continue
4032 c   if(mod(i,step,1,100) .eq.0) then

```

```
4033 write(6,78) istep,time
4034 format('x,'dirt' at step',14,' and time',1pe12.5)
4035 write(6,79) (dirt(i),i=1,30)
4036 format('x,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 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 development. 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	7
1. Numerical Simulations	9
2. Experimental	17
Approach and Test Procedures	17
Test Results	19
3. Comparison Between Numerical Simulations and Experiments	23
4. Summary and Conclusions	27
References	29
Figures and Table Captions	31
Appendix A	83

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 effectiveness of DECA against antitank mines, in two tests M-15 mines were destroyed 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 chosen 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 initial 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 (usually 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 $t = 0.13\text{ms}$, $t = 0.30\text{ms}$, $t = 0.49\text{ms}$, $t = 0.78\text{ms}$ 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 insure 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 $t = 0.25\text{ms}$, $t = 0.50\text{ms}$, $t = 1.0\text{ms}$, after detonation. The maximum impulse achieved in this simulation was $\approx 1500\text{psi} \cdot \text{ms}$. These levels were observed directly under the lines of explosives. In the areas between the lines of Primacord, the impulse is around $450\text{psi} \cdot \text{ms}$. These areas of small impulse are $\approx 20\text{cm}$ wide. The blast wave contributes 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 corresponding to Fig. 1.5a is located under the explosive. That leads to very high pressure 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 · 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 pseudo 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 $4.5\text{m} \times 4\text{m}$. 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 x 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 previous 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 correspondingly, are shown for $t = 0.15\text{ms}$ and $t = 0.35\text{ms}$. It is easy to see in these figures that for that case, the planar front forms almost immediately 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 previous case, and the impulse is 550psi·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 neighboring 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 array 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 suspension 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 sufficient 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 $t = 0.06\text{ms}$, $t = 0.12\text{ms}$, and $t = 0.27\text{ms}$. Better grid resolution allows us to follow details of the evolution of the blast wave for this DECA detonation. At $t = 0.06\text{ms}$ the blast wave is formed by cylindrical detonations from the parallel line charges, and collisions 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 $t = 0.12\text{ms}$ the reflection from the ground begins. The reflected blast wave is stronger between the charges, because of collision of the blast waves from two parallel sources. At $t = 0.27\text{ms}$ 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 $\approx 800\text{psi} \cdot \text{ms}$. These levels of blast loads should be sufficient 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 $\log(\text{Pressure})$ vs. $\log(\text{Impulse})$ graph. DECA charges for this study were about $4\text{m} \times 4\text{m}$ 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 advantage 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 from 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 air-blast 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 x 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/m² (0.53lb/ft²) to 3.9kg/m² (0.80lb/ft²). 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 \times 2.5m. However, the total explosive surface density (2.6kg/m²) 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/m² compared with 2.6kg/m² 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 S530A) 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 2 μ sec 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 30 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 396kPa, 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 $2.5\text{m} \times 2.25\text{m}$. 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 2860kPa (415psi) for the Iremite array and 4860 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 (11.8in.) in diameter and about 15cm (6in.) high. They were placed in the center of the Primacord array, which was 25cm above 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 severely 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 2.8a. 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

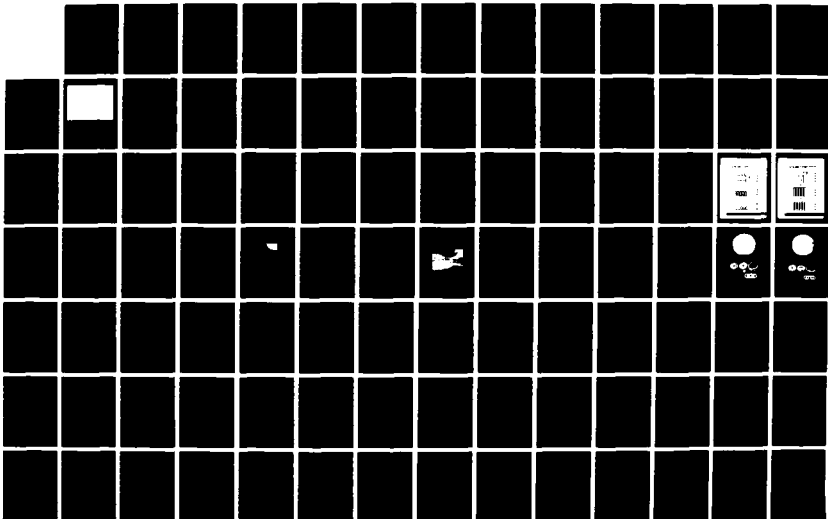
NUMERICAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA H A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

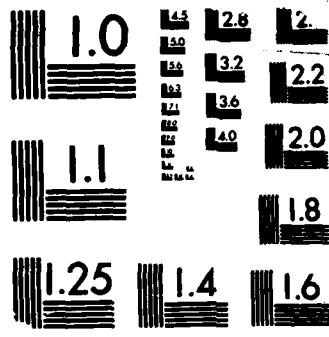
2/3

UNCLASSIFIED

F/G 19/11

NN





MICROCOPY RESOLUTION TEST CHART
NBS-1963-A

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 constraints 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 measurements 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 pattern 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 disappears. 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 asymmetric 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 representing 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 are located symmetrically. 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 relatively low detonation velocity of 3500m/sec. At that velocity, the detonation wave will reach the center of the DECA array in ≈ 0.57 ms. That is comparable to ≈ 0.42 ms 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 horizontal 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 running 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 x 2m Primacord array the obliqueness of the blast front is smaller and the described effect would be negligible. 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 specially important for design of the initiation schemes for the DECA charges. Test 5 was an exact repeat of test 4 and results of pressure measurement 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 G8 in test 9 and numerically 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 rarefaction 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 G8 and G7 are not located in symmetrical positions. As is shown in Fig. 3.4, gauge G8 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 comparison. 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 collisions between the blast waves from adjacent parallel lines, equidistant from the point of measurement.

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 collisions 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). Measurements on the ground will add very important information about peculiarities 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 competitive 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

destroyed 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 Algorithm 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 $t = 0.25\text{msec}$.

Figure 1.4b Impulse on the ground at $t = 0.50\text{msec}$.

Figure 1.4c Impulse on the ground at $t = 1.00\text{msec}$.

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. $X=5$, $Y=0$, $Z=0$

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. $X=20, Y=0, Z=0$

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

Figure 2.1 Perspective of experimental configuration.

Figure 2.2 Explosive array and instrumentation poles.

Figure 2.3 Experimental setup showing instrumentation poles.

Figure 2.4 Experimental configuration for antitank mine tests.

Figure 2.5 Comparison of tests 4 and 5.

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

Figure 2.7 Damage to antitank mine in test 9.

Figure 2.8 Damage to antitank mine in test 10.

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.

a . Experimental data

b . Computer simulation.

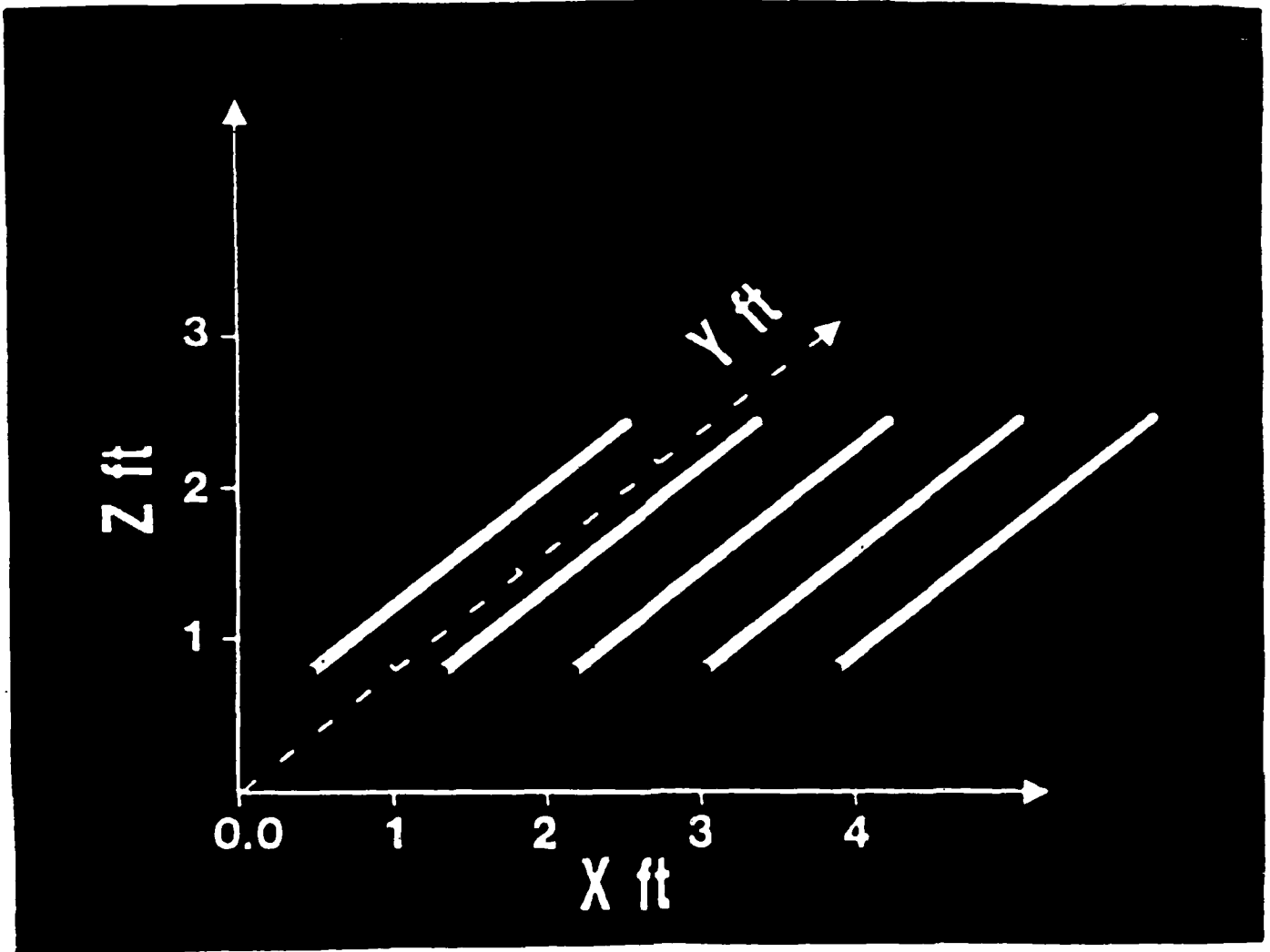


Figure 1.1 Schematic layout of a typical simulation.

DECA BLAST ON THE GROUND WEIGHT=100LB
X-Z PLANE, Y=0 CM, TIME = 0.13MSEC

PRESSURE

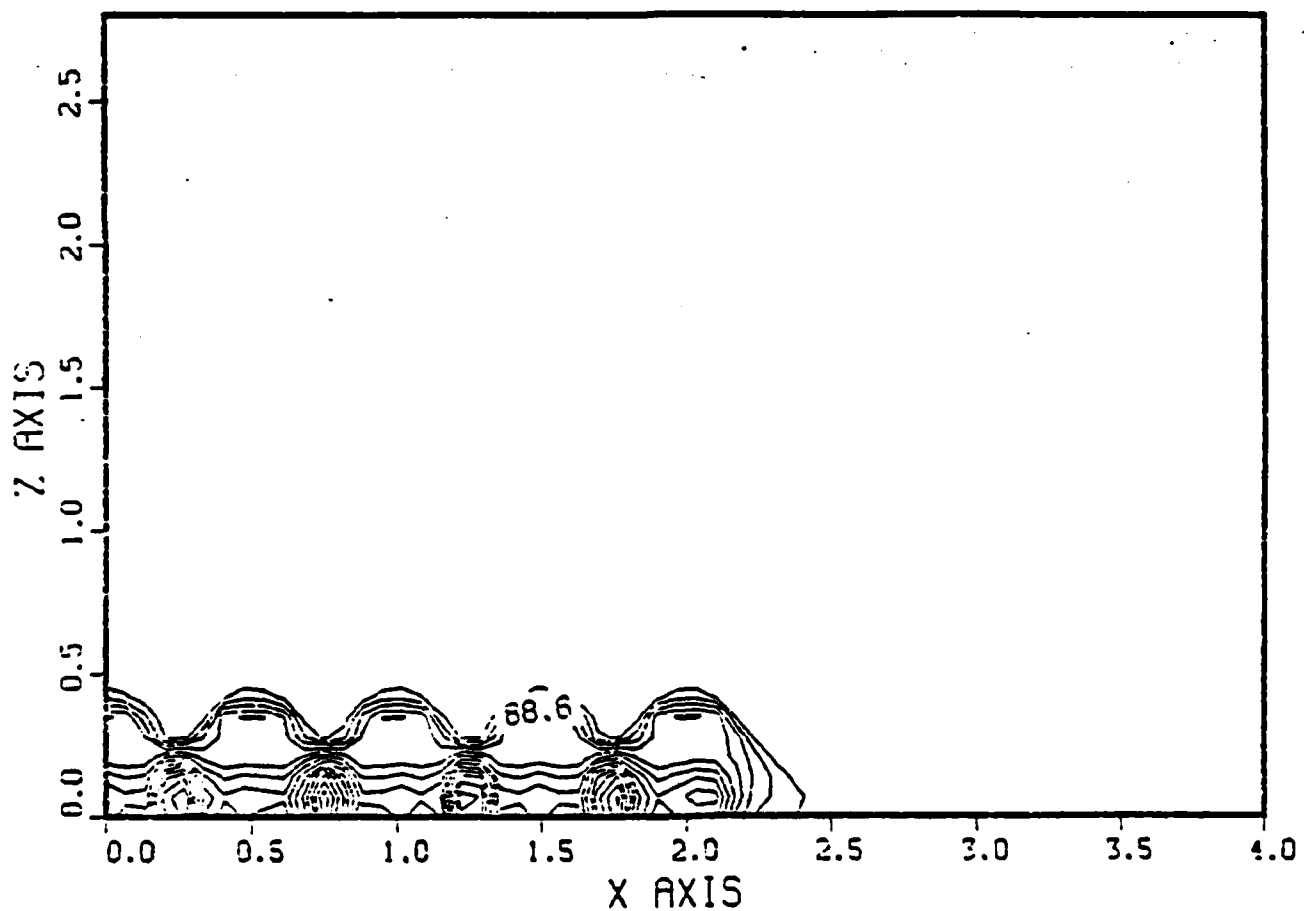


Figure 1.2a Pressure contours for a 9 lines 4m x 4m DECA array test. X-Z cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Z PLANE, Y=0 CM, TIME = 0.30MSEC

PRESSURE

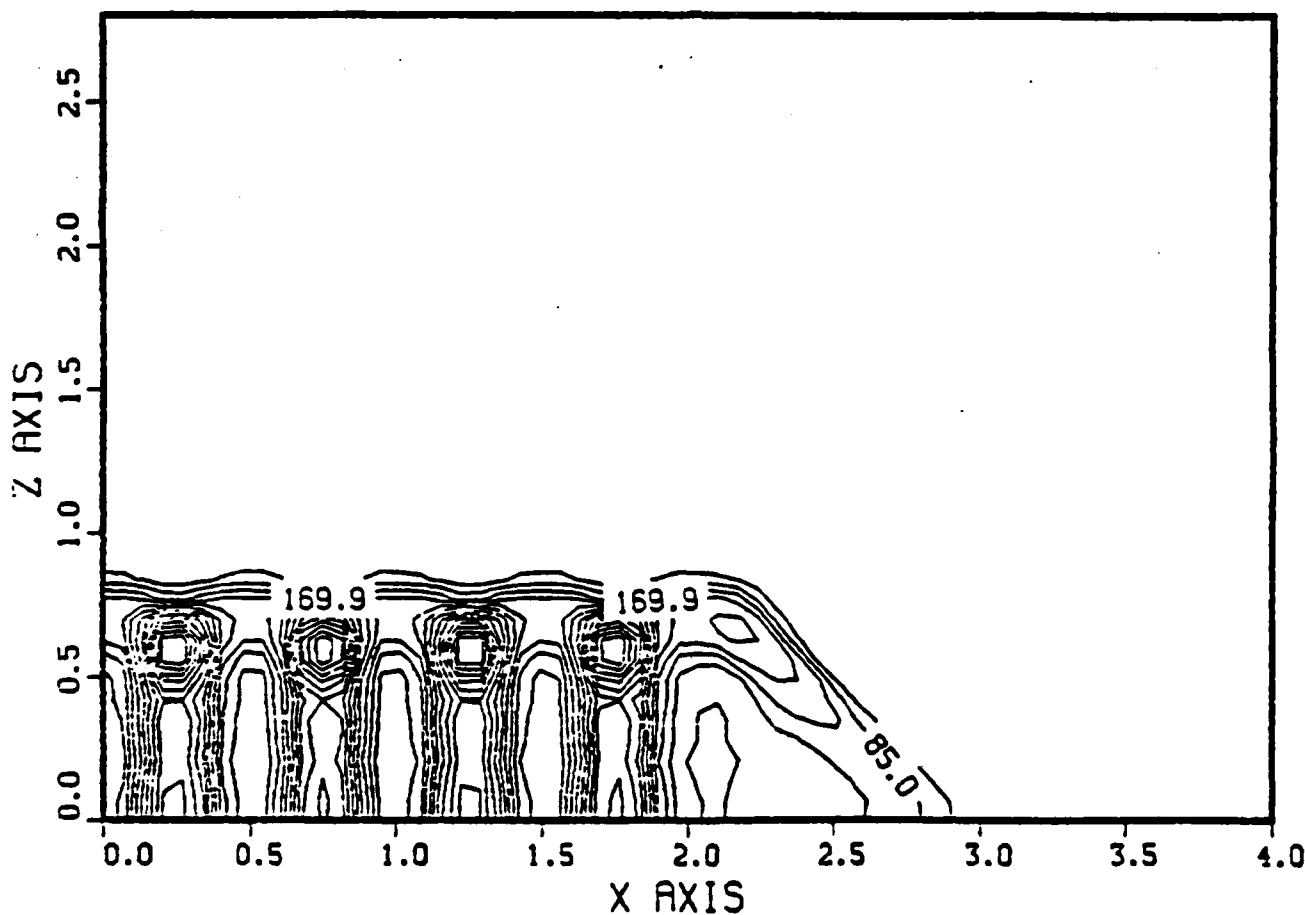


Figure 1.2b Pressure contours for a 9 lines 4m x 4m DECA array test. X-Z cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Z PLANE, Y=0 CM, TIME = 0.49MSEC

PRESSURE

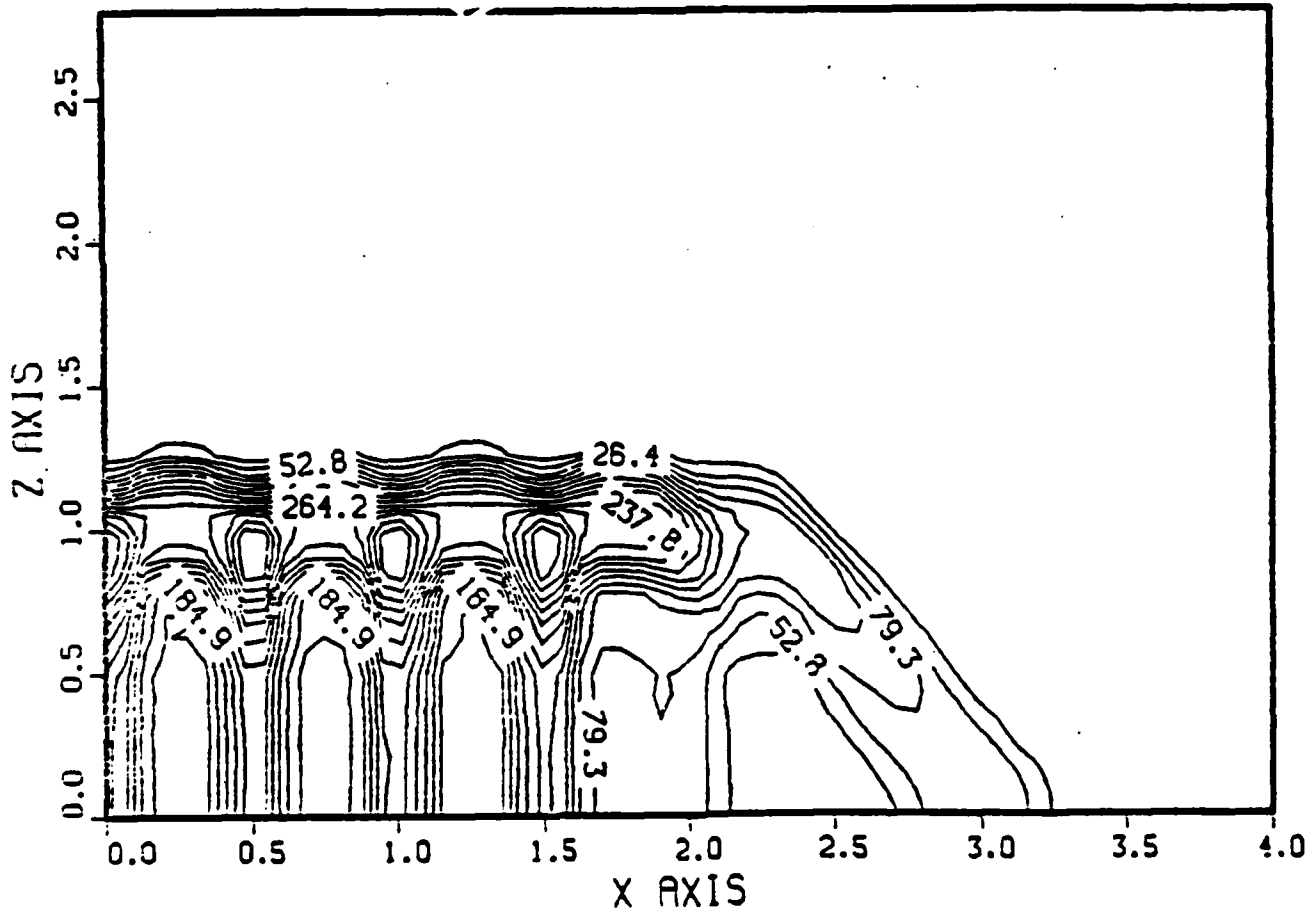


Figure 1.2c Pressure contours for a 9 lines 4m x 4m DECA array test. X-Z cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Z PLANE, Y=0 CM, TIME = 0.72 MSEC

PRESSURE

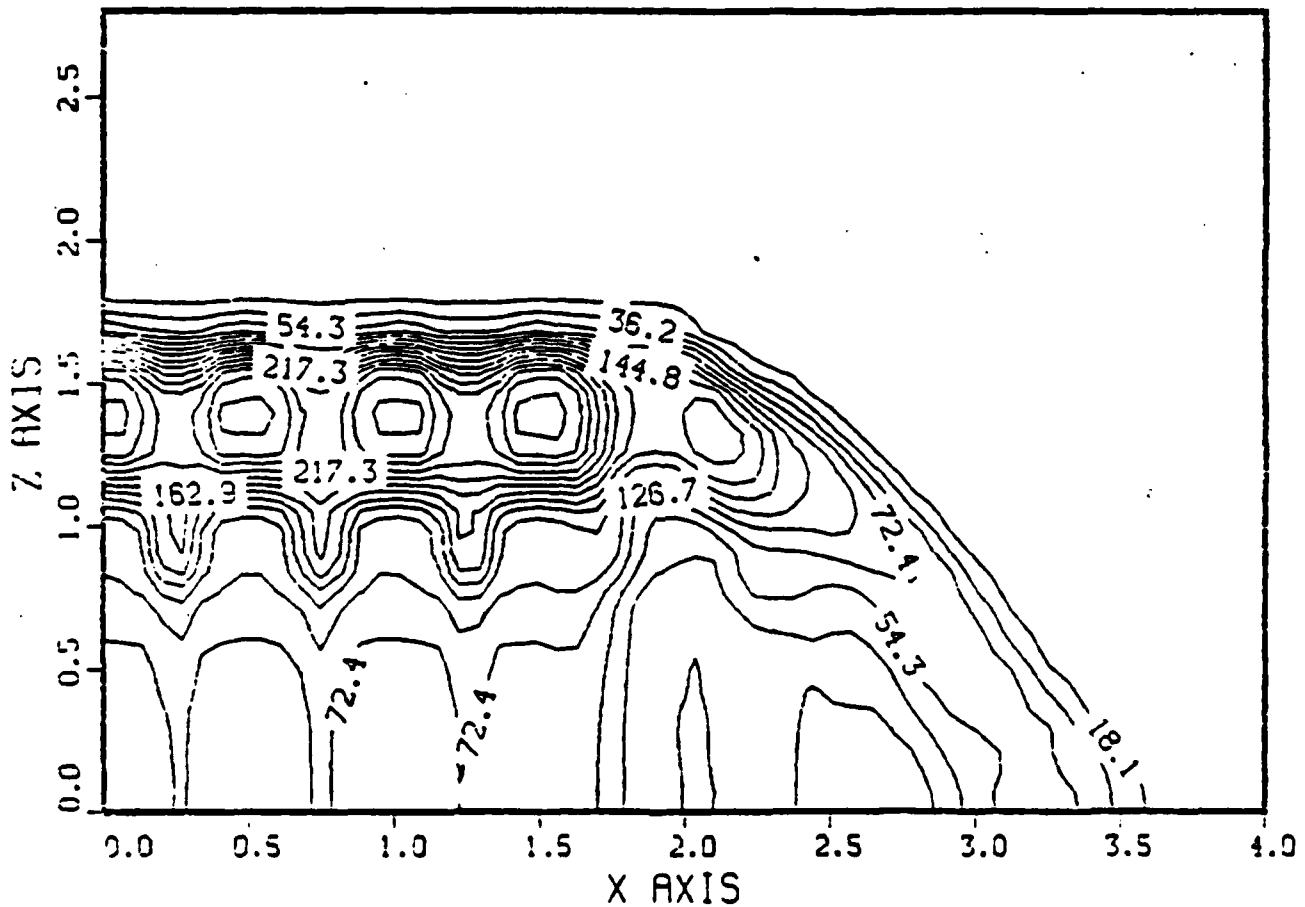


Figure 1.2d Pressure contours for a 9 lines 4m x 4m DECA array test. X-Z cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Y PLANE, Z=0 CM, TIME = 0.13MSEC
PRESSURE

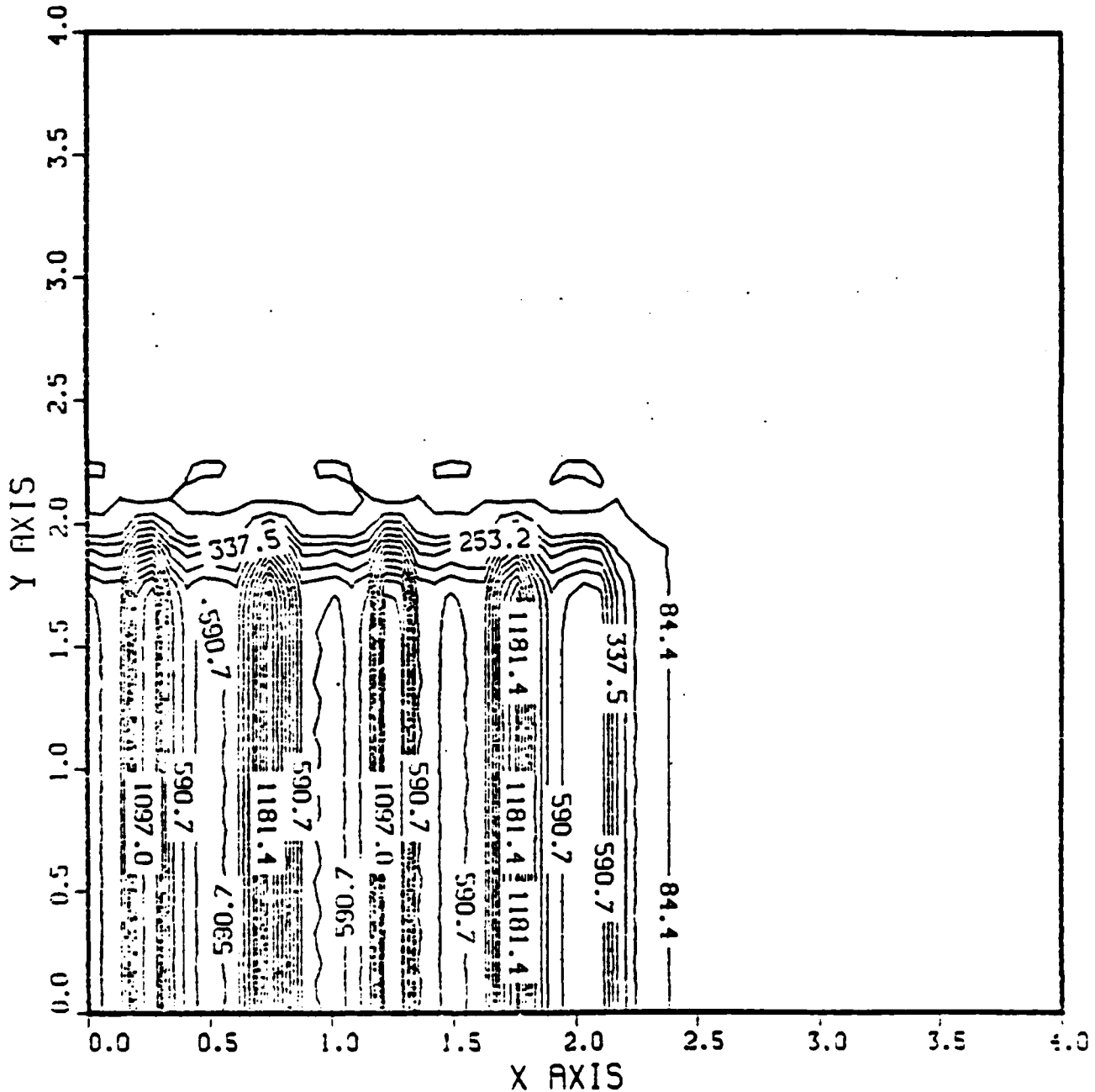


Figure 1.3a Pressure contours for a 9 lines 4m x 4m DECA array test. X-Y cross section.

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Y PLANE, Z=0 CM, TIME = 0.30MSEC
PRESSURE

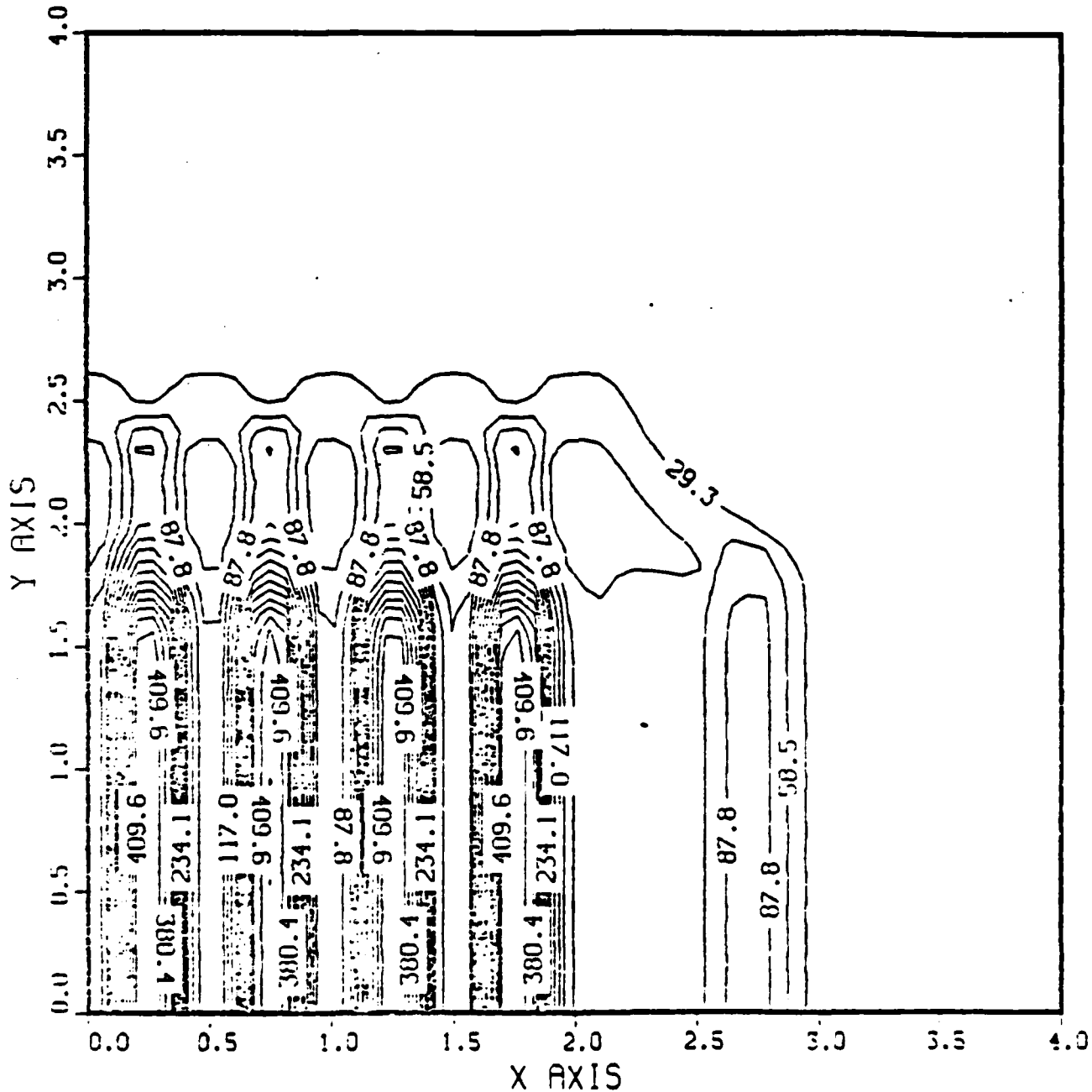


Figure 1.3b Pressure contours for a 9 lines 4m x 4m DECA array test. X-Y cross section.

DECA BLAST ON THE GROUND WEIGHT=100LB

X-Y PLANE, Z=0 CM, TIME = 0.49MSEC

PRESSURE

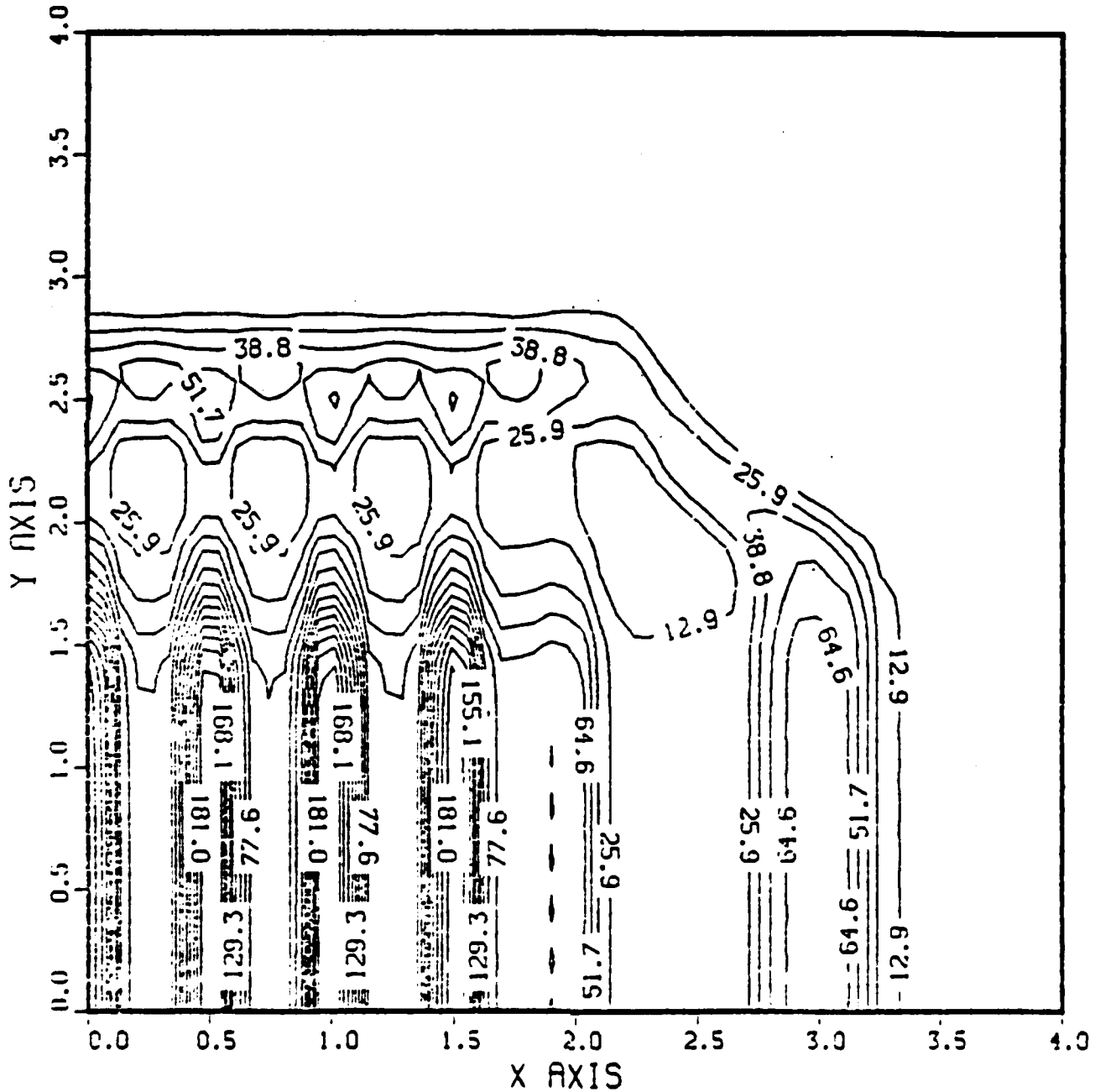


Figure 1.3c Pressure contours for a 9 lines 4m x 4m DECA array test. X-Y cross section.

DECA BLAST ON THE GROUND WEIGHT=100LB
X-Y PLANE, Z=0 CM, TIME = 0.72MSEC
PRESSURE

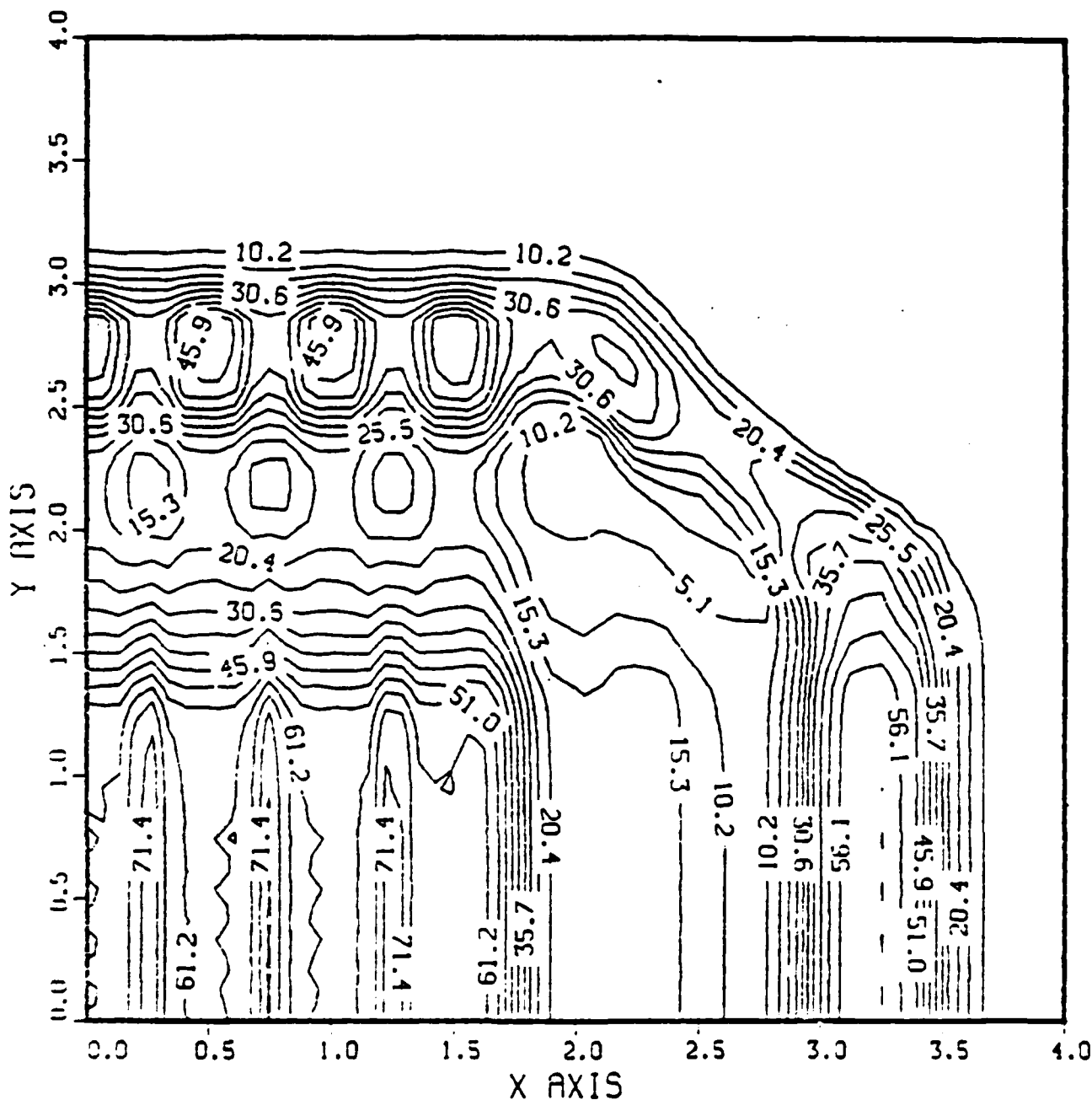


Figure 1.3d Pressure contours for a 9 lines 4m x 4m DECA array test. X-Y cross section.

100 LB. BLAST 9 LINES OF EXPLOSIVE
ISO-IMPULSE CONTOUR PLOTS
LOADS ON THE GROUND

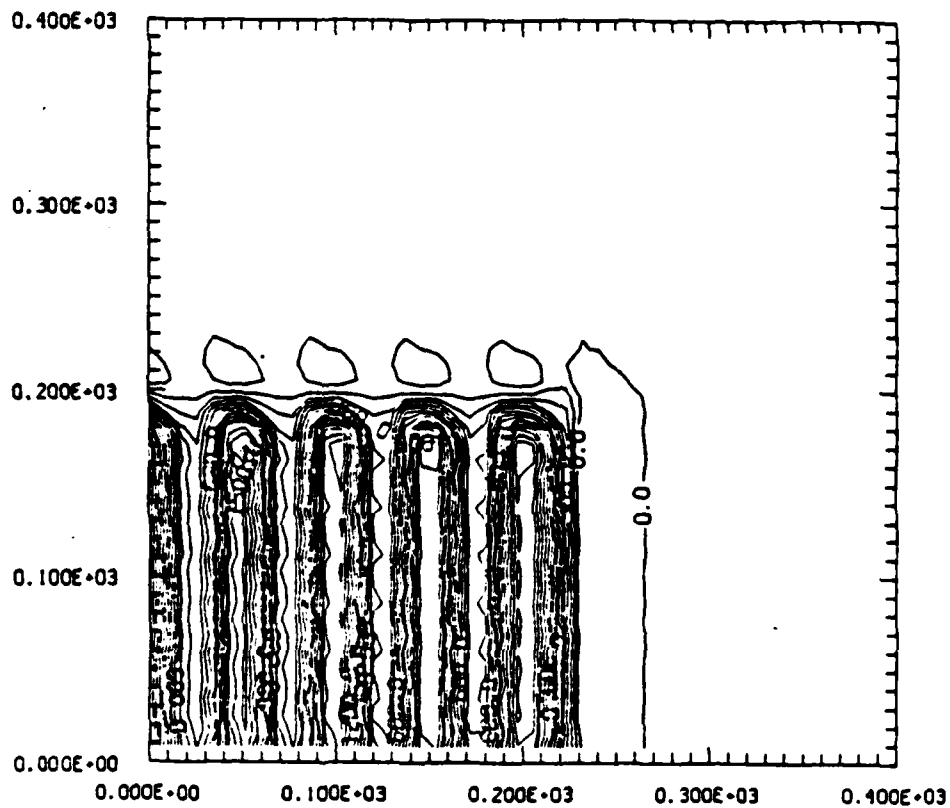


Figure 1.4a Impulse on the ground at $t = 0.25$ msec.

100 LB. BLAST 9 LINES OF EXPLOSIVE
ISO-IMPULSE CONTOUR PLOTS
LOADS ON THE GROUND

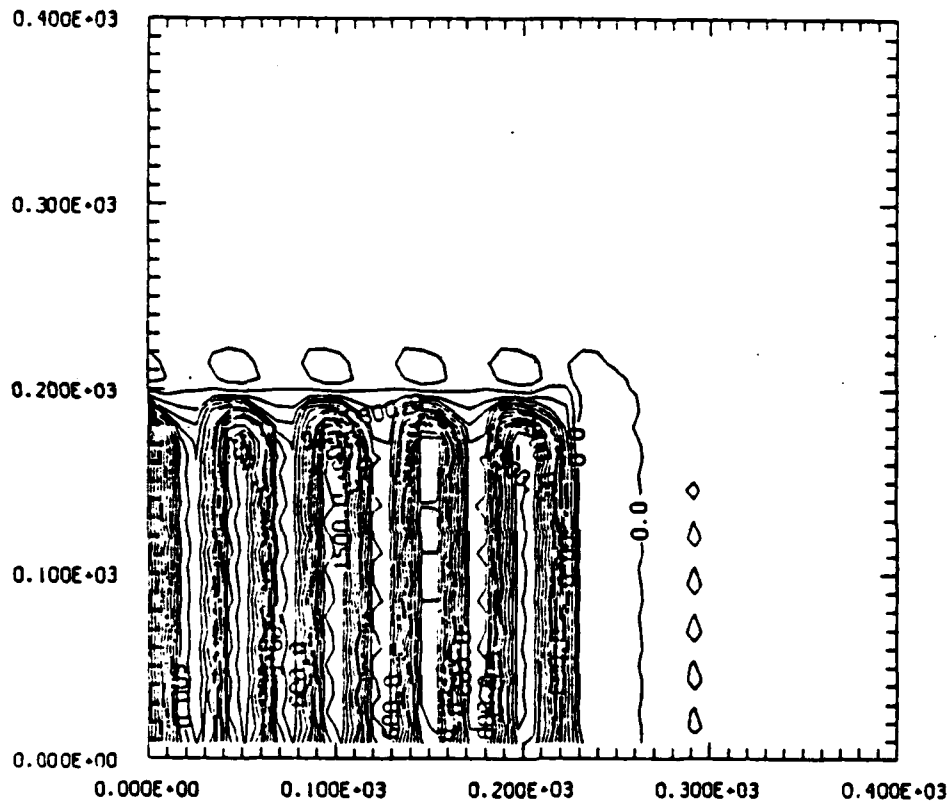


Figure 1.4b Impulse on the ground at $t = 0.50\text{msec}$.

100 LB. BLAST 9 LINES OF EXPLOSIVE
ISO-IMPULSE CONTOUR PLOTS
LOADS ON THE GROUND

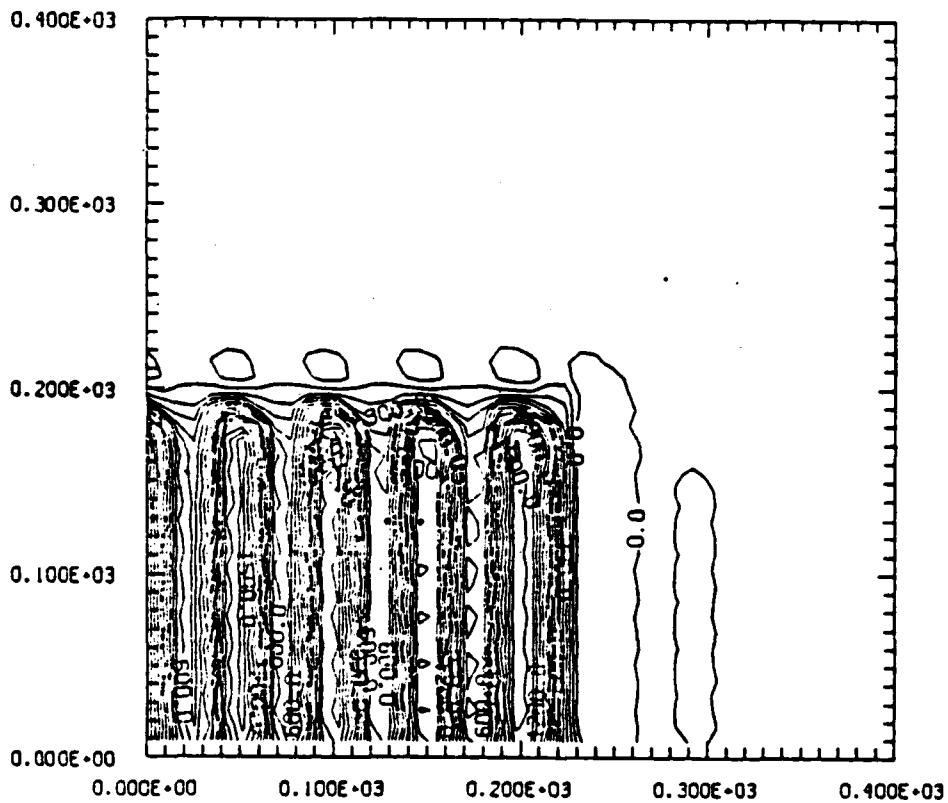


Figure 1.4c Impulse on the ground at $t = 1.00$ msec.

DECH 100 LB BLAST
TIME - HISTORY FOR STATION
X= 0 Y= 0 Z= 0(CM)

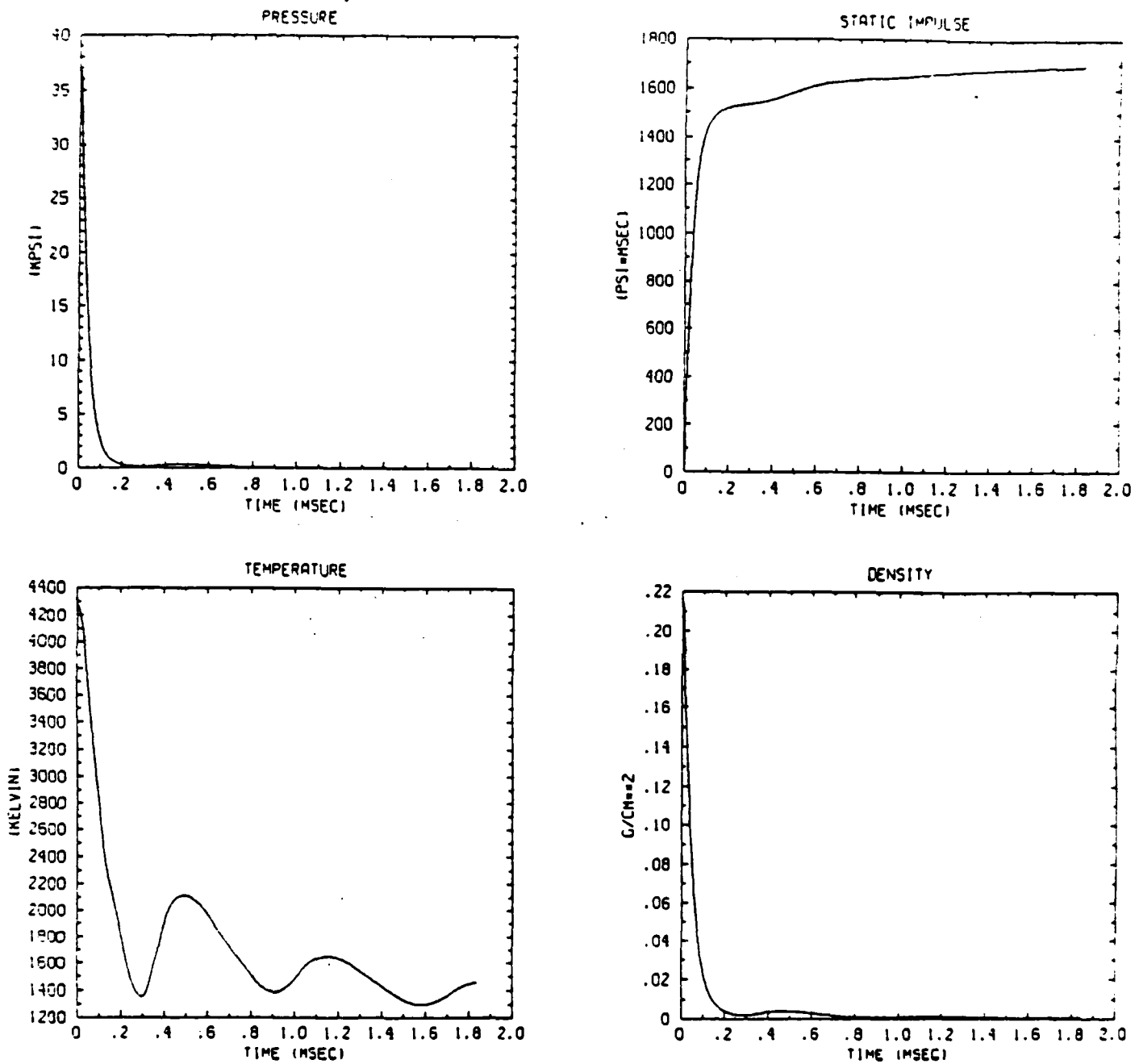


Figure 1.5a Time history of pressure, impulse, temperature and density on the ground under the explosive.

DECA 100 LB BLAST
 TIME - HISTORY FOR STATION
 X= 25 Y= 0 Z= 0(CM)

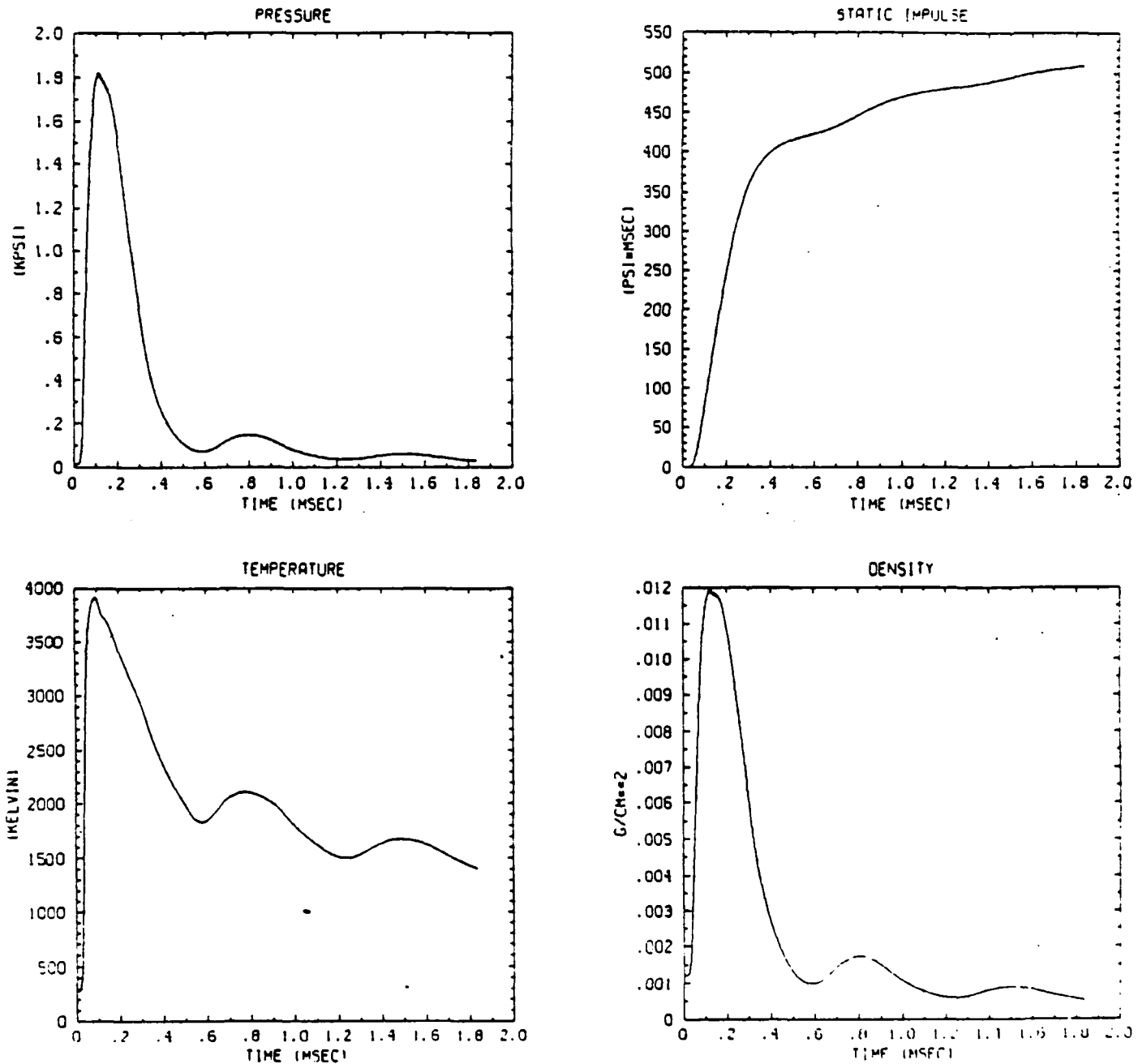


Figure 1.5b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

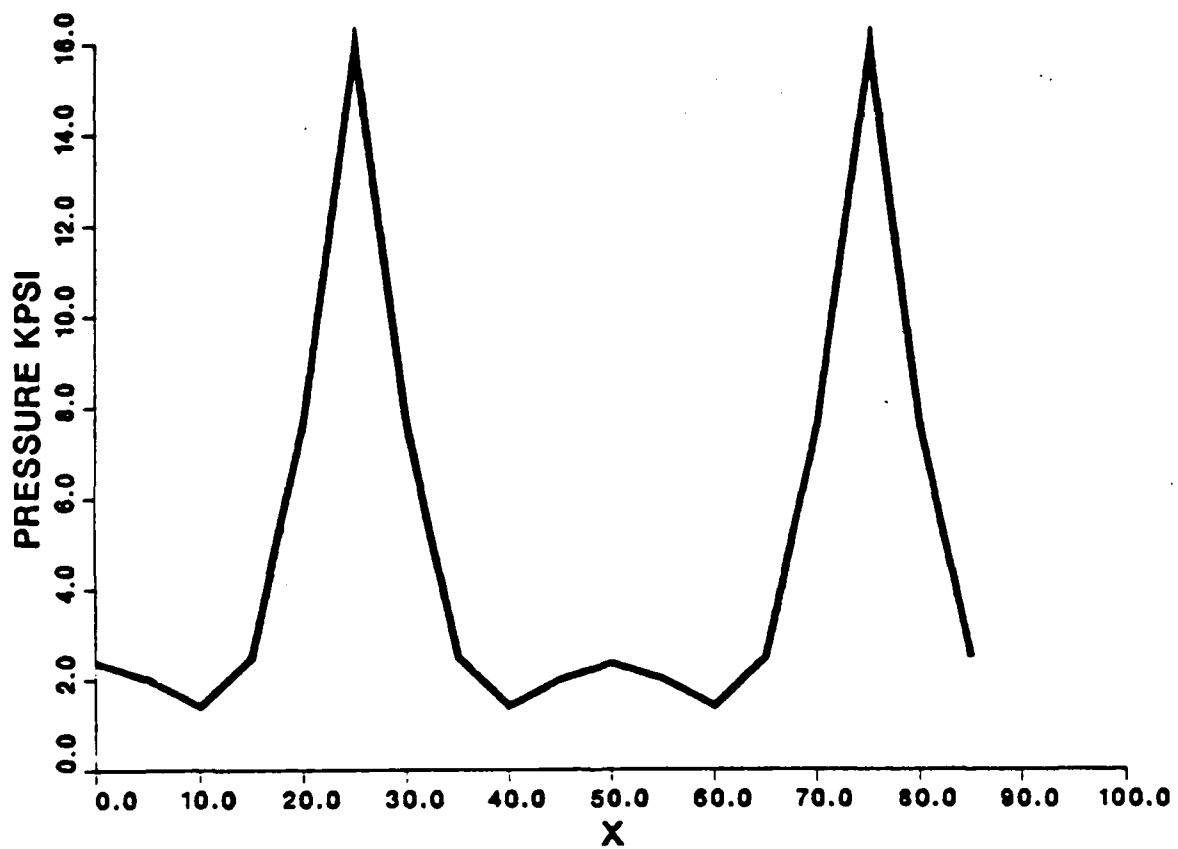
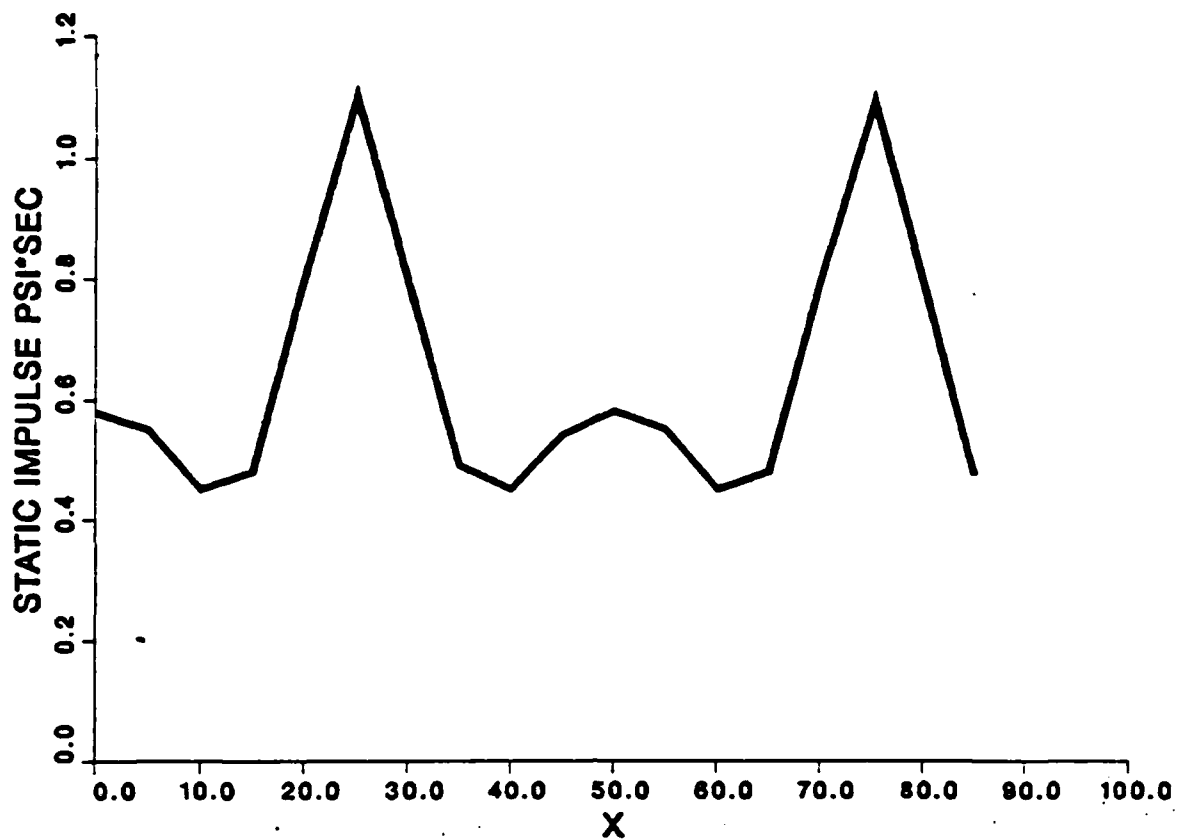


Figure 1.6 Maximum pressure and impulse distribution on the ground for 10 lines (110lb) of Iremite.

NRL
LCP

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Z PLANE, Y=0 CM, TIME = 0.15MSEC

PRESSURE

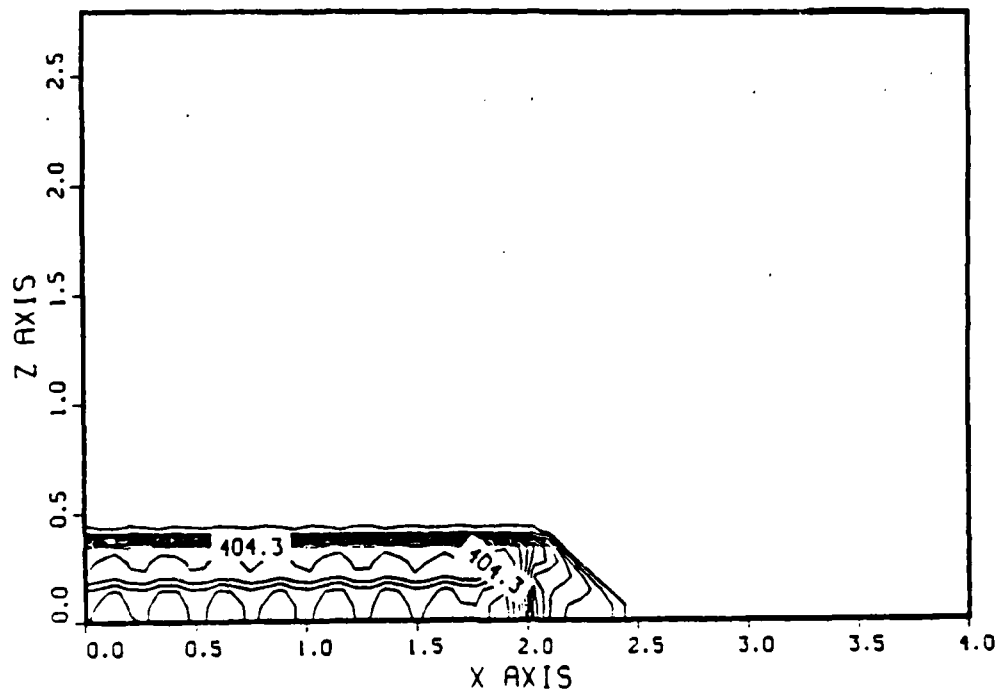


Figure 1.7a Pressure contours for 17 lines 4m x 4m DECA array. X-Z cross section.

NRL
LCP

DECA BLAST ON THE GROUND WEIGHT-100LB
X-Z PLANE, Y=0 CM, TIME = 0.35MSEC

PRESSURE

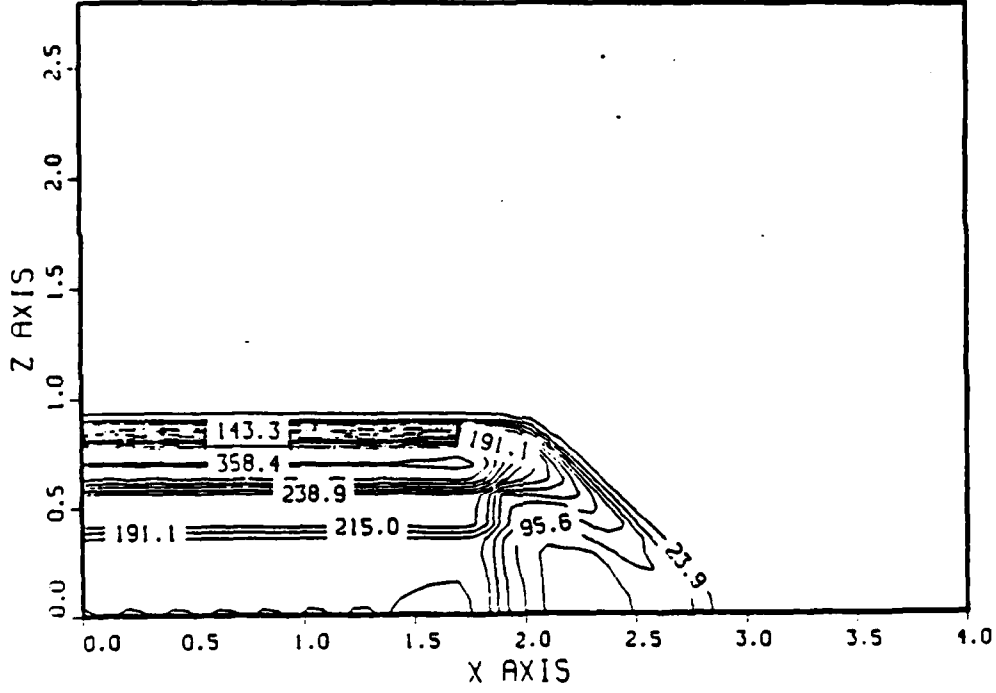


Figure 1.7b Pressure contours for 17 lines $4m \times 4m$ DECA array. X-Z cross section.

17 LINES OF EXPLOSIVE:
DECA BLAST ON THE GROUND WEIGHT-100LB
X-Y PLANE, Z=0 CM, TIME = 0.15MSEC
PRESSURE

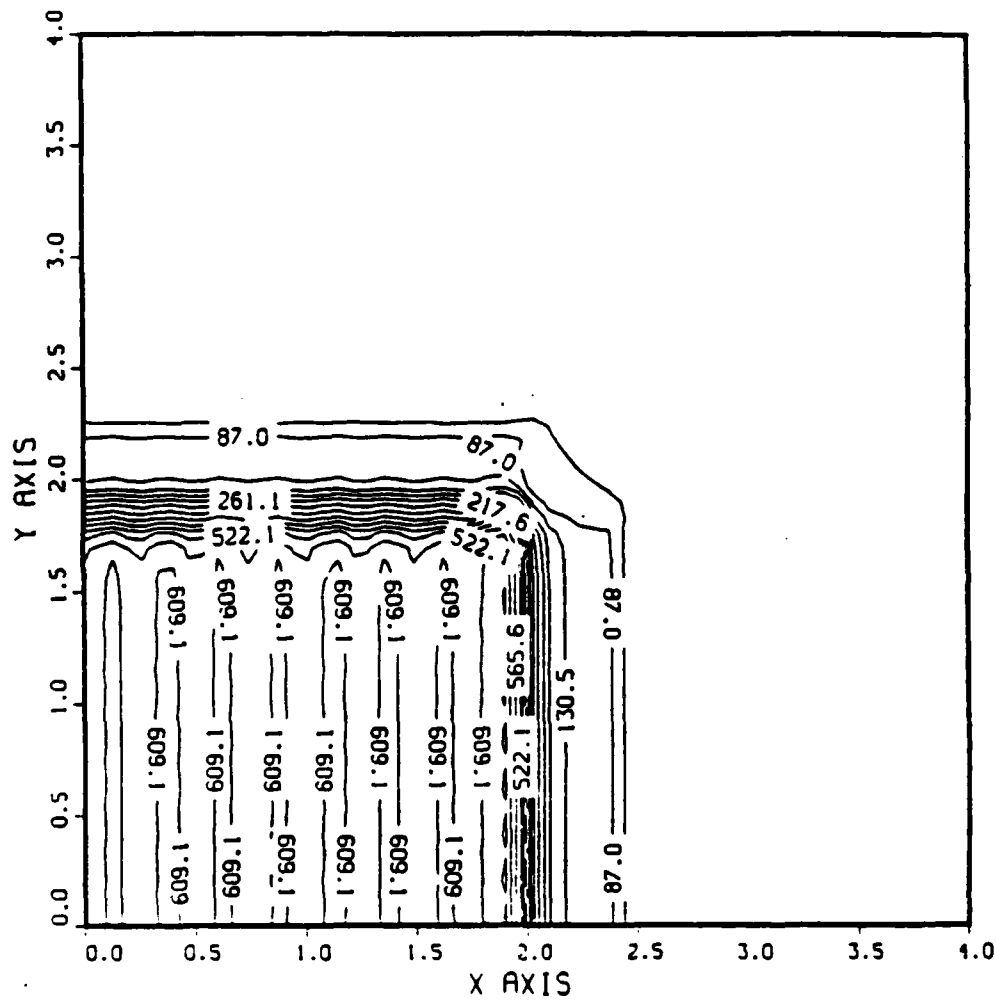


Figure 1.8a Pressure contours for 17 lines 4m x 4m DECA array. X-Y cross section.

NRL
LCP

17 LINES OF EXPLOSIVE.
DECA BLAST ON THE GROUND WEIGHT-100LB
X-Y PLANE, Z=0 CM, TIME = 0.35MSEC
PRESSURE

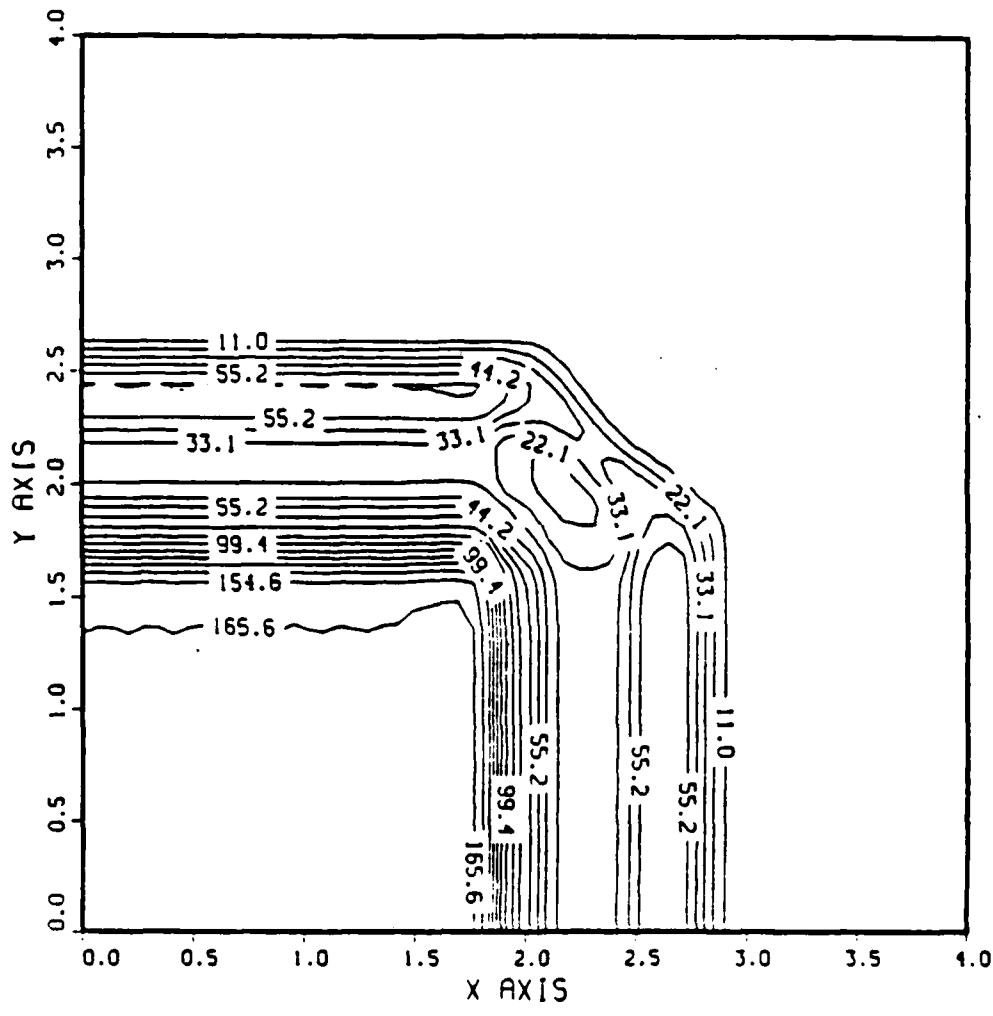


Figure 1.8b Pressure contours for 17 lines 4m x 4m DECA array. X-Y cross section.

ALONG 100 LB BLAST 17 LINES
 TIME - HISTORY FOR STATION
 X = 0 Y = 0 Z = 0 CM

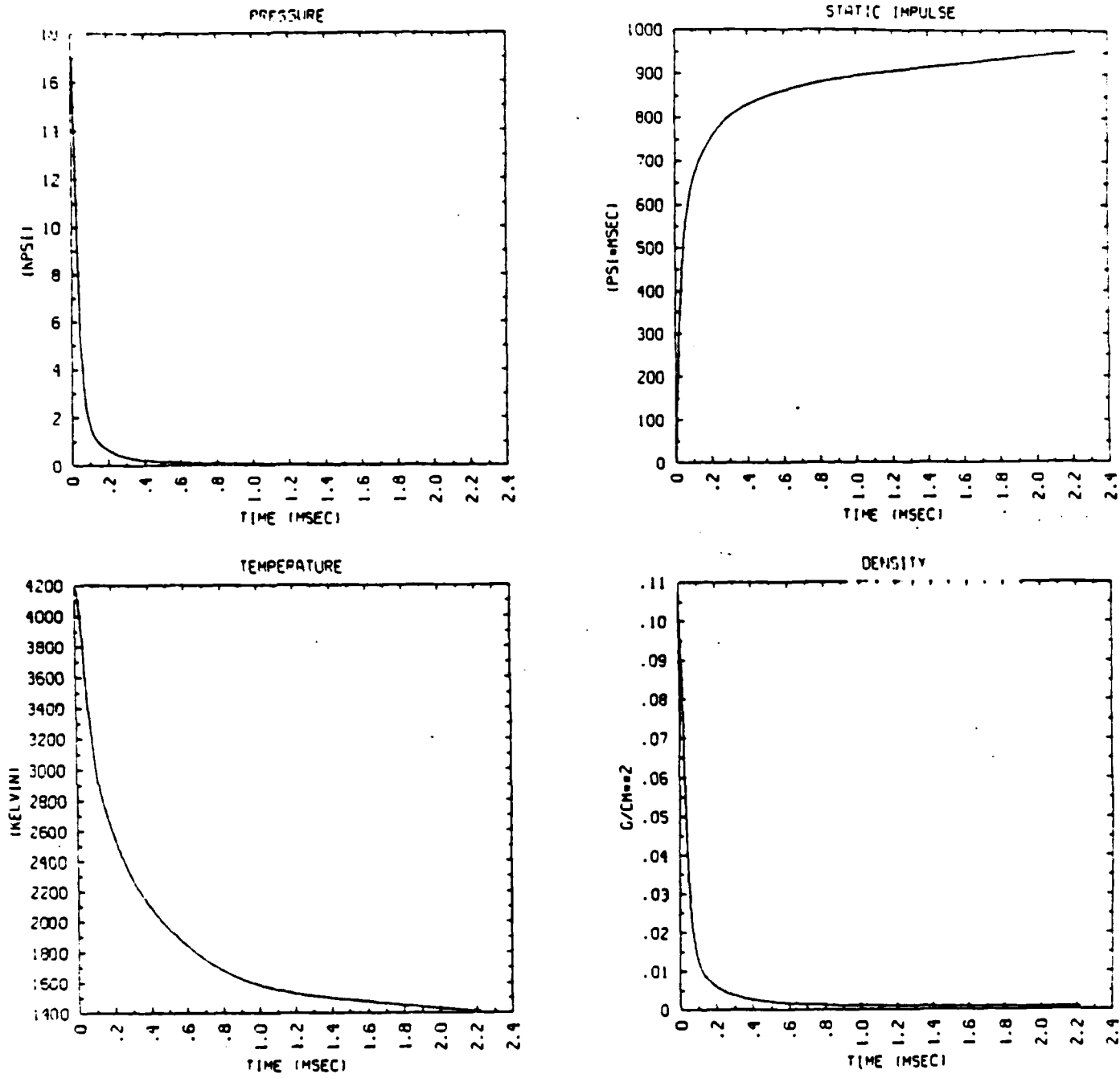


Figure 1.0a Time history of pressure, impulse, temperature and density on the ground under a line of explosive.

DECH 100 LB BLAST 17 LINES
 TIME - HISTORY FOR STATION
 X= 10 Y= 0 Z= 0 CM

17
 LC

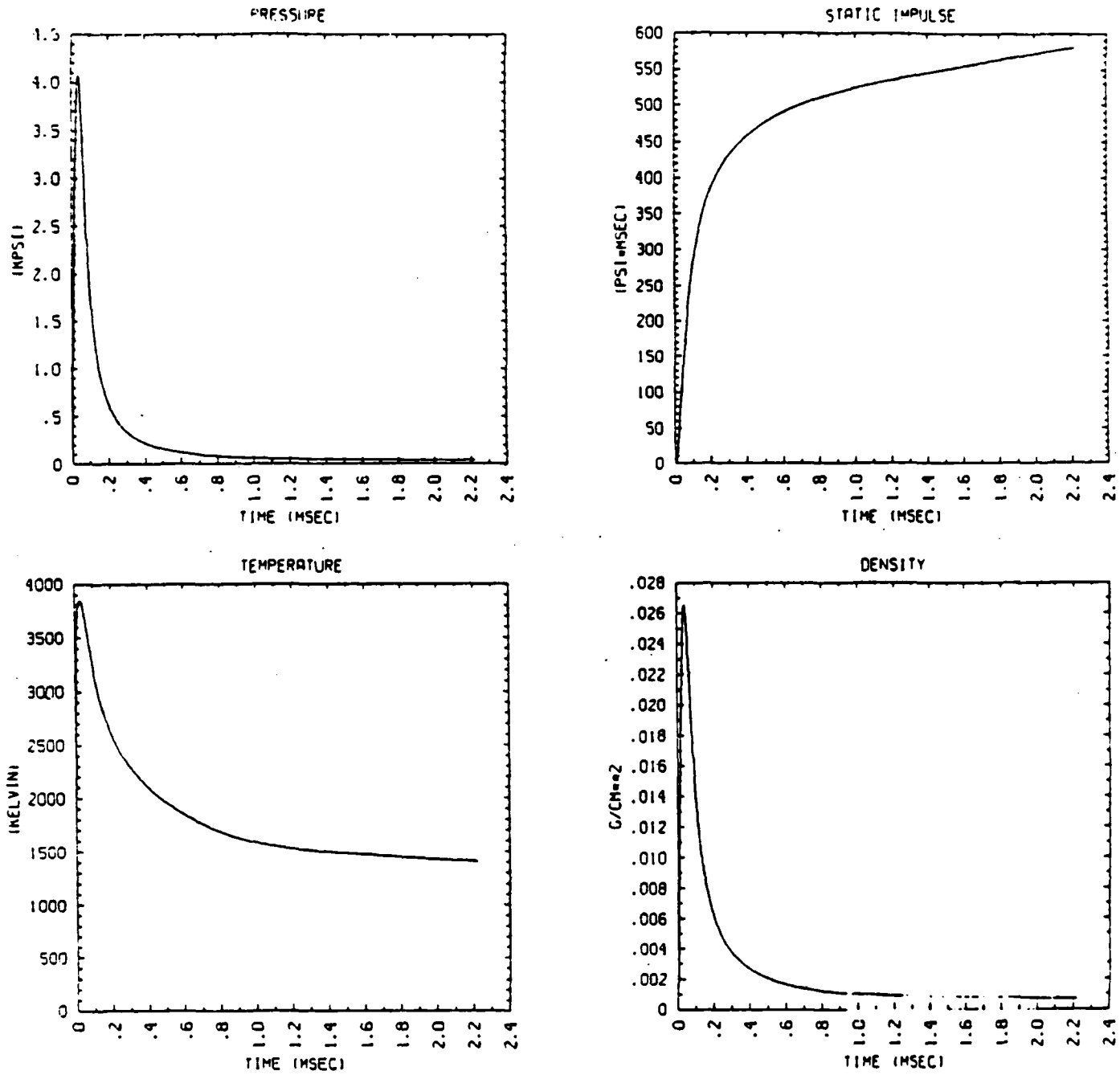


Figure 1.9b Time history of pressure, impulse, temperature and density on the ground between lines of explosive.

DECA 100 LB BLAST 17 LINES UP 20 CM
TIME - HISTORY FOR STATION
X= 0 Y= 0 Z= 0 CM

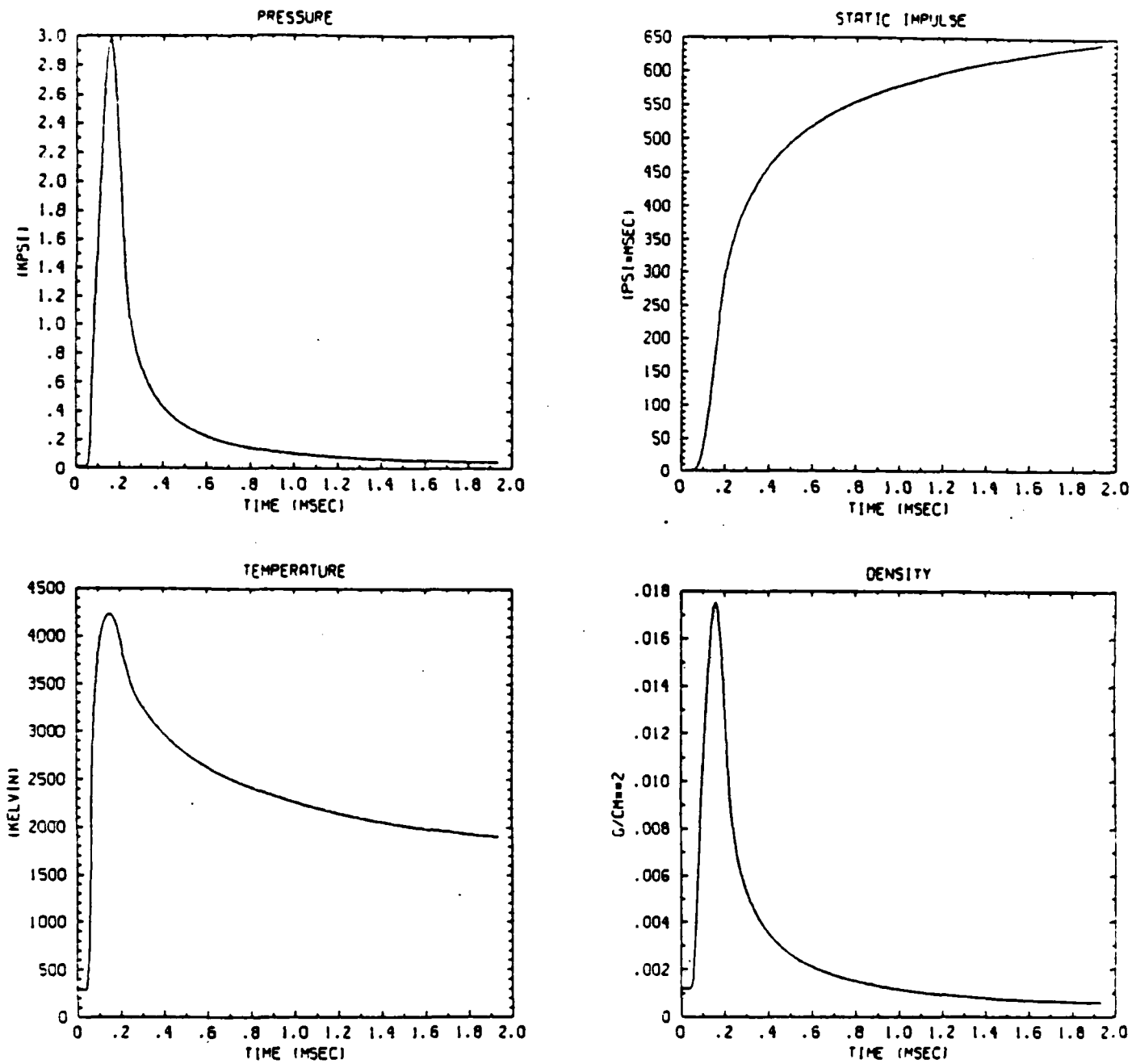


Figure 1.10a Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

100 LB BLAST 17 LINES UP 20 CM
 TIME - HISTORY FOR STATION
 X = 10 Y = 0 Z = 0 CM

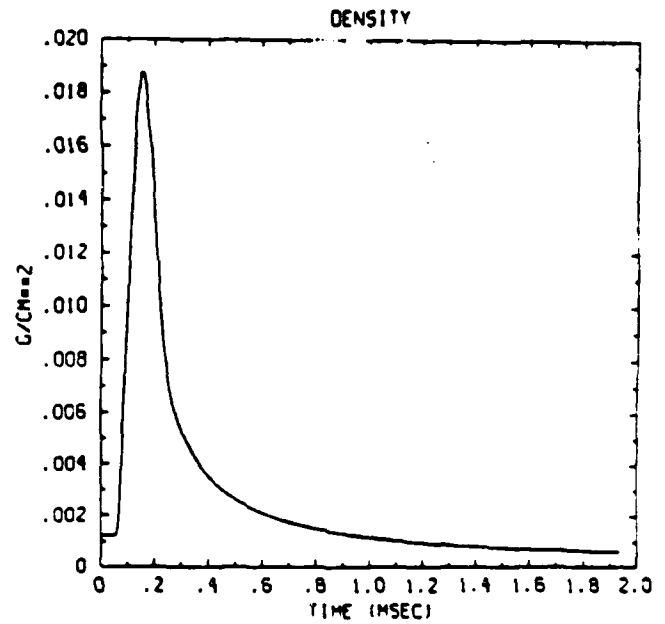
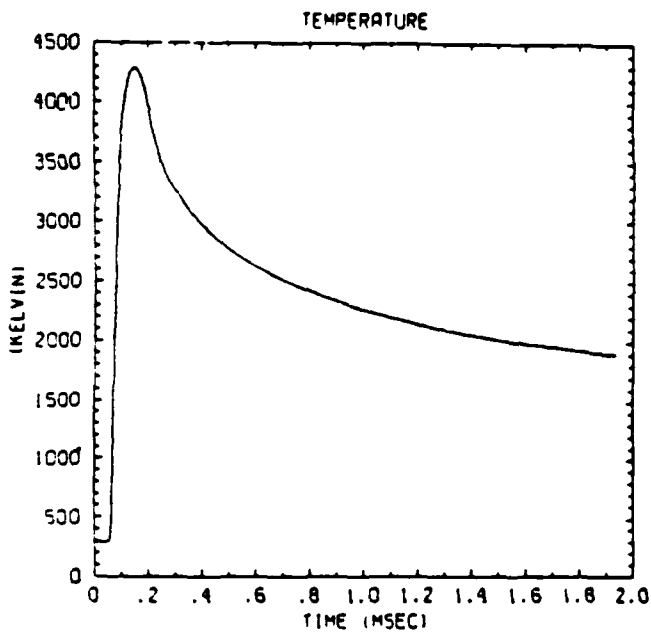
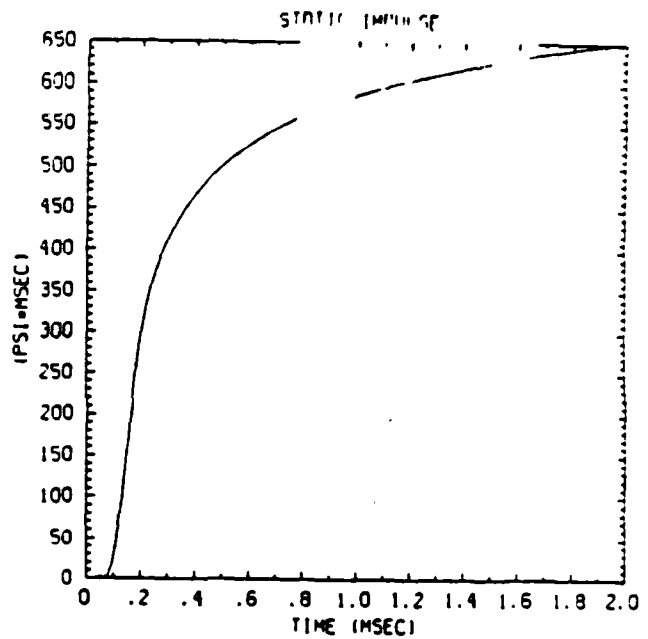
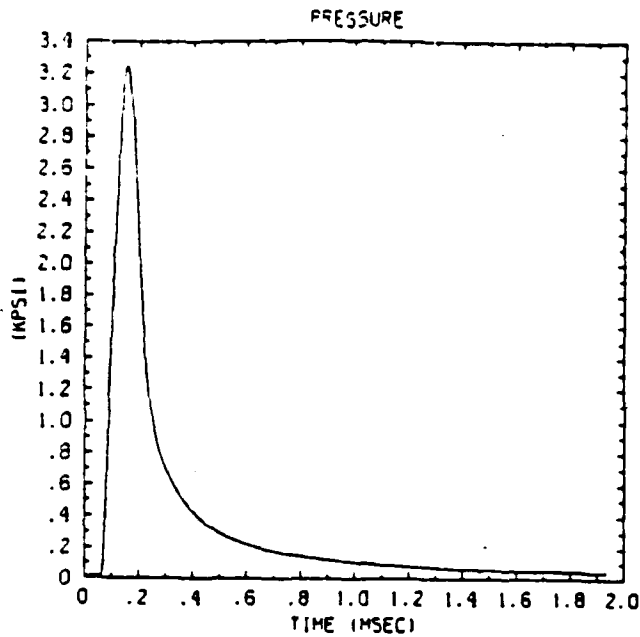


Figure 1.10b Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

DECA 100 LB BLAST 17 LINES UP 20 CM
 TIME - HISTORY FOR STATION
 X= 25 Y= 0 Z= 0 CM

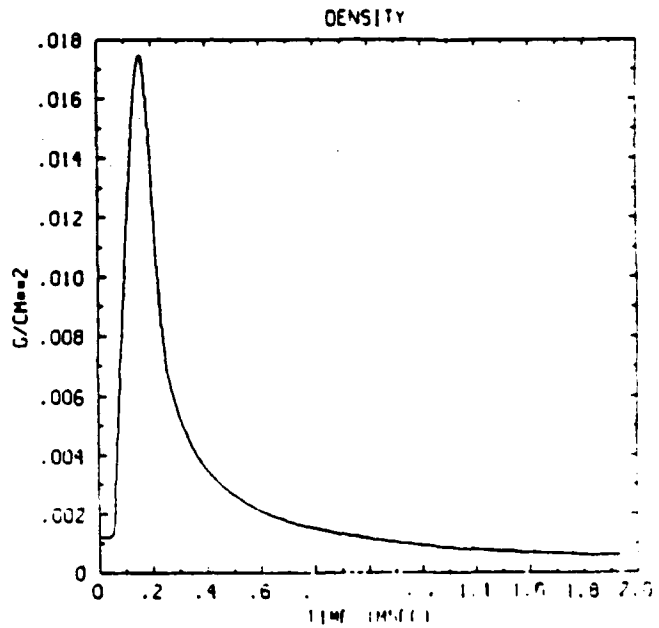
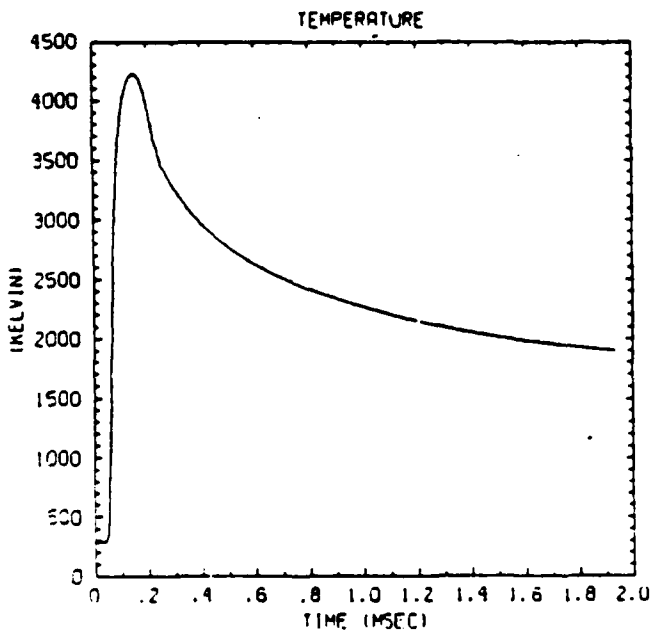
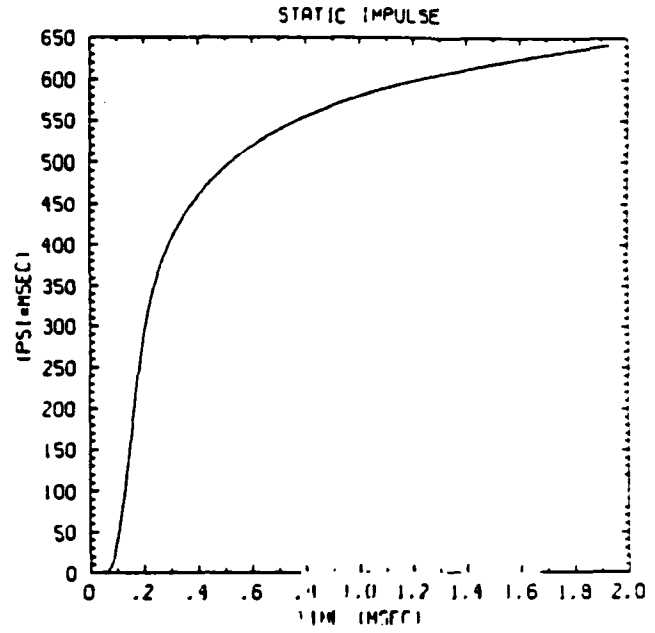
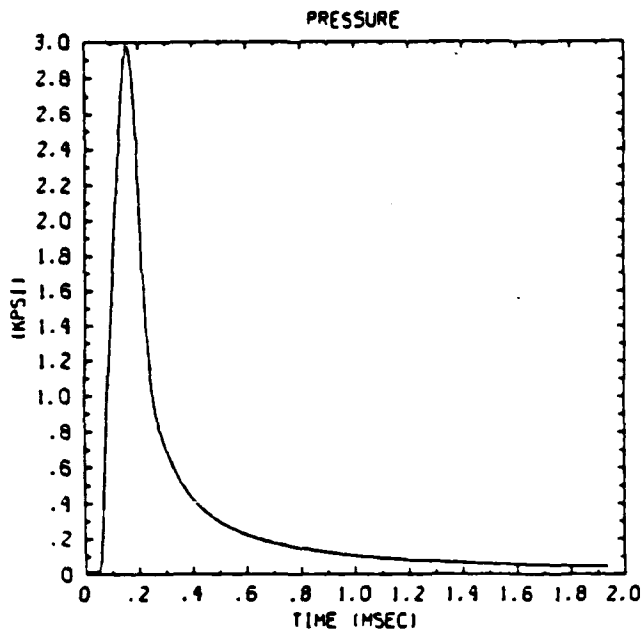


Figure 1.10c Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

DECA 100 LB BLAST 17 LINES UP 20 CM
 TIME - HISTORY FOR STATION
 X= 40 Y= 0 Z= 0 CM

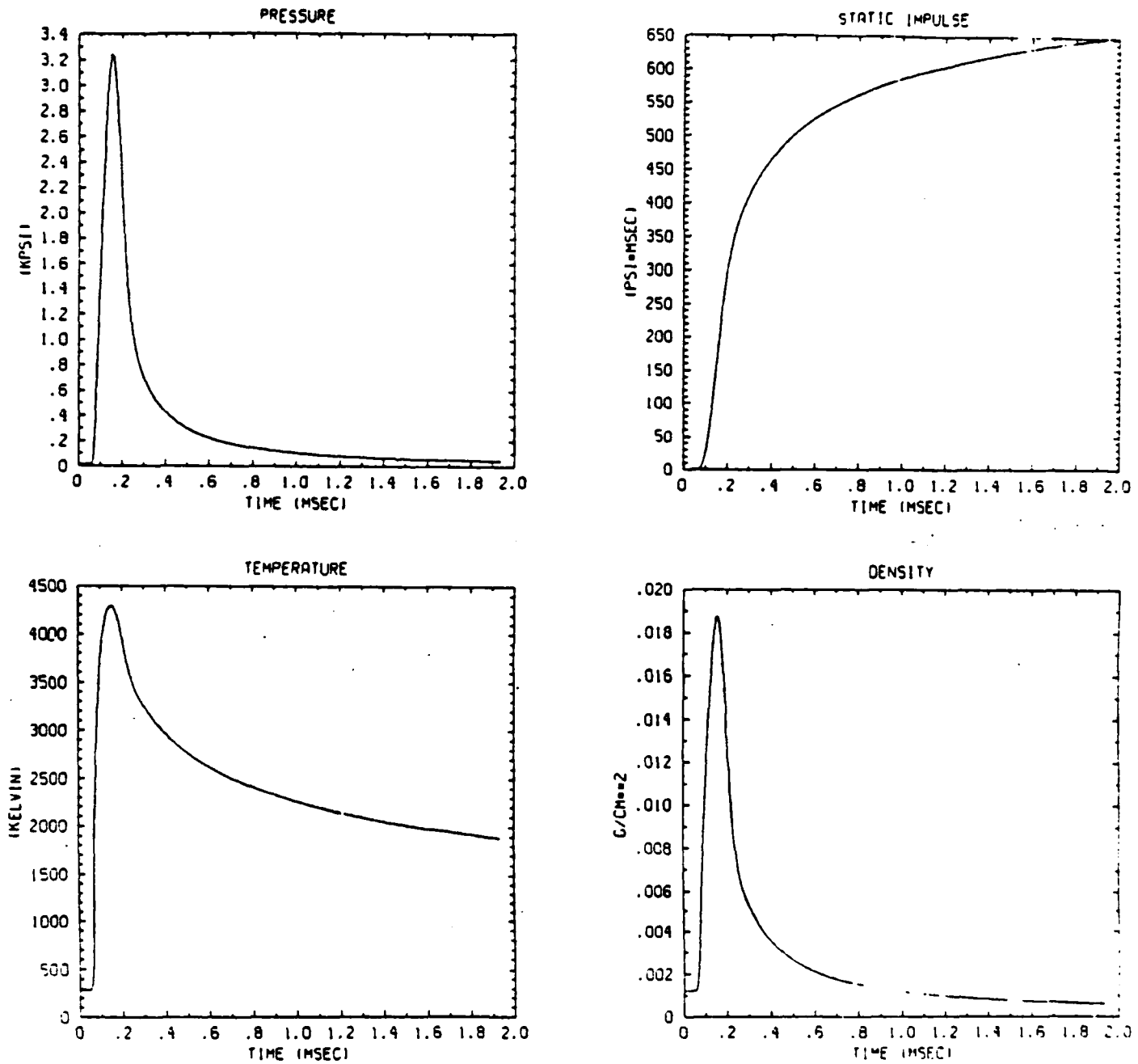


Figure 1.10d Time history of pressure, impulse, temperature and density on the ground for array detonation 20cm above the ground.

**30 LB DECA BLAST 25 CM ABOVE THE GROUND
2M X 2M ARRAY, X-Z PLANE, Y=0 ft**

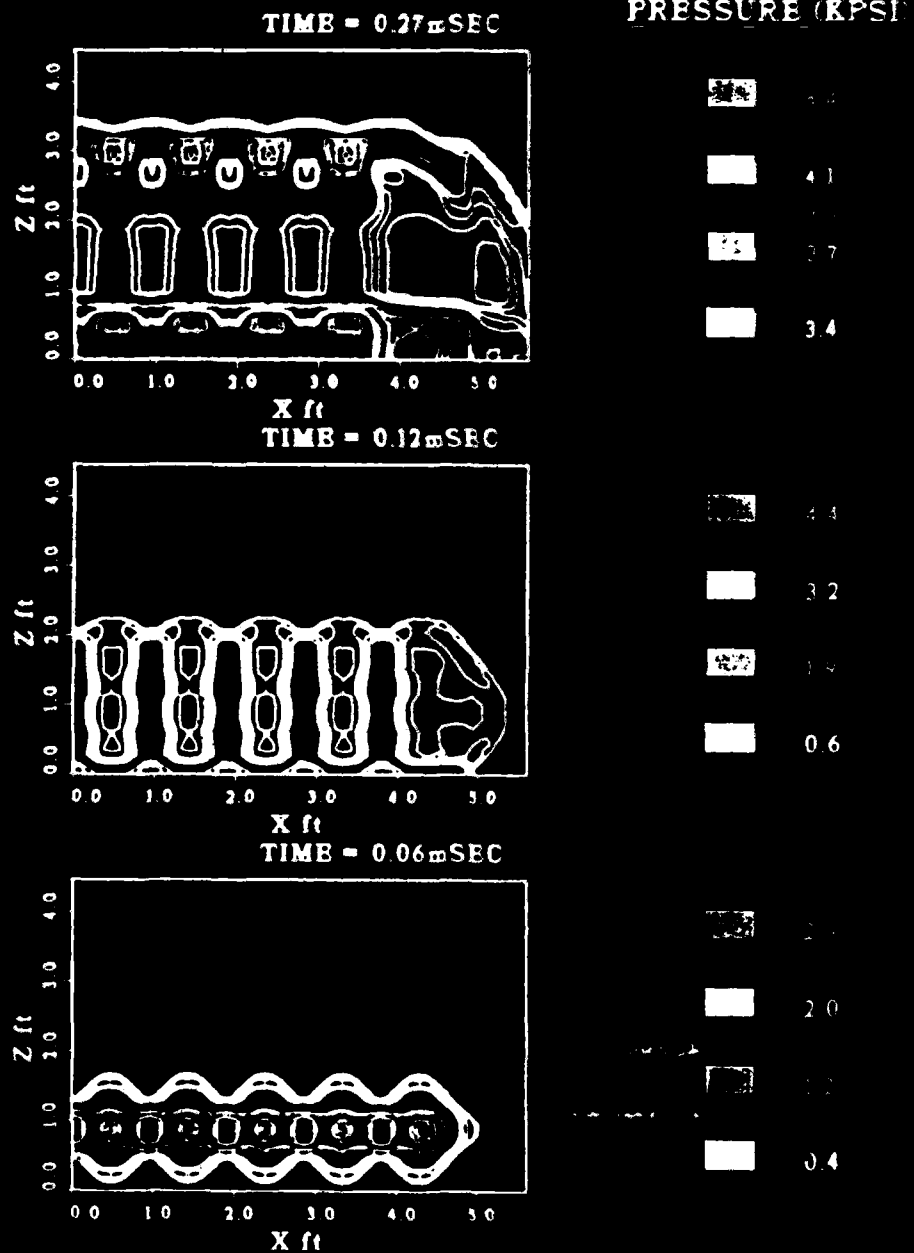


Figure 1.11 Pressure contours for 10 lines (2.25m x 2.0m) Primacord detonation.

**30 LB DECA BLAST 25 CM ABOVE THE GROUND
2M X 2M ARRAY, X-Y PLANE, Z=0 ft**

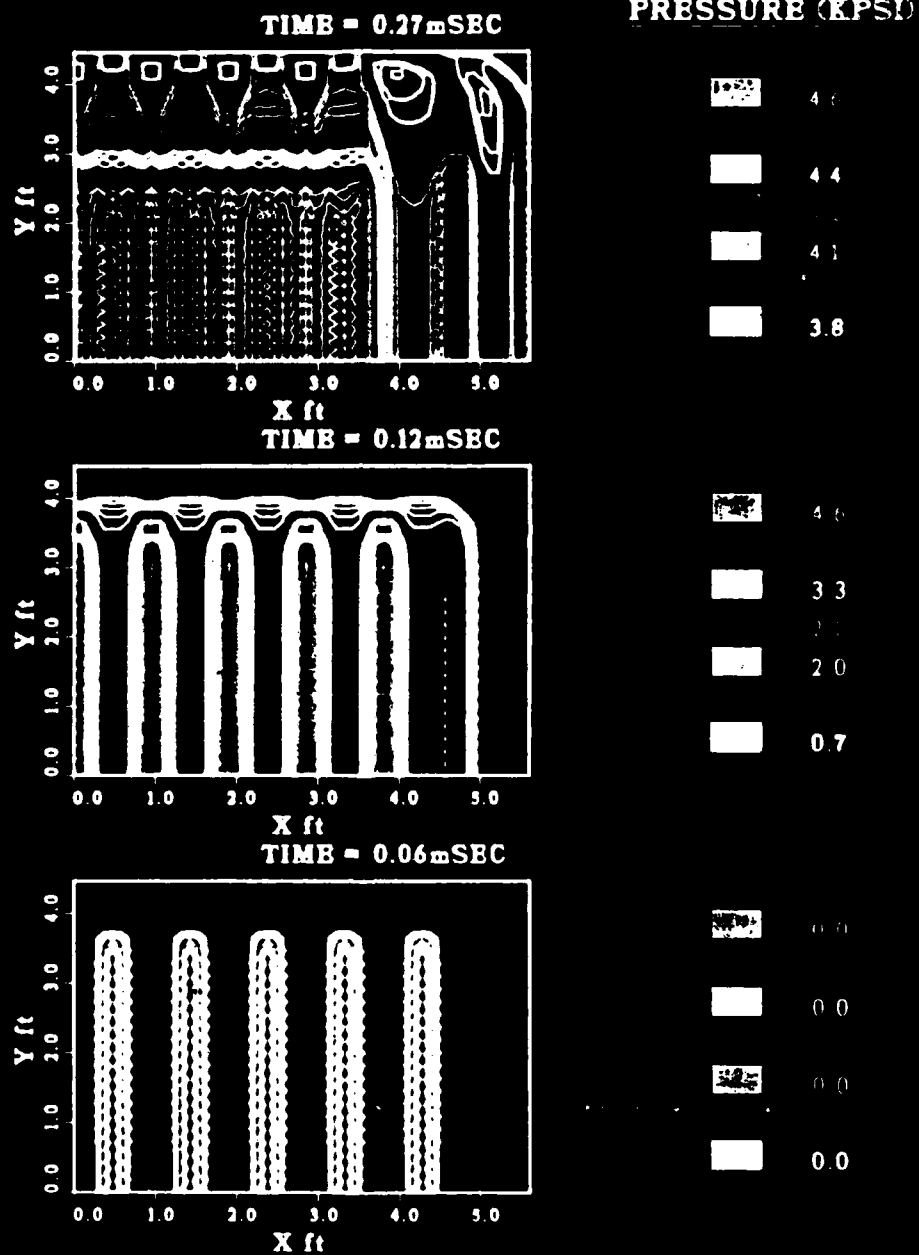


Figure 1.12 Pressure contours for 10 lines (2.25m x 2.0m) Primacord detonation.

W LINES BULB UP ZSUN MINDS (II)
 TIME - HISTORY FOR STATION
 X= 0 Y= 0 Z= 0 CM

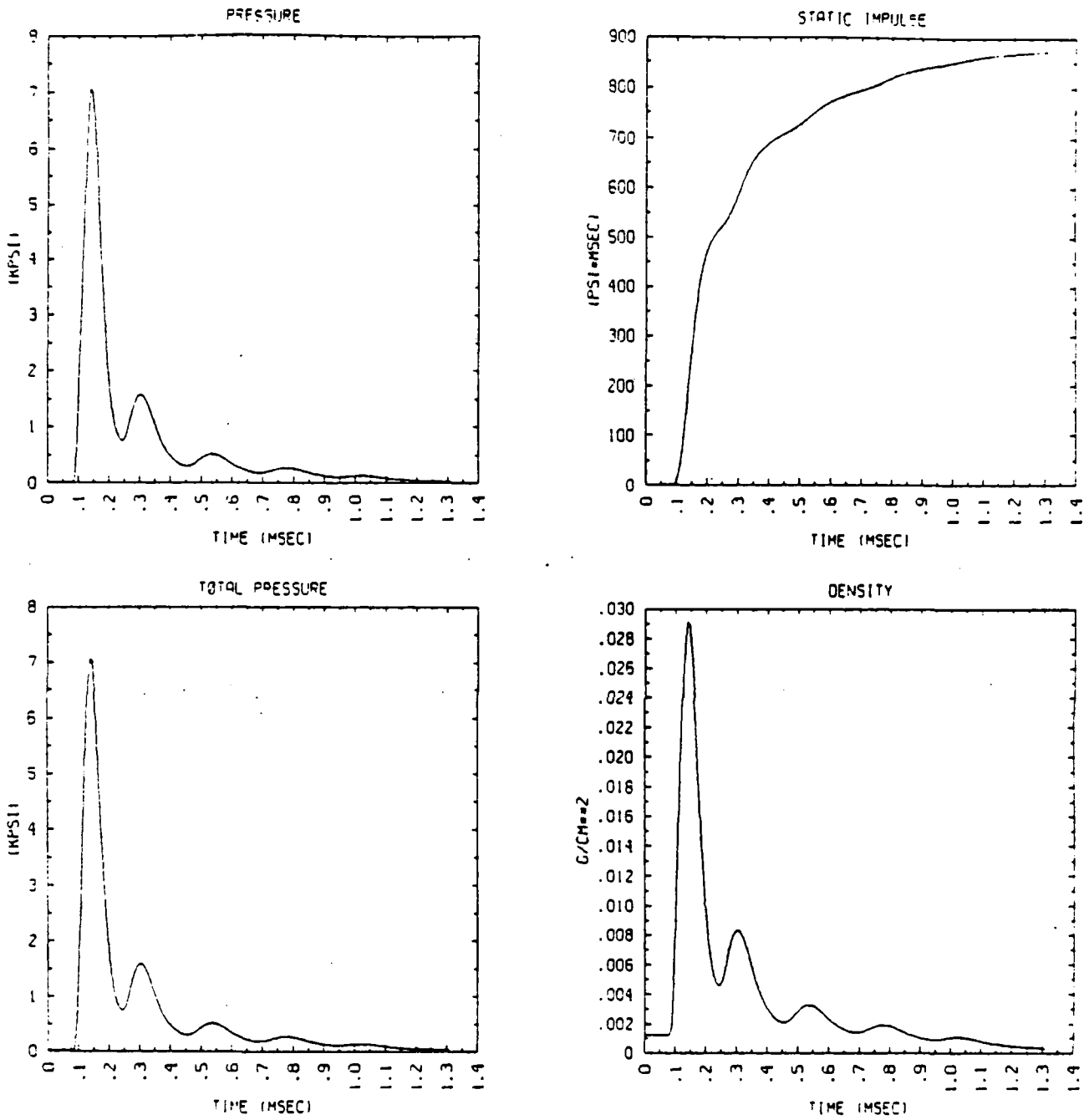


Figure 1.13a Time history for pressure, impulse, total pressure and density on the ground.
 X=0, Y=0, Z=0

TO LINE 5 COL 17 ZOOM PRINTED ON
 TIME - HISTORY FOR STATION
 X= 5 Y= 0 Z= 0 CM

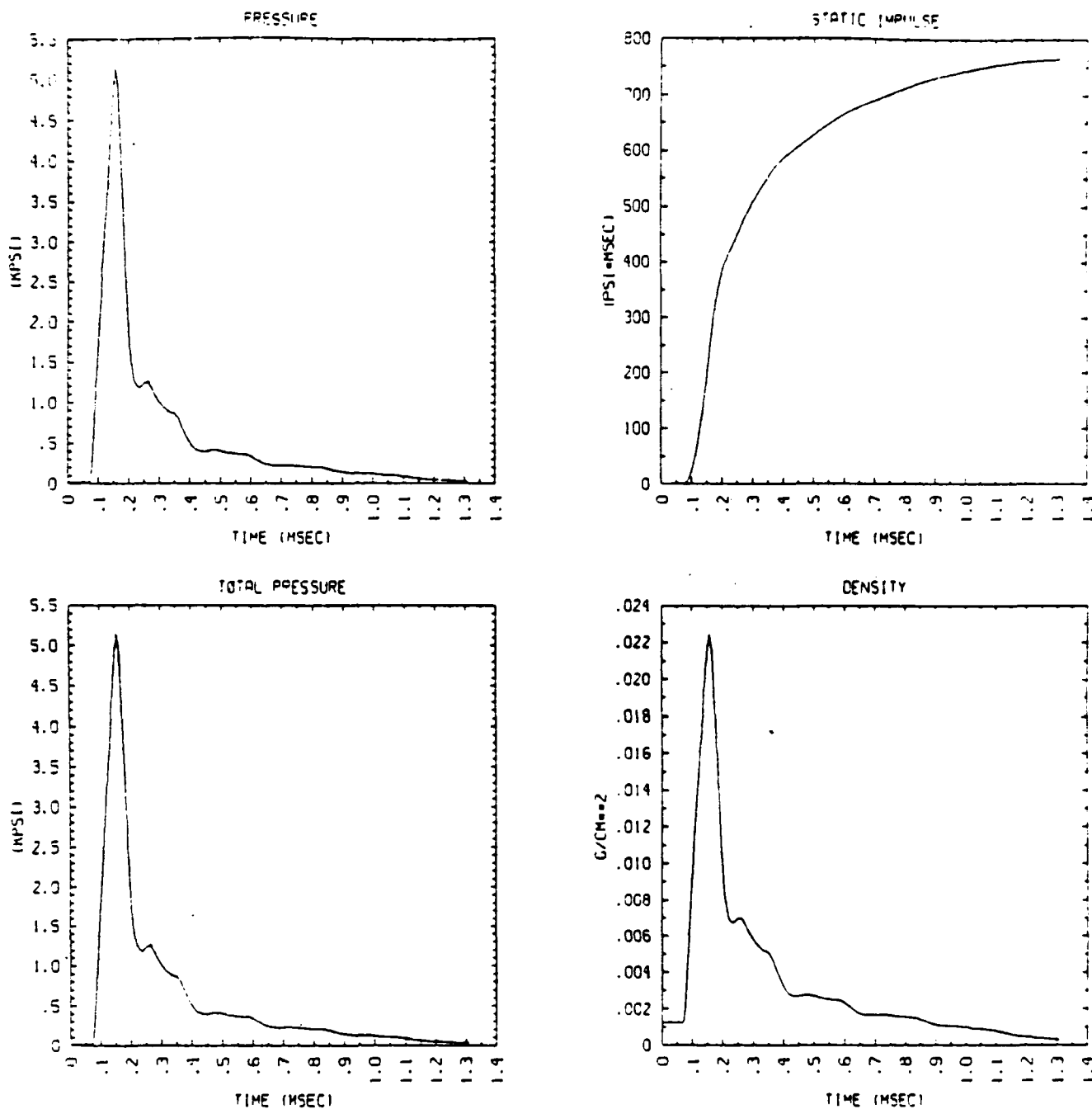


Figure 1.13b Time history for pressure, impulse, total pressure and density on the ground.
 X=5, Y=0, Z=0

10 LINES 3000 FT 2500 FTILES ON
 TIME - HISTORY FOR STATION
 X= 10 Y= 0 Z= 0 CM

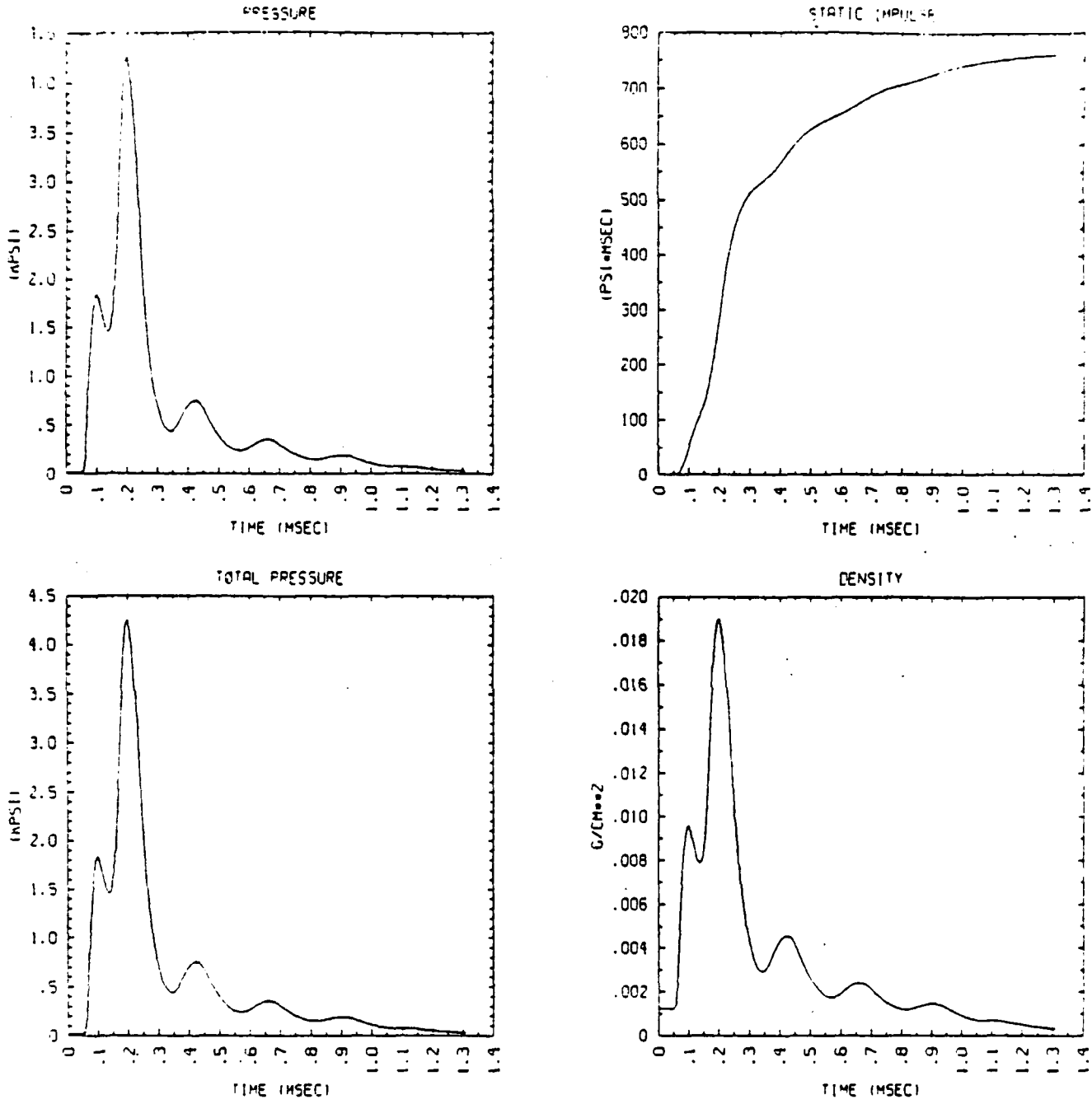


Figure 1.13c Time history for pressure, impulse, total pressure and density on the ground.
 X=10, Y=0, Z=0

TIME - HISTORY FOR STATION
 X= 20 Y= 0 Z= 0 CM

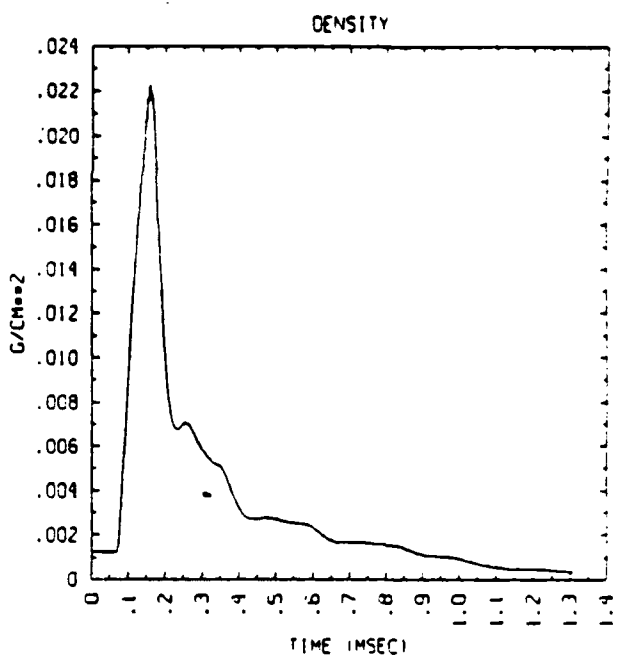
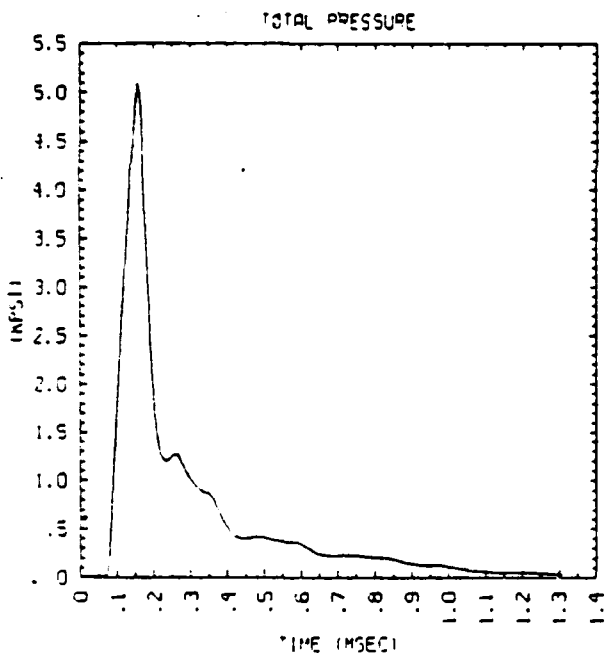
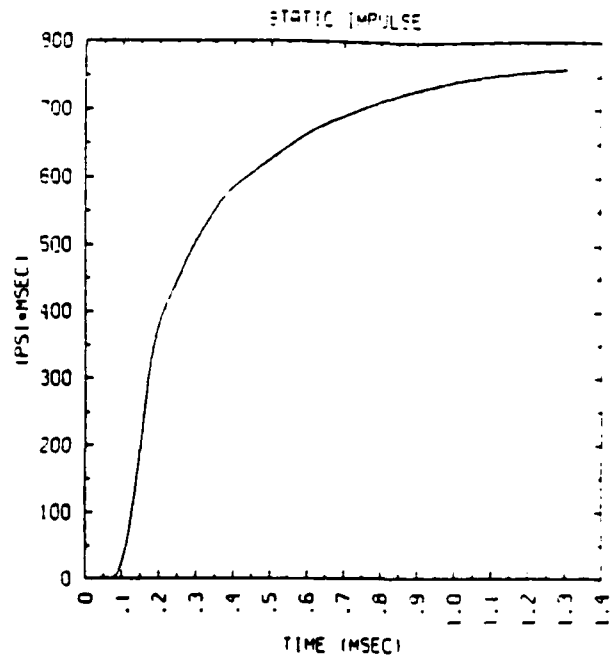
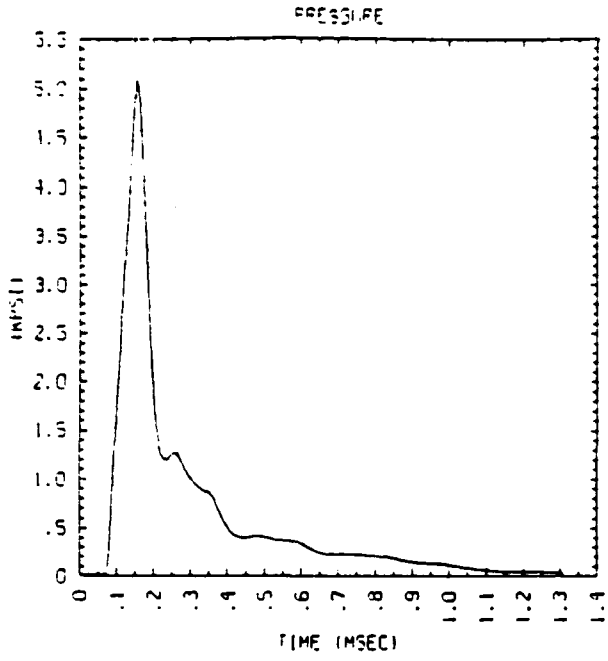


Figure 1.13d Time history for pressure, impulse, total pressure and density on the ground.
 X=20, Y=0, Z=0

COMPARISON OF M58, FAE AND DECA

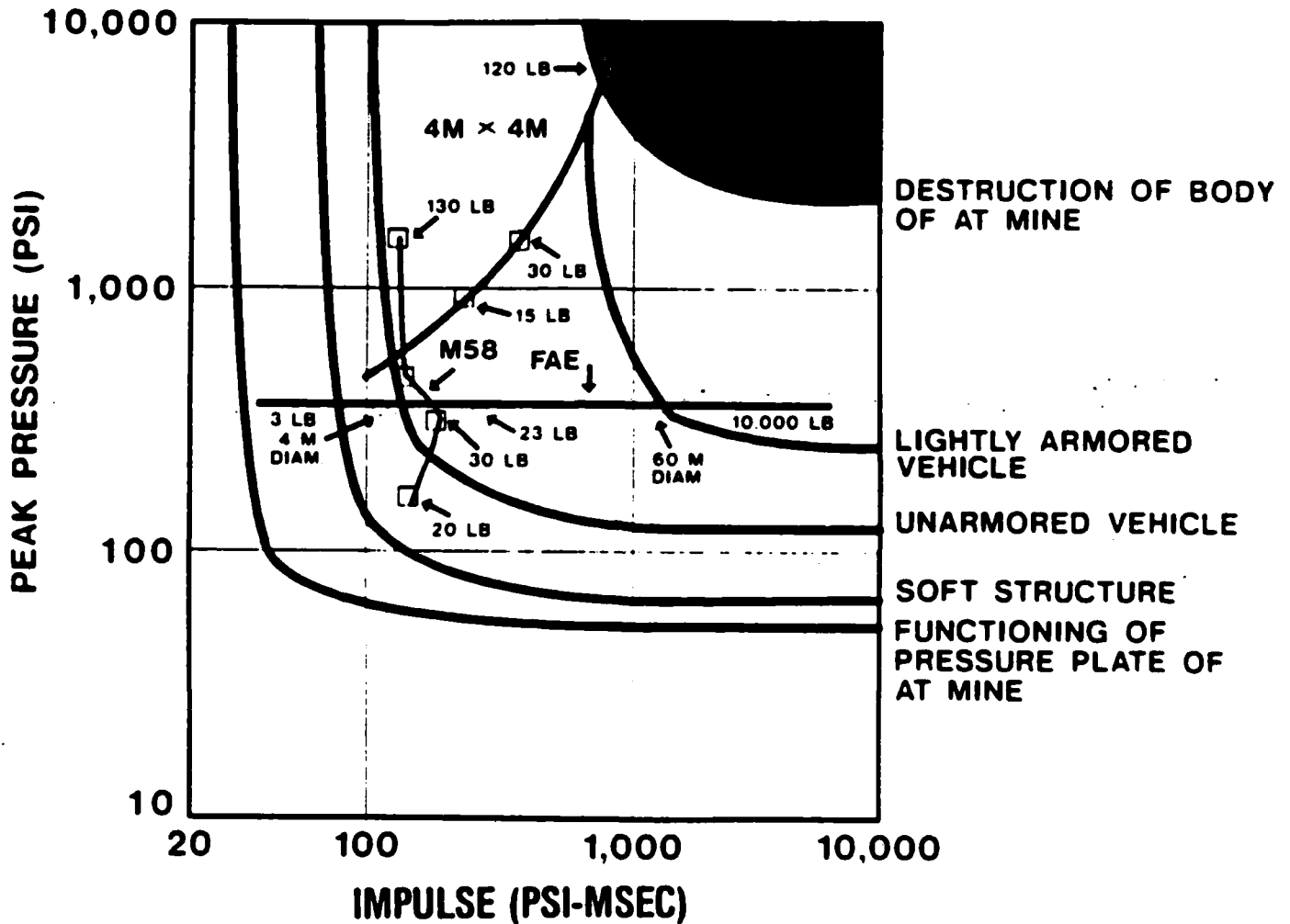


Figure 1.14 Maximum pressure and impulse as function of surface (volume for FAE) density of explosive.

Table 1. SUMMARY OF TEST RESULTS

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

^aNo useful data were obtained because of severe vibrations in instrumentation poles.

^bExplosive did not fully detonate (about 50% detonation).

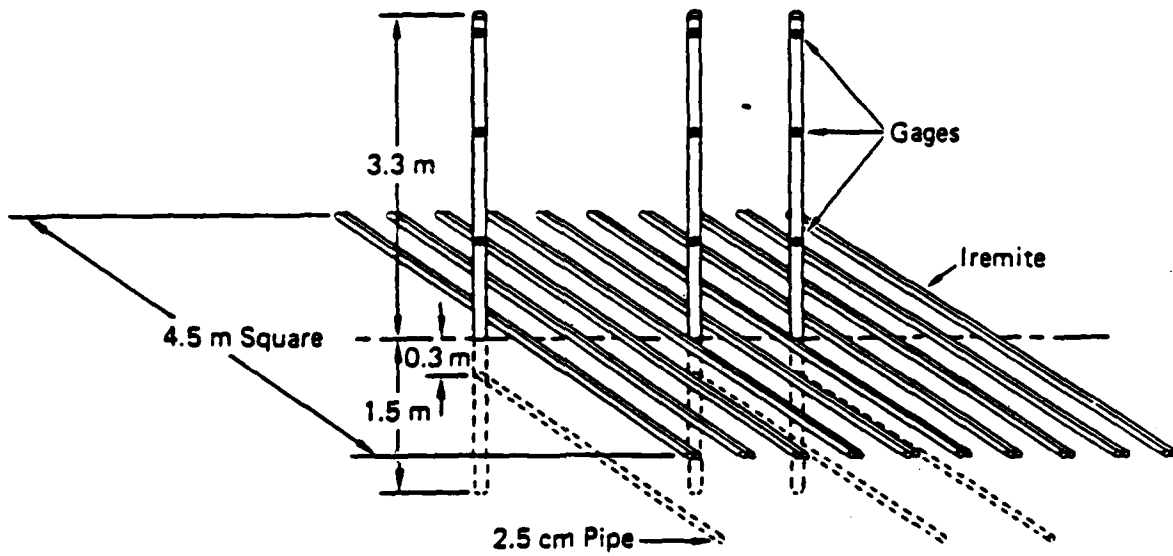


Figure 2.1 Perspective of experimental configuration.



Figure 2.2 Explosive array and instrumentation poles.

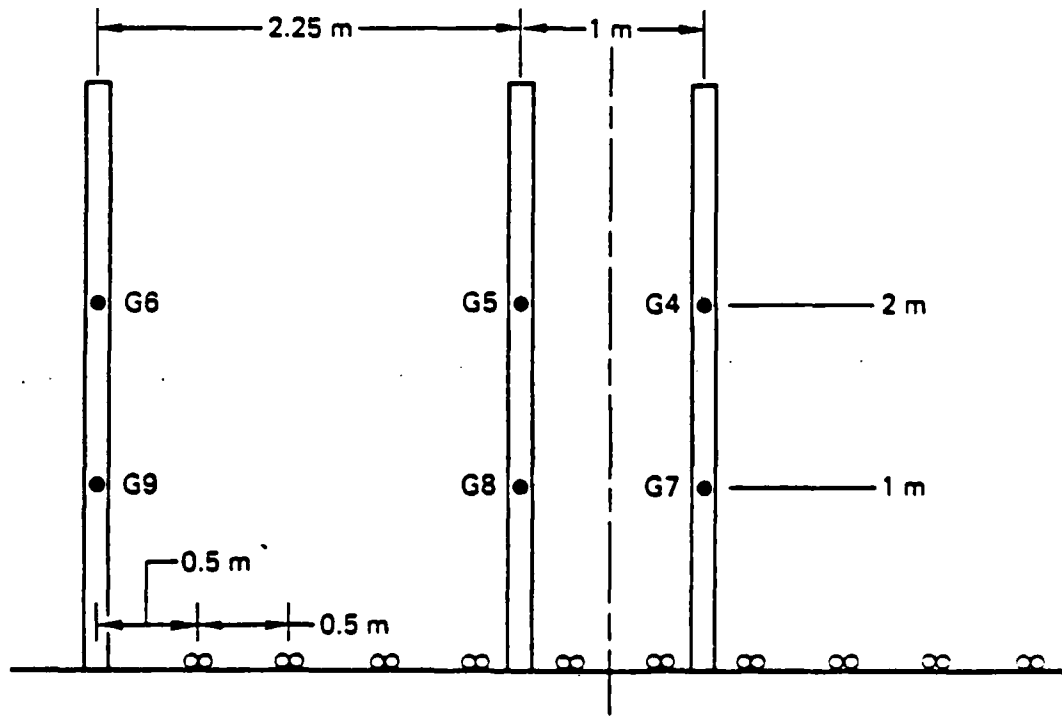
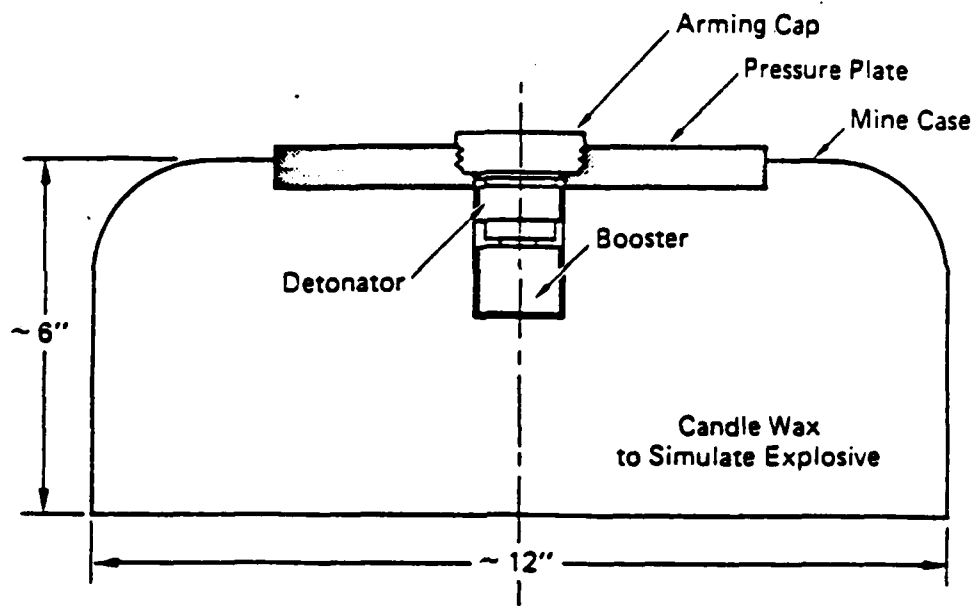
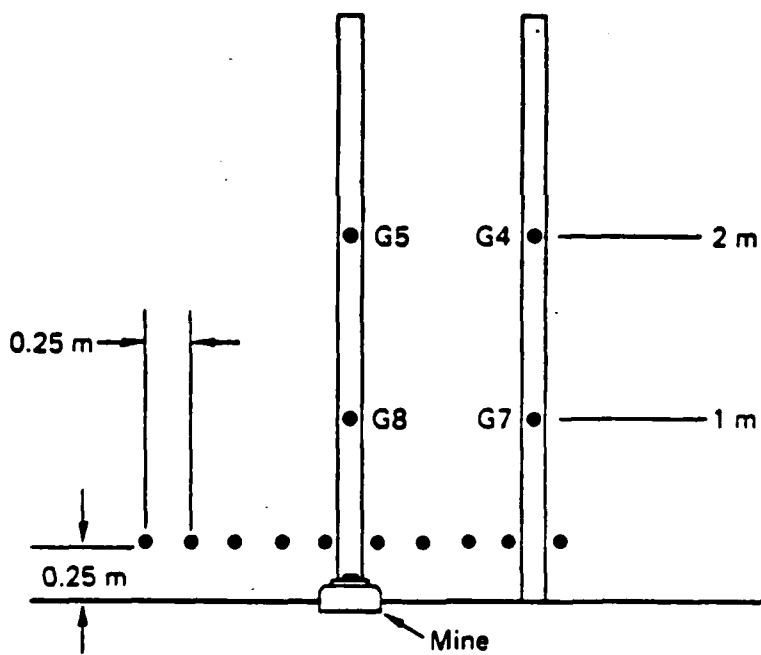


Figure 2.3 Experimental setup showing instrumentation poles.

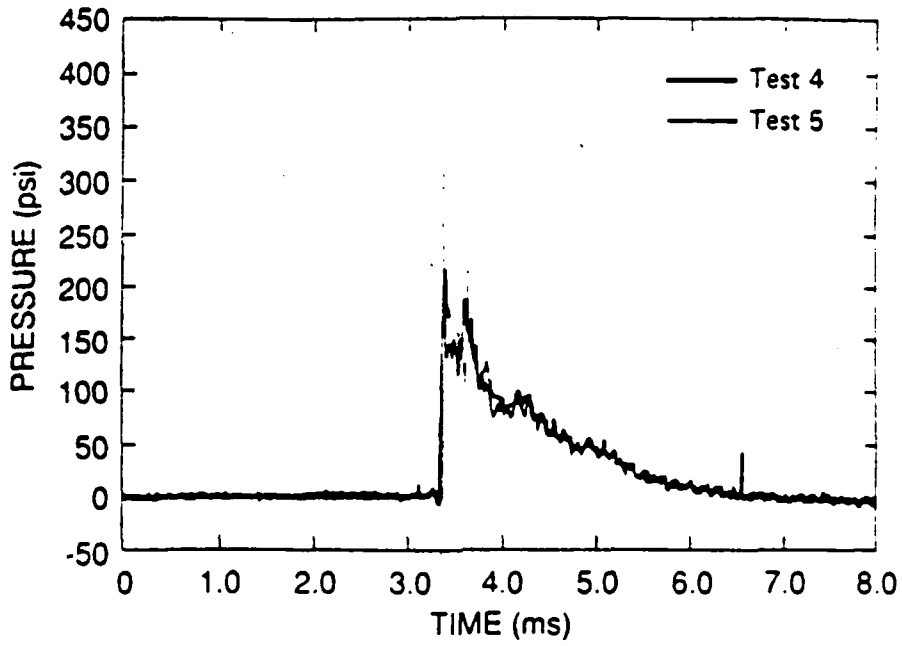


(a) Cross Section of Antitank Mine

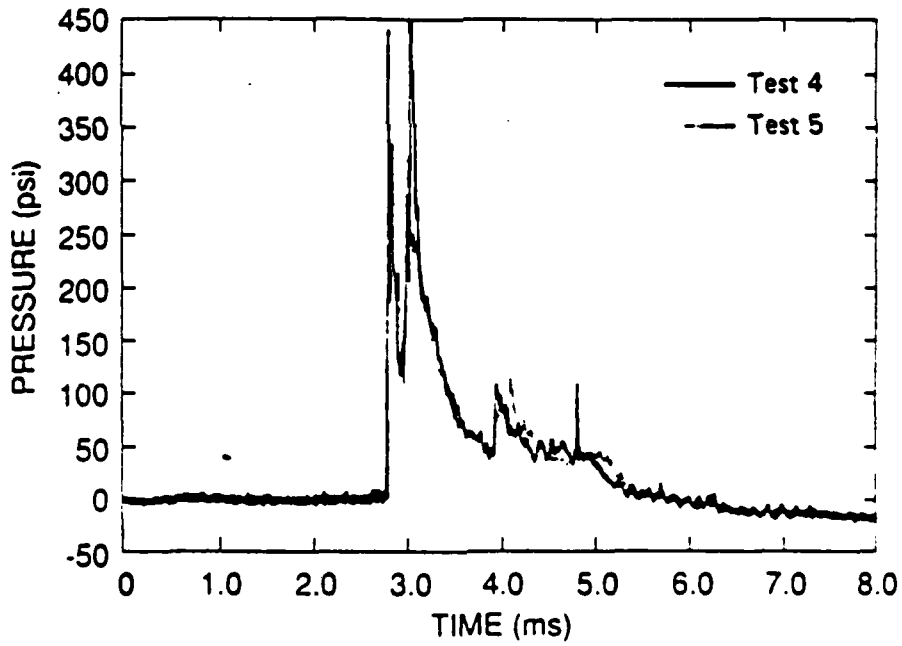


(b) Explosive Array and Instrumentation

Figure 2.4 Experimental configuration for antitank mine tests.

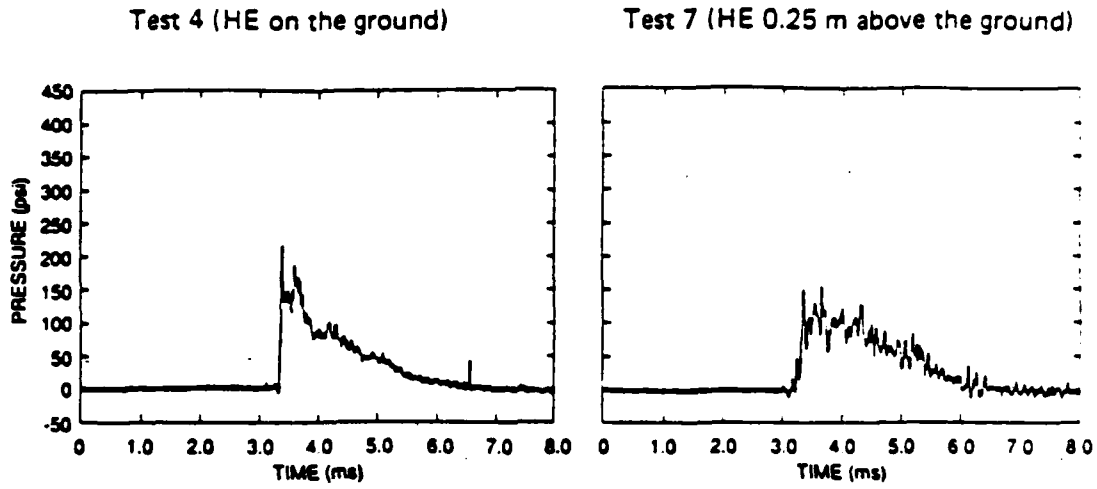


(a) Gage G4 (2 m above soil)

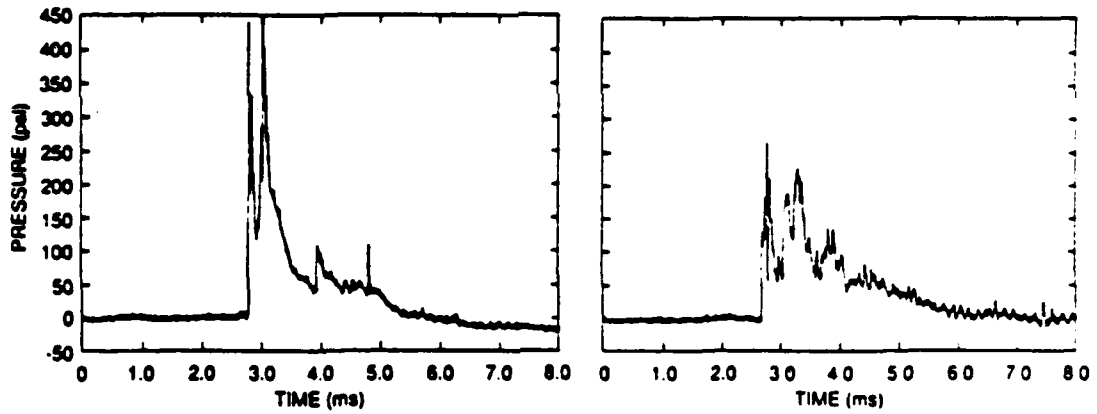


(b) Gage G7 (1 m above soil)

Figure 2.5 Comparison of tests 4 and 5.

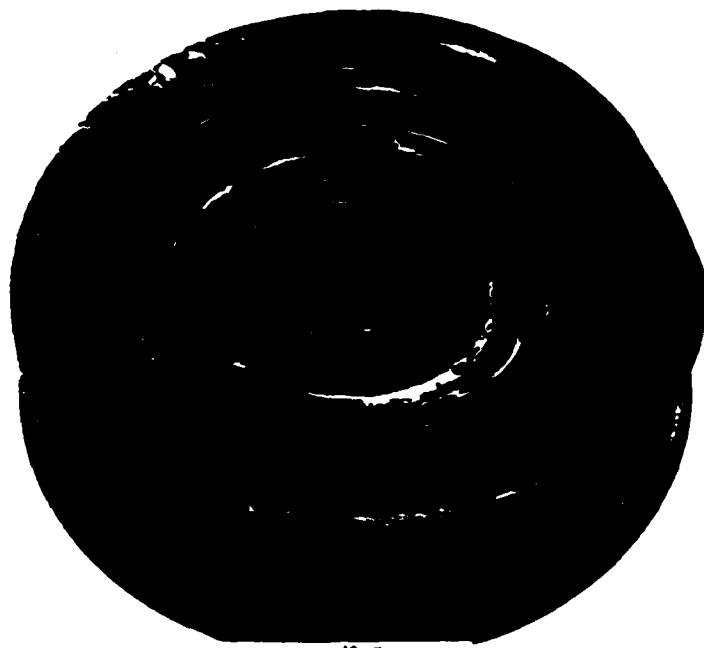


(a) Gage G4 (2 m above the ground)



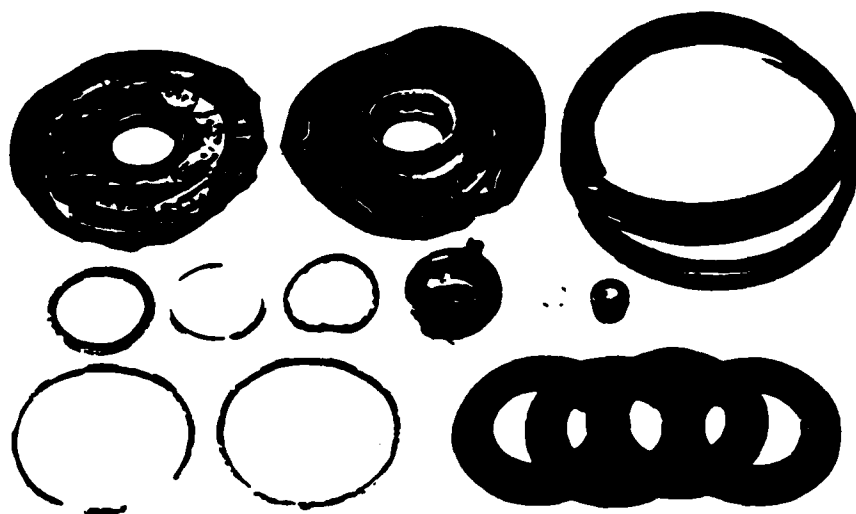
(b) Gage G7 (1 m above the ground)

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



9 ■ ■ ■ 10 ■ ■ ■

(a) Mine Casing



• ■ ■ ■ ■ ■

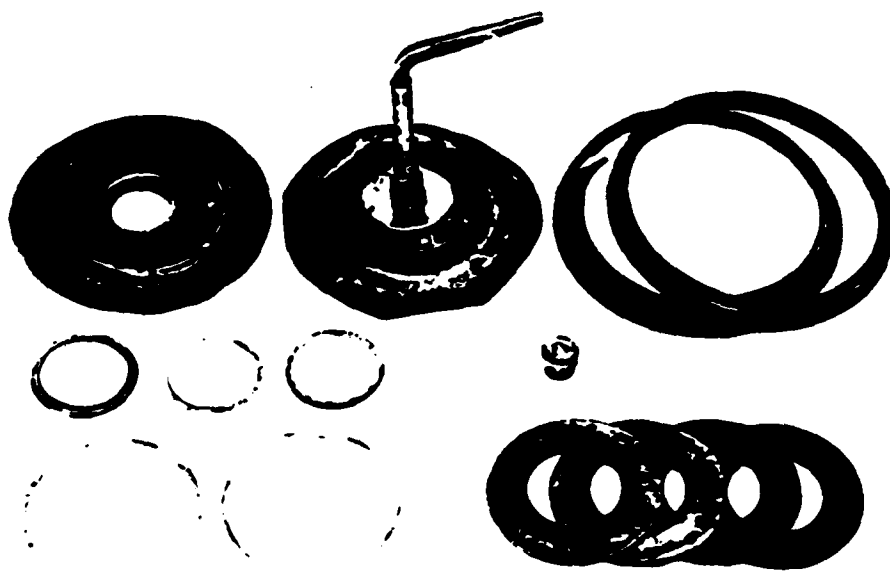
(b) Pressure Plate and Detonator Components

Figure 2.7 Damage to antitank mine in test 9.



10 ■ ■ ■ ■ ■

(a) Mine Casing

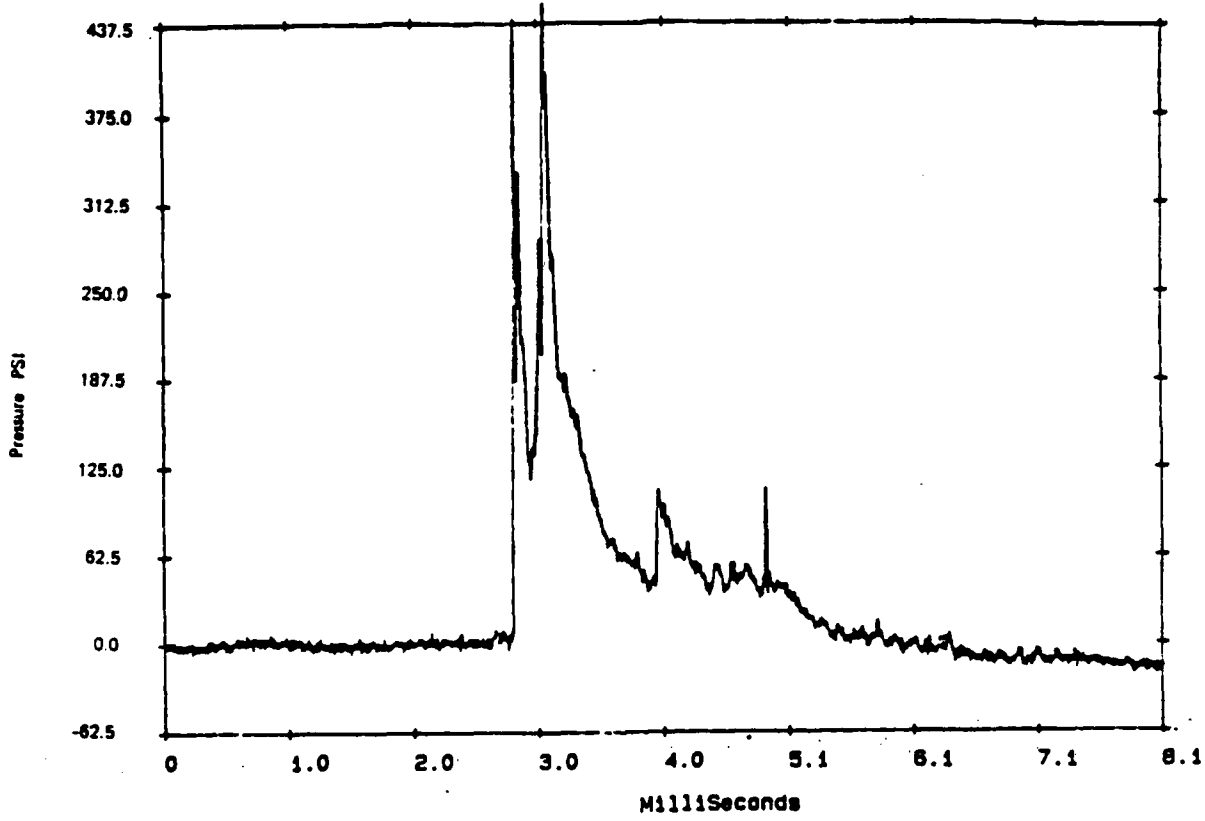


10 ■ ■ ■ ■ ■

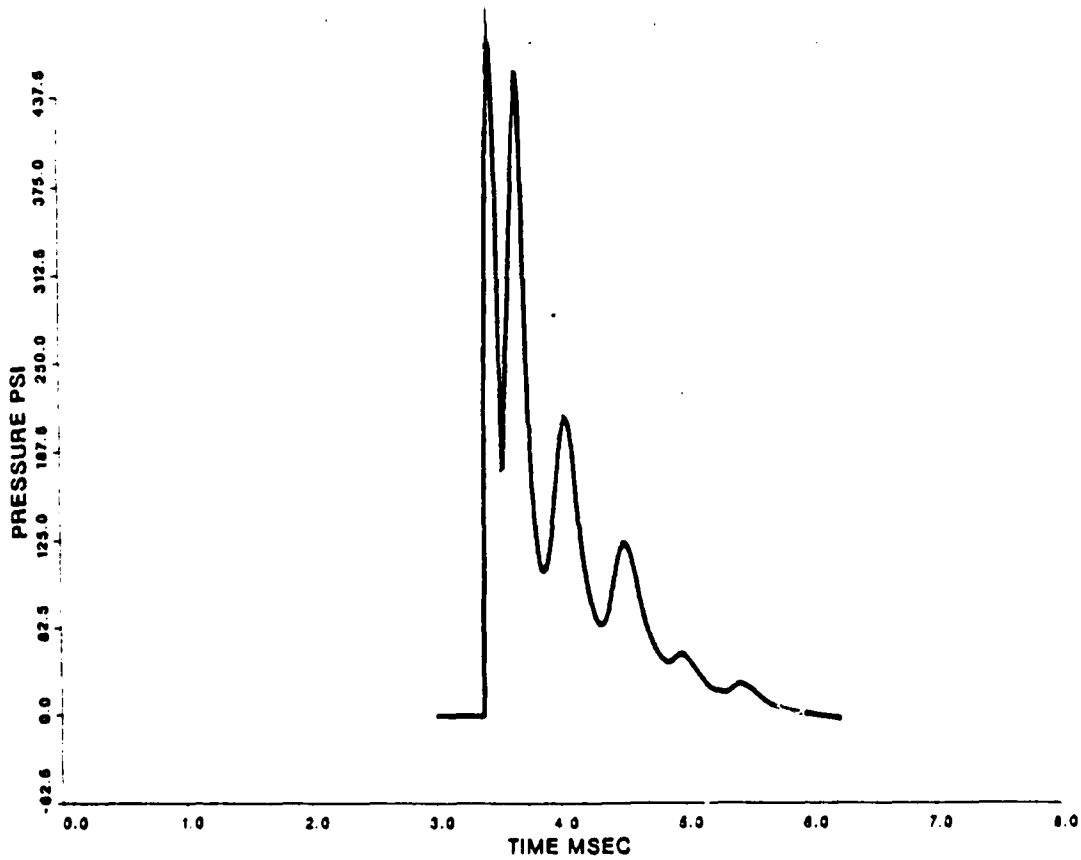
(b) Pressure Plate and Detonator Components

Figure 2.8 Damage to antitank mine in test 10.

Test 4 4-01-88



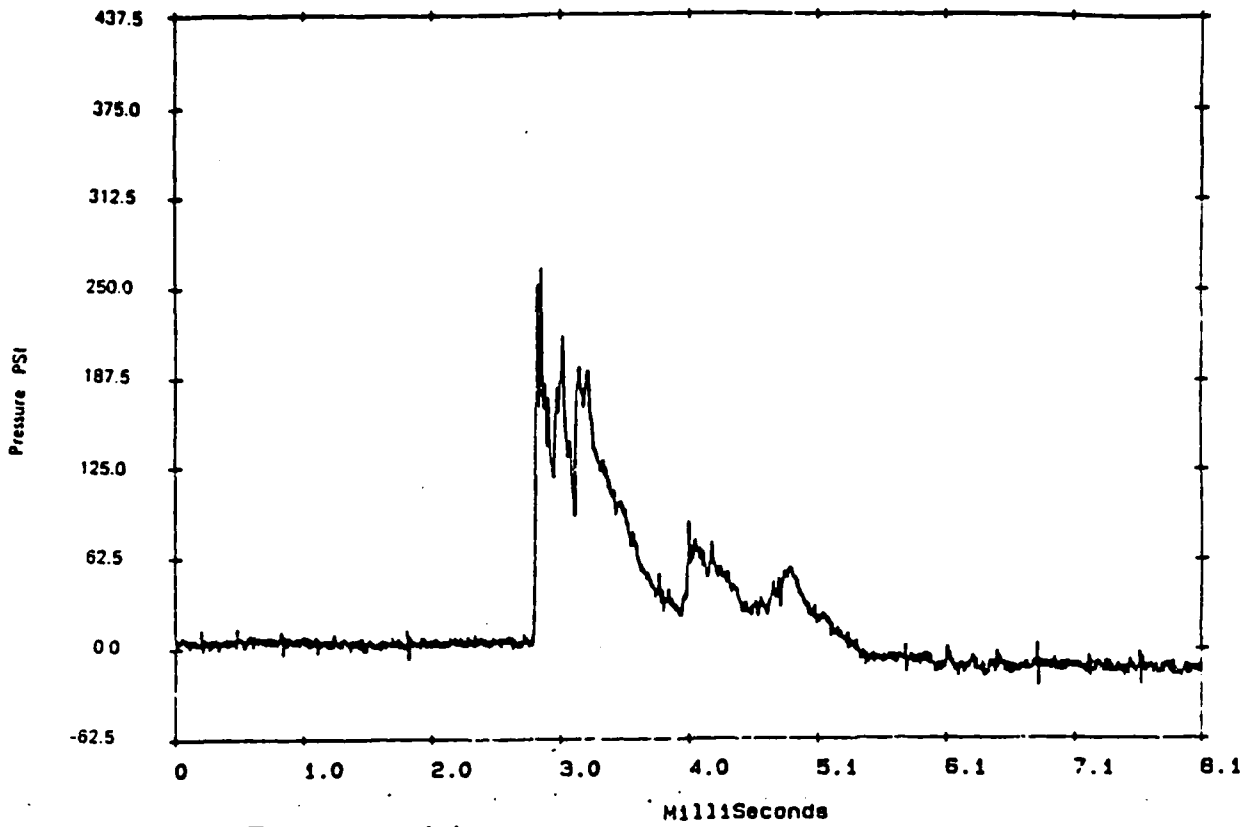
a . Experimental data



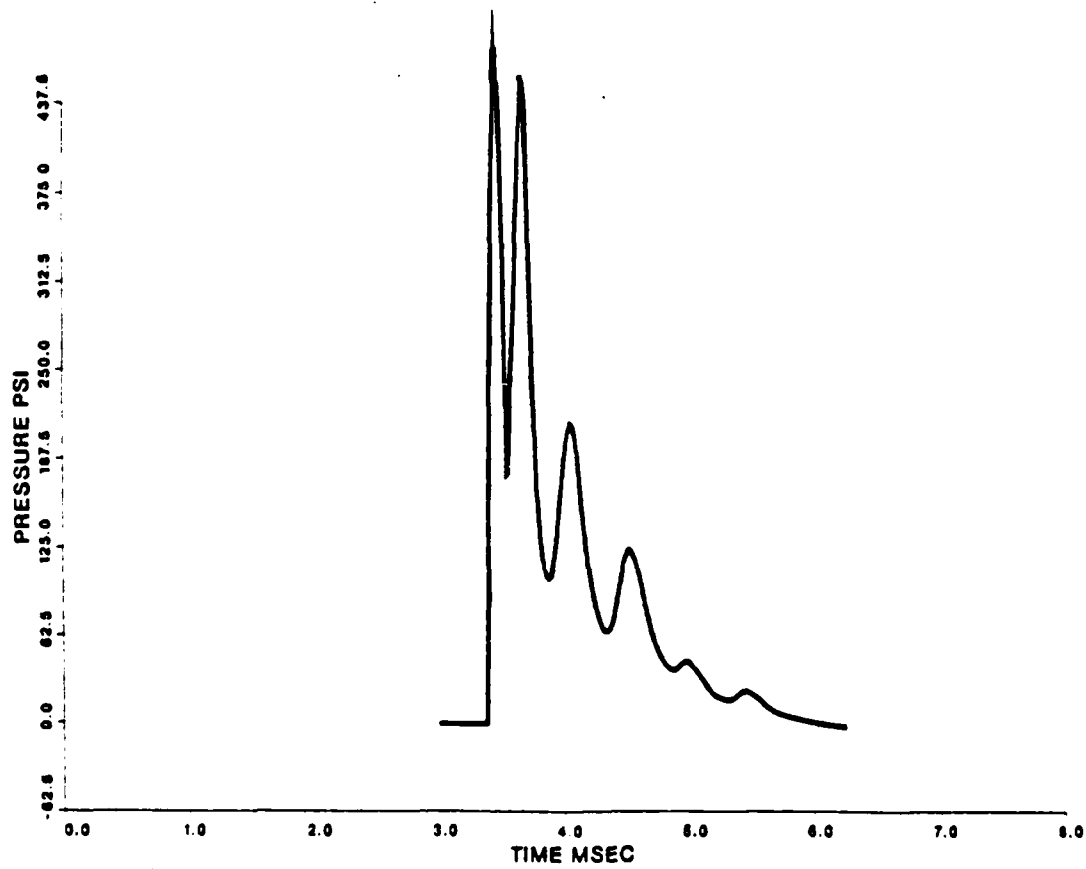
b . Computer simulation.

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

Test 4 4-01-86

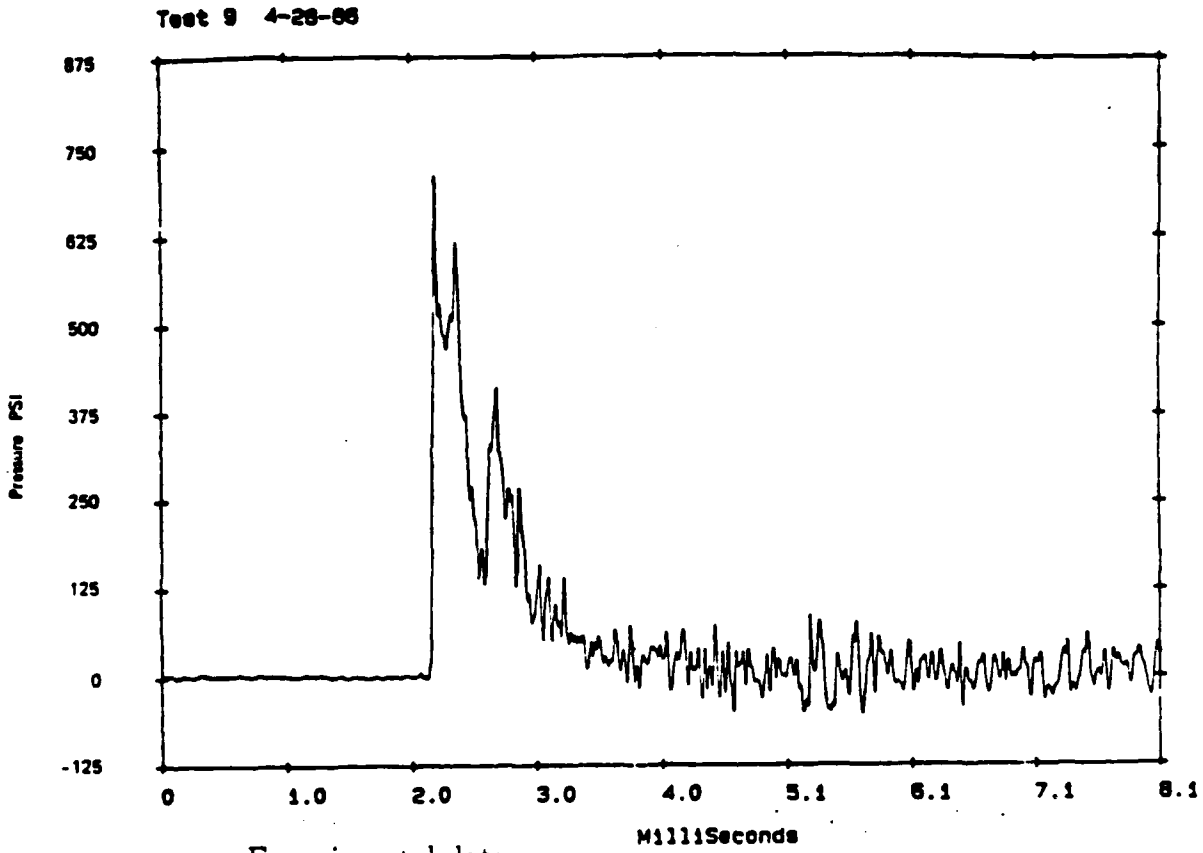


a . Experimental data

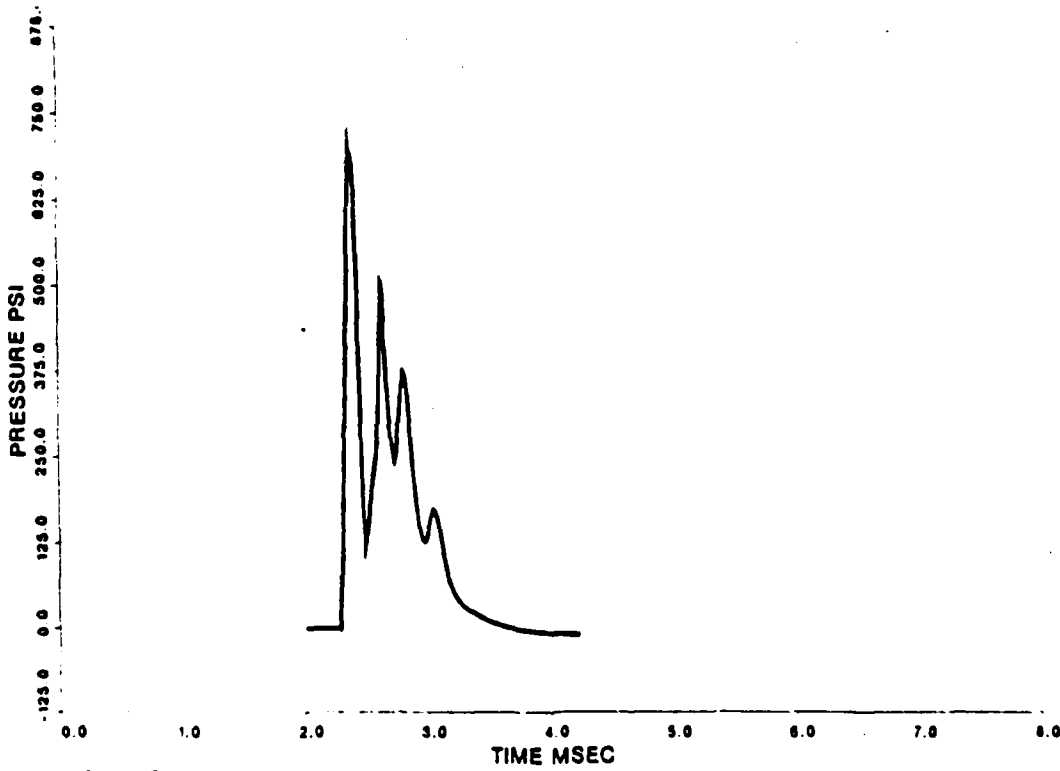


b . Computer simulation.

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



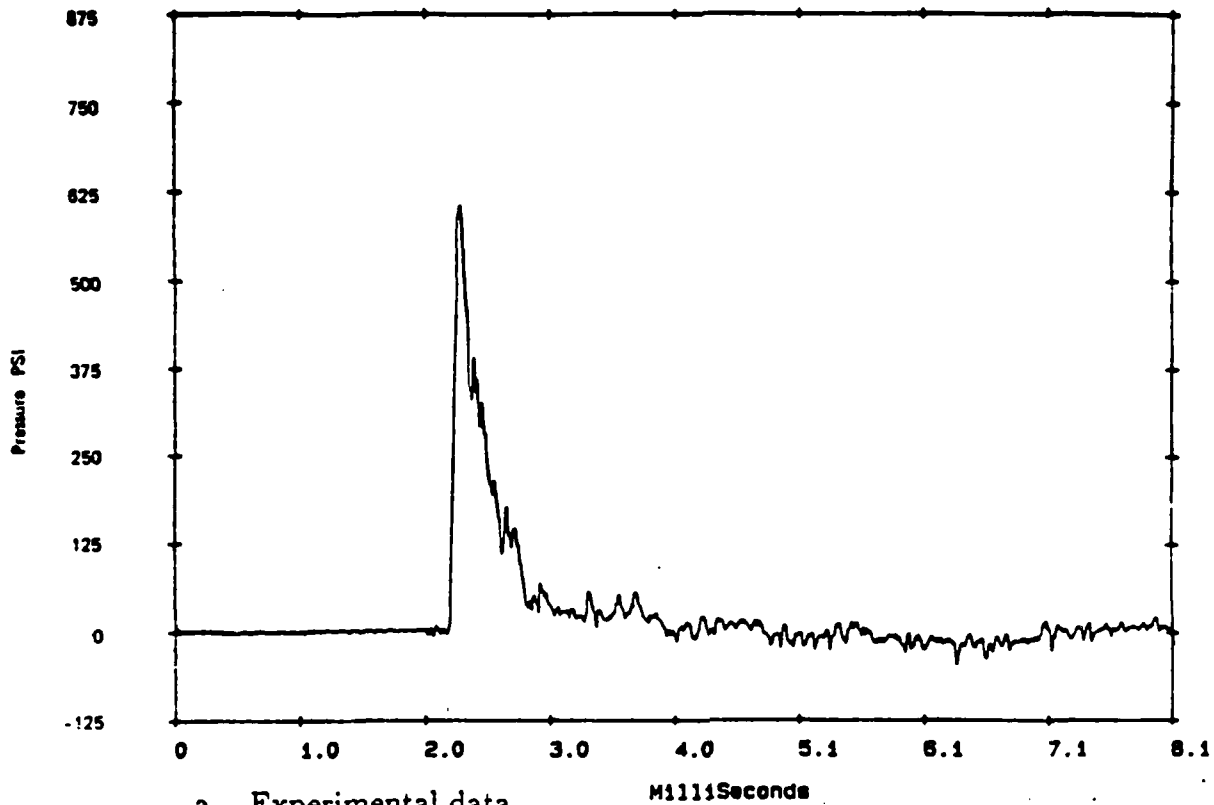
a . Experimental data



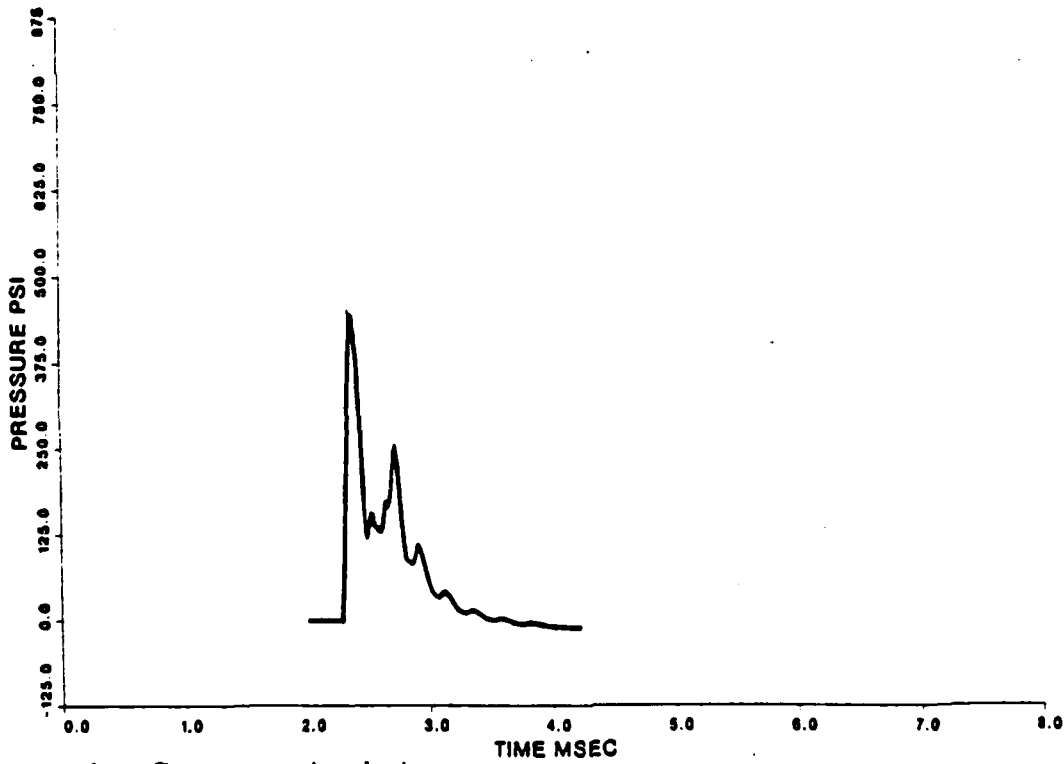
b . Computer simulation.

Figure 3.3 Comparison of data from gauge G3 (test 9) and simulation.

Test 9 4-25-66



a . Experimental data



b . Computer simulation.

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

test 9 4-28-65

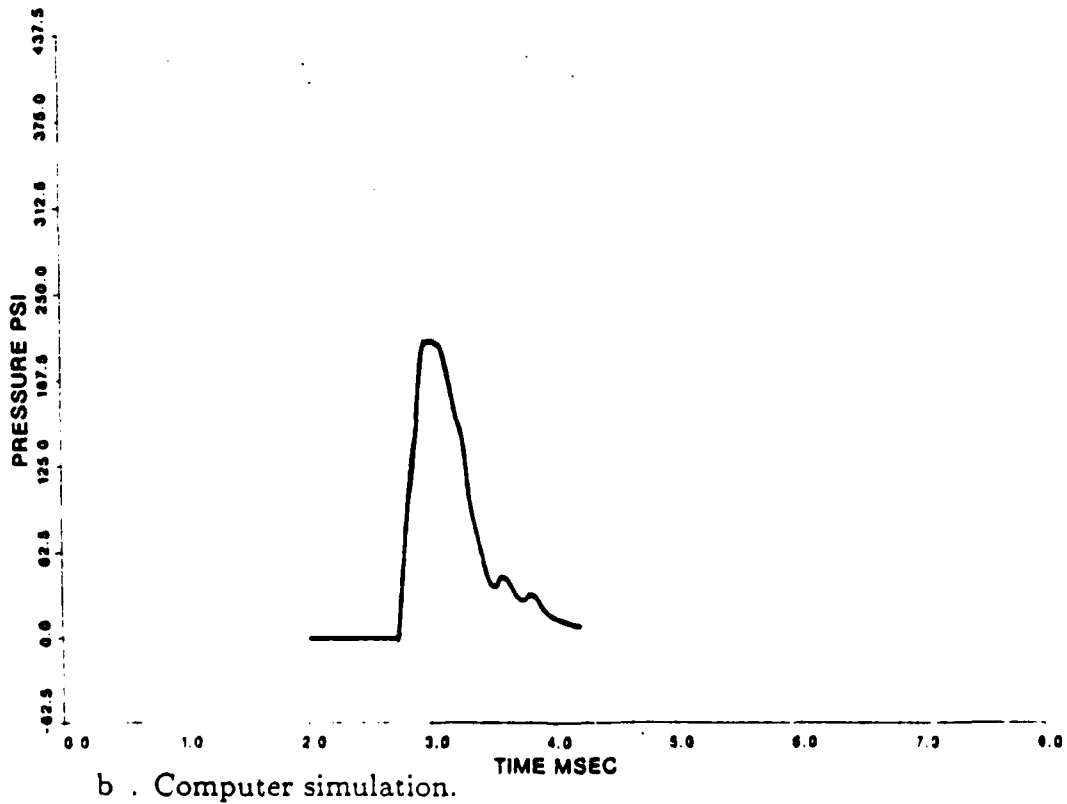
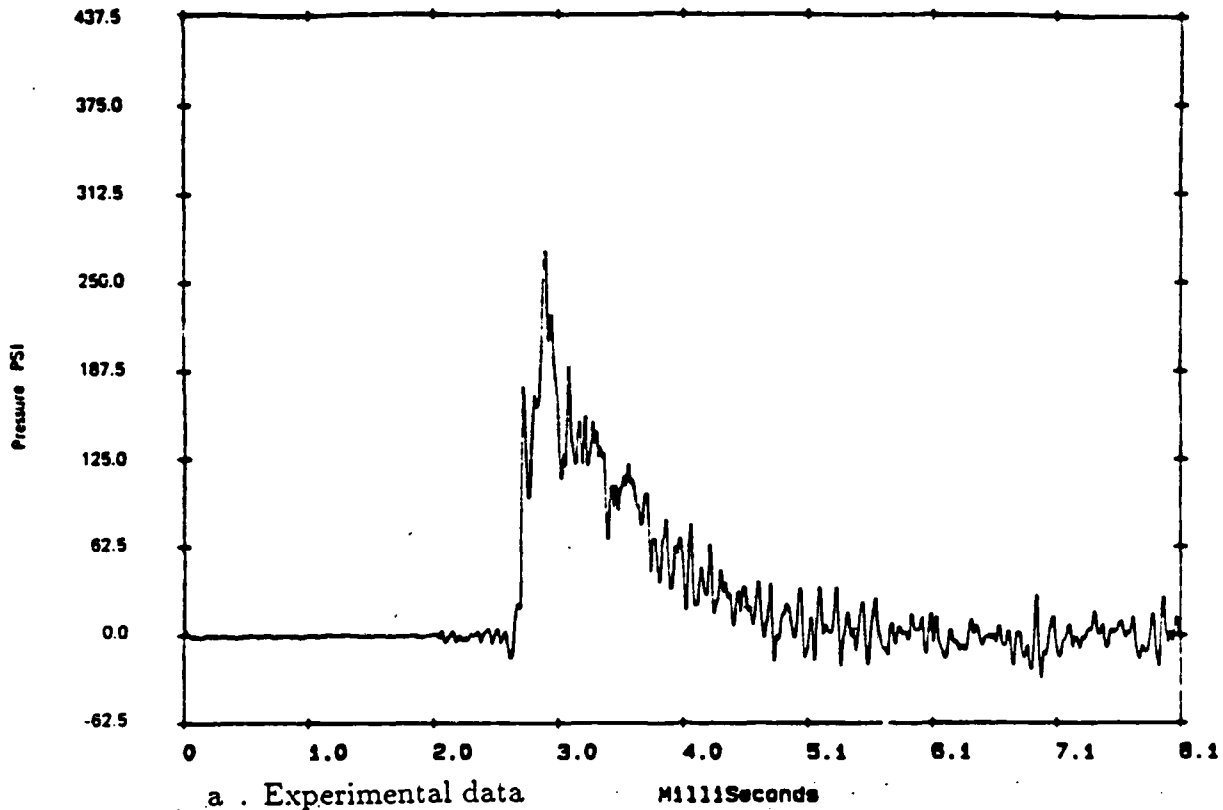
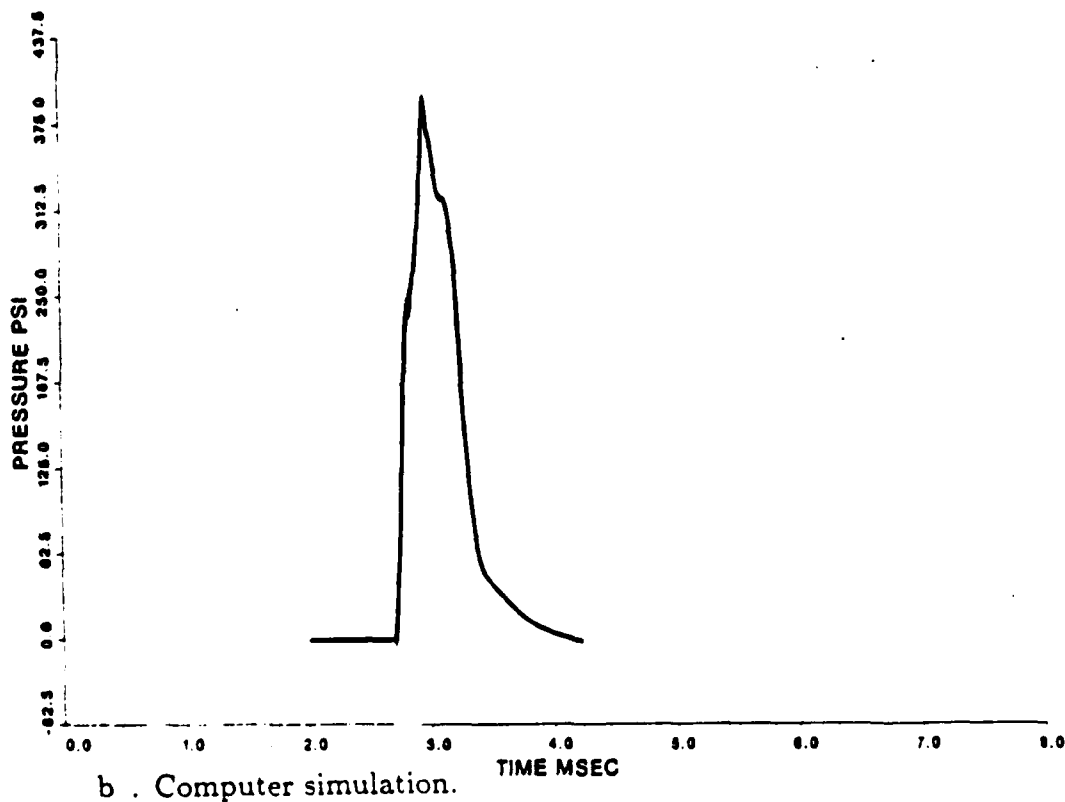
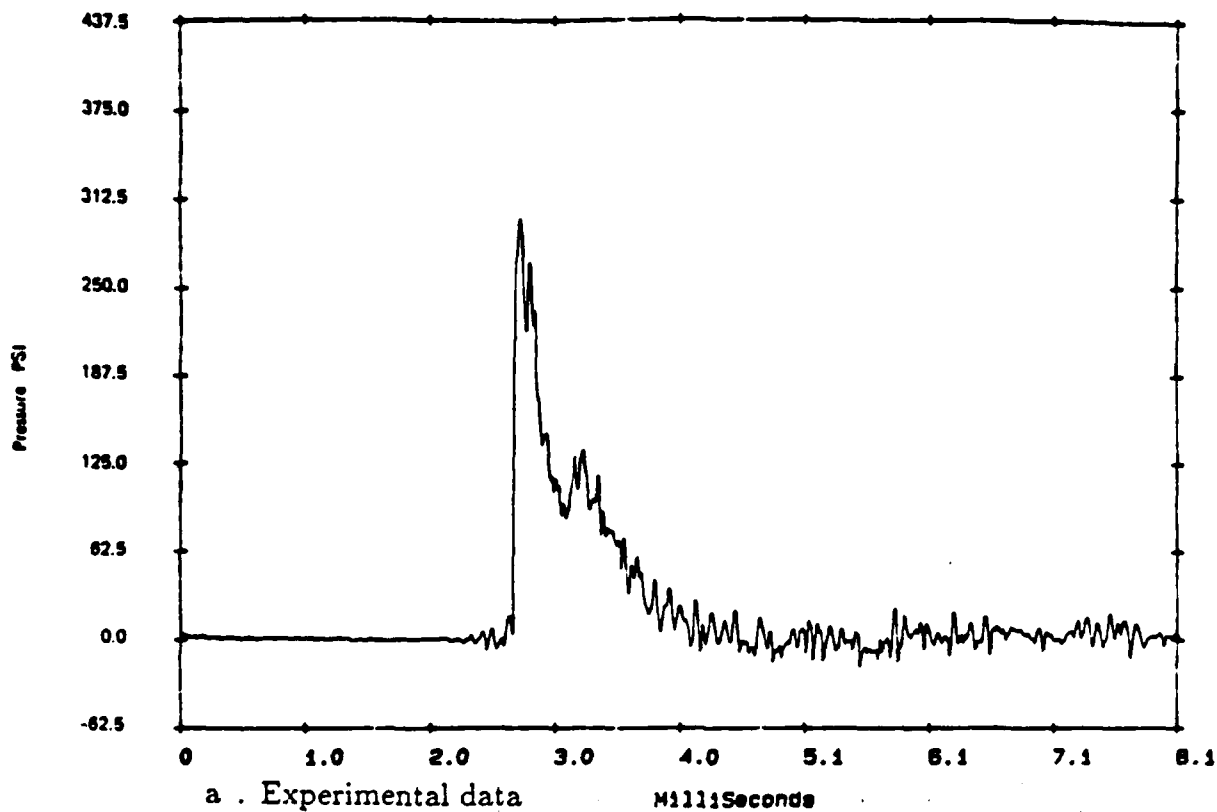


Figure 3:5 Comparison of data from gauge G4 (test 9) and simulation.

Test 9 4-28-88



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 developed utilizing a virtual system for data storage. The code called FAST3D solves partial differential equations approximated 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 processor 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

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

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

$$\frac{\partial E}{\partial t} + \nabla \cdot (E v) = \nabla \cdot (P v) + S - L. \quad (3)$$

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

$$P = P(e, \rho), \quad (4)$$

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 (monotonicity) 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

$$F_{ijk}^{td} = F_{ijk} + \frac{\Delta t}{\Delta x} [(Fu)_{i-1} - (Fu)_{i+1}] + \frac{\Delta t}{\Delta x} u_{i+1} (F_{i+1} - F_i) - \frac{\Delta t}{\Delta x} u_{i-1} (F_i - F_{i-1}) + C(F_{i+1} - 2F_i + F_{i-1}) + \text{other two directions} \quad (5)$$

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

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

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:

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

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

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

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

and the inactive zones,

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

$$\phi_{i+1/2} = \left(\frac{\Delta t}{\Delta x} u_i - C \right) (F_i^{td} - F_{i-1}^{td}). \quad (9b)$$

Using the difference between neighboring values of the transported diffused

quantities,

$$\Delta_{i+1/2} = F_{i+1}^{td} - F_i^{td}, \quad (10)$$

and the signs of the raw fluxes,

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

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

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

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

$$F_{ijk}^{\text{new}} = F_{ijk}^{td} + \phi_{i-1/2}^c - \phi_{i+1/2}^c + \text{other two directions}. \quad (13).$$

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 accuracy. 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 available in the longitudinal direction. A variable time step is used based on the minimum CFL limit in each direction over the whole mesh, $\Delta t < \min[\Delta x/(v + c), \Delta y/(v - c), \Delta z/(v + c)]$. 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, conventionally 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 implementation 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 arithmetic 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 vector hardware that exists on machines such as the Cray-1. When fully vectorized, FAST3D runs on the Cray-1 at about 100ns/(zone-step). The code has also been run on the Cray-XMP/12 at 150ns/(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 γ 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 presence of heated layers and obstacles. Helium gas layers are generally used to simulate 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 obstacle 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 shown 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 $\rho = 5.0 \times 10^{-7}$ which represents the helium layer. The intersection of the three axes 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 insufficient 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 systems 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-dimensional 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/cm³. 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

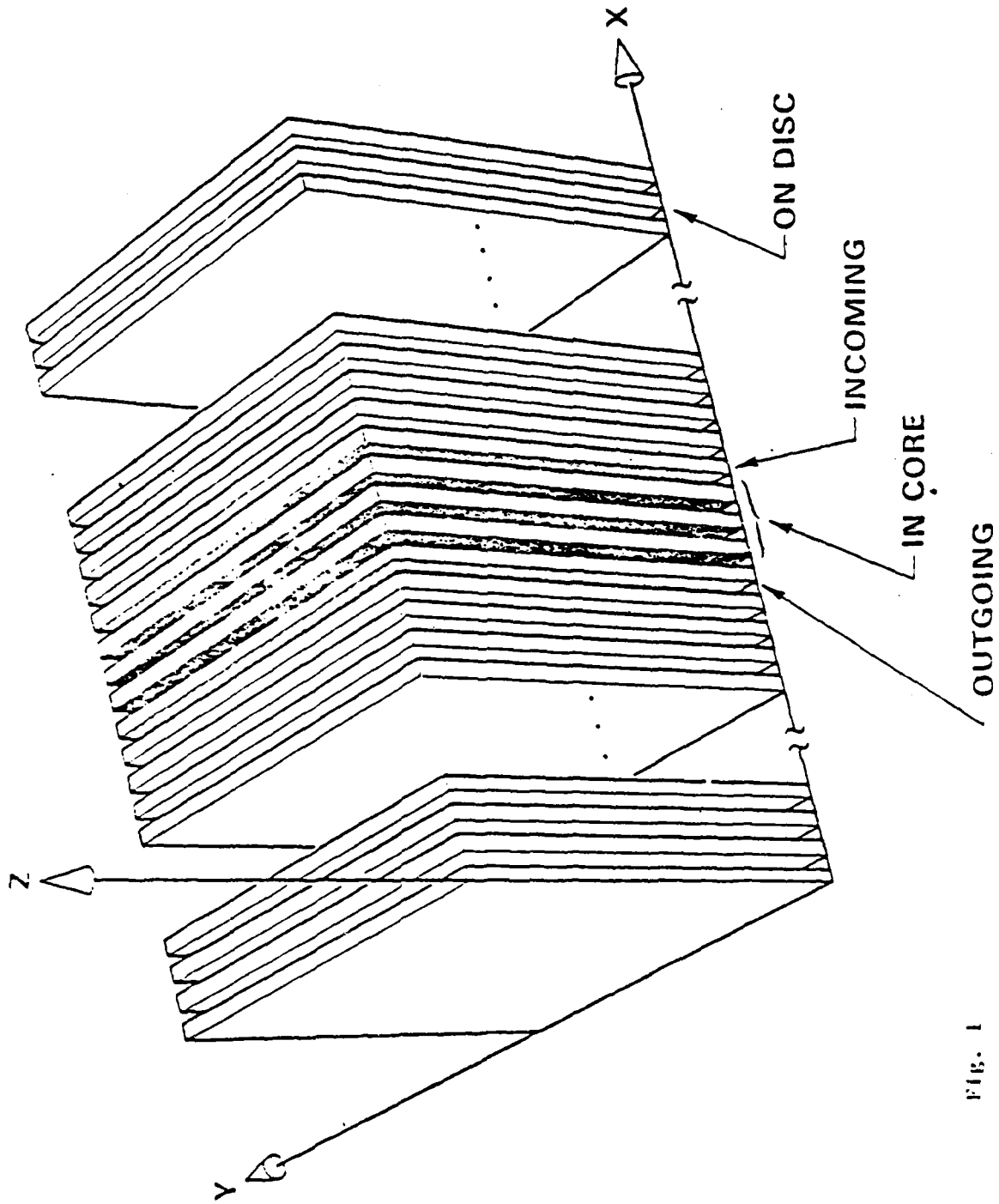
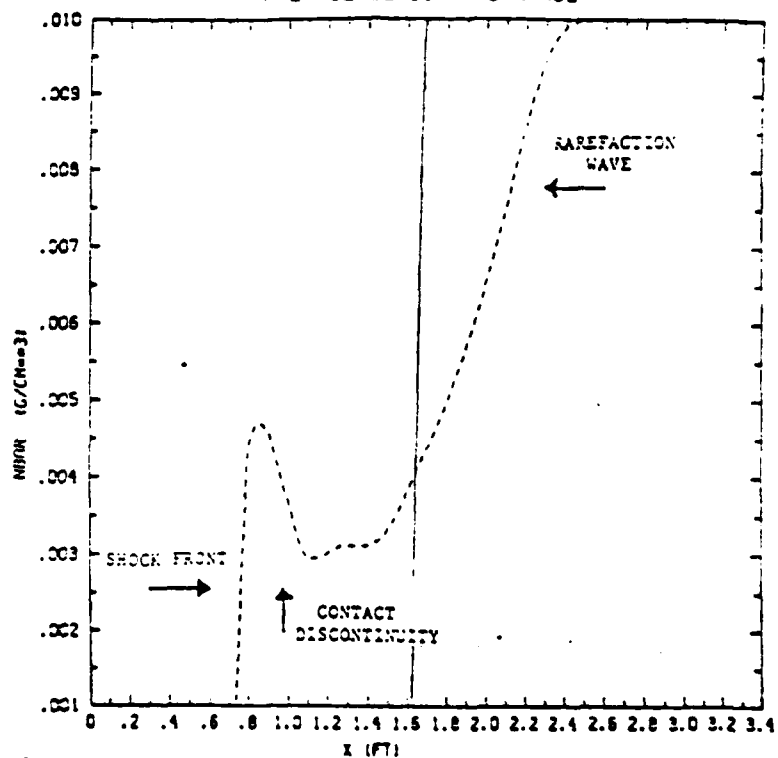


FIG. 1

RIEMANN PROBLEM
AVERAGE DENSITY VS RANGE



RIEMANN PROBLEM
AVERAGE PRESSURE VS RANGE

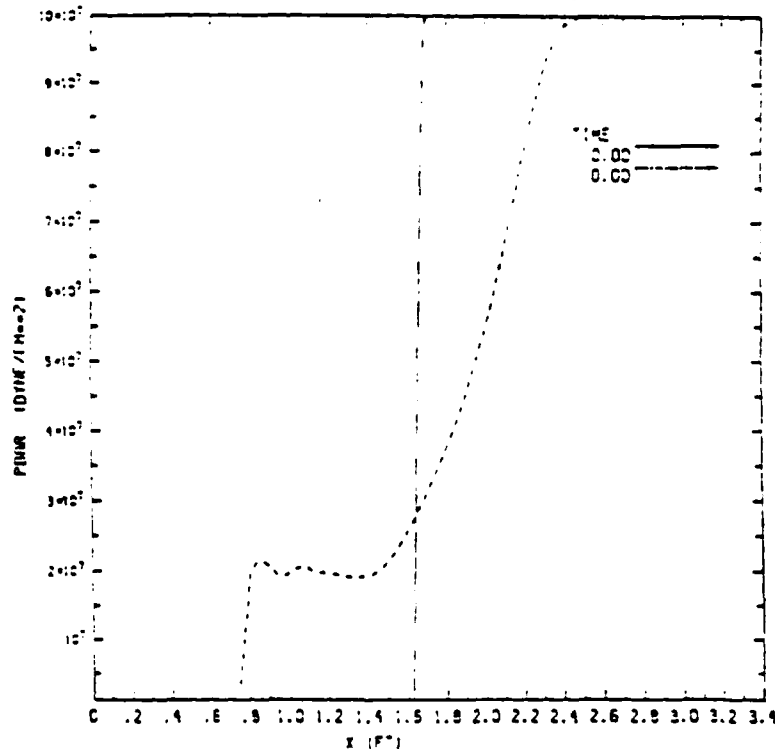
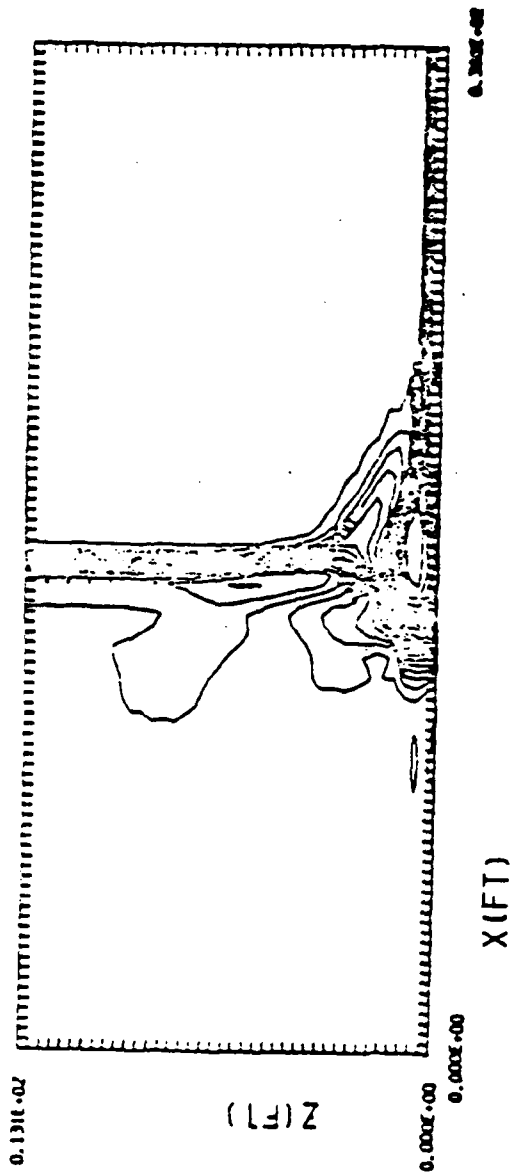


Fig. 2

CYCLE = 301 TIME = 0.01
DENSITY



PRESSURE

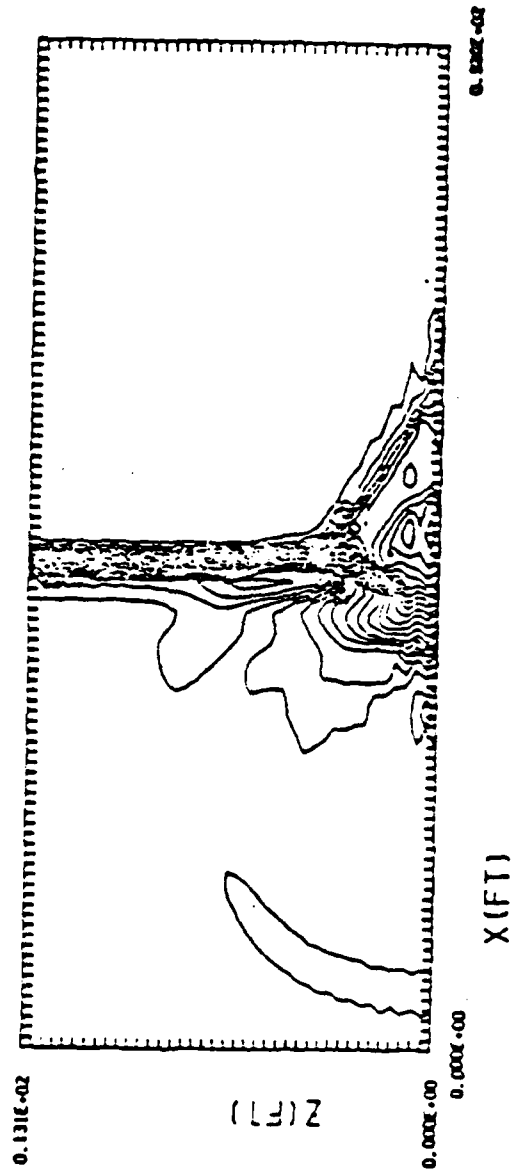
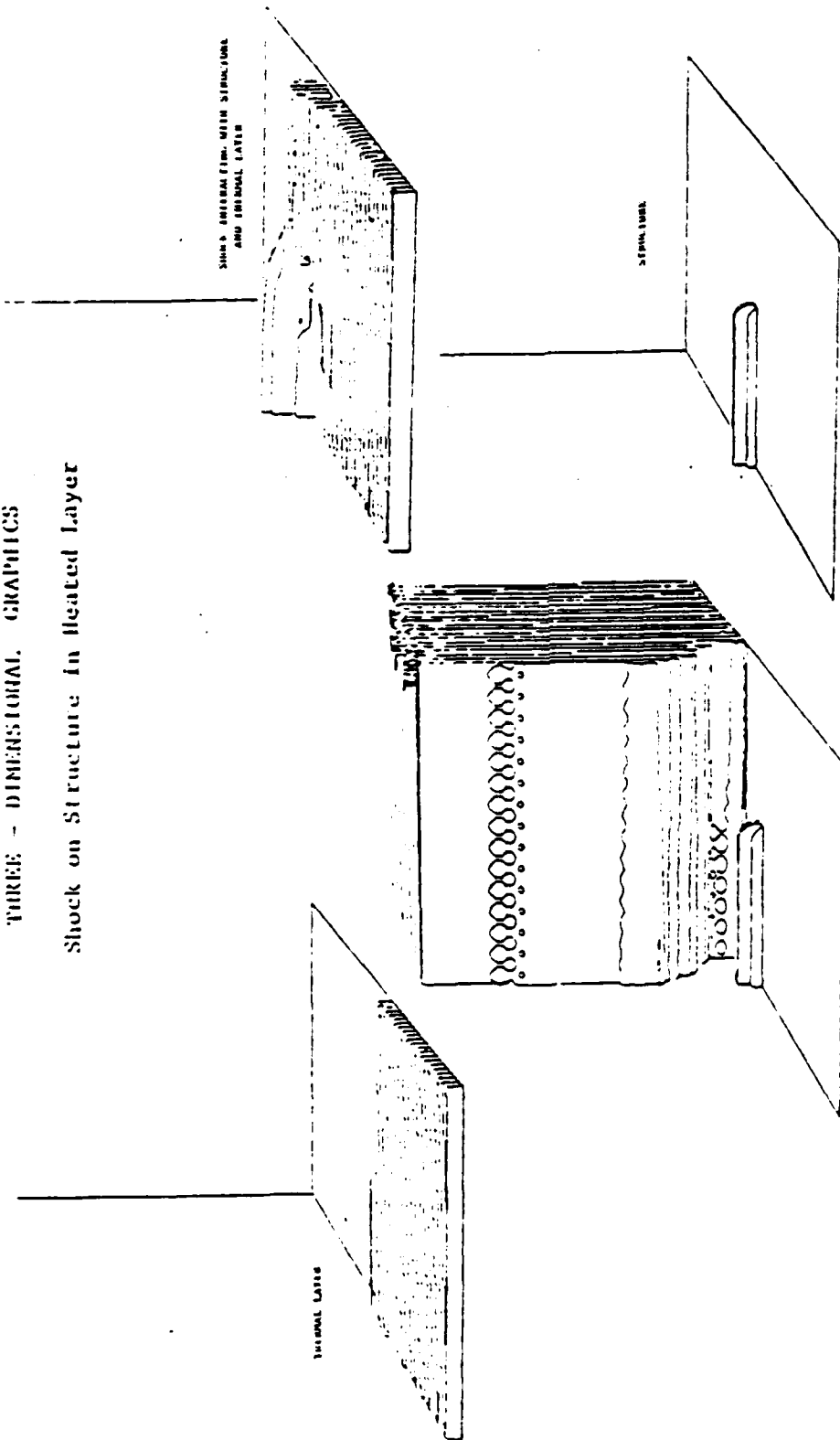


FIG. 3

THREE - DIMENSIONAL GRAPHICS

Shock on Structure in Heated Layer



SHOCK AND STIPPLES

FIG. 4

APPENDIX D

**COMPILATION LISTING OF
FLIK 3D
GRAPHICS CODE**

THE KURAN OF ILLIK3D

Illik3d, or Nplot3d as its current incarnation is named, is a graphics package created to plot the results of anblast 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 DISPLA (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 n20.00001
2 n20.00002
3 n20.00003
```

```
12 n20.00012
```

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(i)=1, then the x-y plane will be the first to be plotted. Nplot3d will plot the first nslice slices in msct.

mswitch - this is an array (14 by 3) of integers. Each value in mswitch represents one plot. If the value of mswitch(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 rectangular grid used in the calculations)

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

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

`llef` -- `llef` 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.

`lrig` -- `lrig`, similarly to `llef`, gives the rightmost boundary in the x direction for each slice.

`lflr` -- `lflr` gives the "frontmost" boundary of the region to be plotted in the z direction for a particular slice.

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

`lbot` -- `lbot` 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.

`dmth` -- this is the minimum contour level for the contour plot of density. If `dmth-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 `dmth`.

`displa` -- `displa` is a logical which determines which graphics library will be used. If `displa = true` then the graphs will be plotted using DISPLA. Other wise, NCAR will be used.

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

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

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

`ljlines` -- 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 subtler problem is that, in compiling and loading the code, the DISPLA library discset MUST be loaded first. The proper command would then be

```
ref t tplotid lib-discset.ncar-discset.o(fmath)
```

or

```
ref t tplotid  
ld lib-discset.ncar-discset.o(fmath)
```

If this is not done, then conflicts arise between `LEAR` and `DISPLA` commands with the same names.

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

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 `mplof` and switch `EVERYWRITE` in the program. Secondly, look for the `do` loops in `conv3d` and `prof3d`. They are of the form

```
do xxx ip:m,n
```

These `do` loops, and the assigned `qtos` 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 (lspec .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 `lchar` and `lmark` 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 `nmz`, `nmz1`, `ndm1`, `ndm2`, and `nny`. `Nmzx`, `nnyx`, and `nmz` 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 OVRWRITE 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 `plot`, or printed out on the electrostatic printer using `pspp`, or printed to film using `pfilm`. Good luck with this program. You may well need it!

Todd Brun
19 August 1986

Footnote

At the end of every plot file will be a blank page, present for reasons unknown. Update 11, 11's, handles.
Every color Vektor plot will be followed by a blank page. This is a

small bug in texture, 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 (ty,output (ty))
2
3  c
4  c main program for FAST3D graphics
5
6  c flik3d is a graphics package designed to print several different
7  c plots from information stored in a dumpfile, produced by an airblast
8  c simulation program. These plots are produced by two major routines,
9  c Conv3d (which produces contour plots, velocity vector plots, particle
10 c plots, and a template grid), and Prof3d (which produces various profile
11 c plots), supported by a number of other routines
12 c The graphics are drawn using two different graphics libraries:
13 c DLSBPA (by ISSCO) and NCAR. The library used is controlled by a
14 c logical switch 'display', which is read in in the namelist gdata and
15 c passed in the common block gform.
16 c The specifications of the plots are input from the file ndat. Most
17 c of these are passed in the namelist gdata. The array mswitch specifies
18 c which plots are to be drawn, while the switch displa determines which
19 c library is to be used. Nslice holds the plane which is to be plotted:
20 c x, y, z, or x z. For full explanation of the ndat file, see the
21 c flik3d Koran
22 c To run correctly, the source code must be compiled and loaded correctly.
23 c The proper format is
24 c ref i=plot3d lib=(discf,ncarcft,cgscft,cflmath)
25 c The discft library must be loaded first, in order for the program to run
26 c correctly.
27 c Many parts of this code are hardwired; they cannot be changed
28 c easily without altering the code in several places. Several examples
29 c are listed in the Koran. Several important ones: the number and order
30 c of the plots is controlled by the array mplot. If more plots are added,
31 c be sure to alter mplot's dimension EVERYWHERE. Also, one must alter
32 c a number of do statements (of the form do xxx (p=yyy,zzz) which control
33 c the order of plot calls.
34 c Furthermore, the size of the page and the subplot area is set, in
35 c the routine Conv3d. Great care should be exercised in changing them, as
36 c this will require that one re-scale a number of other calls, such as
37 c the SET calls which precede the velocity vector plot.
38 c Also, the line numbering is irregular, at best. Check to make sure
39 c that you have not duplicated any existing line numbers.
40 c Good luck with these graphics!!
41
42 c character *40 label
43 c character *8 readf, printf, namef, restf
44 c data readf /'ndat'/, printf /'nout'/, namef /'ndump'/
45
46 c
47 c station arrays
48
49 c parameter (mxx-1, nsta-1)
50 c parameter (mdimp-1, mdex=mdimp*mxx)
51
52 c logical lstat
53
54 c real rbs(mxx,nsta), vis(mxx,nsta), gms(mxx,nsta)
55 c real ps(mxx,nsta), lme(mxx), xs(nsta), ys(nsta), zs(nsta)
56 c common /stats/ lstat, rbs, vis, gms, ps, lme, xs, ys, zs, nxx
57 c real tim(mdex), psta(mxx,nsta), dsta(mxx,nsta)
58 c real star(mdex,nsta)
59 c pointer (upstar,star)
60
61 c local declarations-----
62
63 c parameter (paper=3)
64 c logical inter(paper), incore, linterp, colour, ladj, displa
65 c logical dabled
66

```

```

57 integer mswitch(14, npar), msect(npar), ilef(npar)
58 integer ncp(npar), jplot(npar), jbak(npar), kbot(npar)
59 integer ktop(npar), mplot(14), ilines(npar), jlines(npar)
60 namelist /pdata/ mslice, msect, mswitch, ilef, irig, jful, jbak,
61 kbot, ktop, nfiles, incore, colour, lstat, interp, dmin, dmax,
62 displa, nbn, i lines, j lines, darkhd
63 namelist /history/ ista, jsta, ichose, iadj
64 common /qfcom/ mslice, min, imax, jmin, jmax, kmim, kmax,
65 mplot, linterp, colour, dmin, dmax, displa, il, jl, darkhd
66
67 data i lines, j lines /npar+1, npar+1/
68
69 c
70 c open files.
71 c
72 c open (5, file=readf, status='old')
73 c
74 c open (6, file=prntf, status='new')
75 c
76 c open (7, file=namef, status='old')
77 c
78 c read in namelist data.
79 c this data specifies the types of plots to be drawn
80 c read (5, pdata)
81
82 c initialize graphics and NCAR or DISSPLA.
83 c call gplot (llu, 4hplot, 12)
84 c if (displa) then
85 c call lndrps
86 c else
87 c call labcar
88 c endif
89
90 c if (lstat) go to 120
91
92 c plot contours and velocity vectors.
93 c .....
94 c do 10 i 1, nfiles
95 c read (7, 2/0, end=110) restf
96 c call restf (lstep, incore, restf, label)
97 c
98 c count number of characters in label.
99 c nlabel=0
100 c do 10 i 1, 40
101 c nlabel=nlabel+1
102 c if (label(i) eq '$') go to 20
103 c continue
104 c 10
105 c 20
106 c if (incore) call readin
107
108 c process one slice at a time.
109 c the data controlling the appearance of the plots is read in from
110 c the file read in namelist pdata.
111 c do 100 l=1, mslice
112 c mslice=msect(lslice)
113 c imin=ilef(lslice)
114 c imax=iql(lslice)
115 c jmin=jbot(lslice)
116 c jmax=jbak(lslice)
117 c kmax=kbot(lslice)

```



```

113      fmax ktop(lslice)
114      f1=lines(lslice)
115      f2=lines(ustlice)
116      fimp=interp(lslice)
117      do 30 ip=1,14
118      30  mpot(ip)=mswicht(ip,lslice)
119
120      c
121      c      f111 the plot arrays and plot
122      c      go to (40,60,80), msllice
123
124      c
125      c      X Y slice.
126      c      Reads in the data and calls the routines to draw profile
127      c      and contour plots.
128      c      do 50 kk=kmin,kmax
129      c      call datain (imin,imax,jmin,jmax)
130      c      call prof3d (istep,restf,label,nlabel,kk)
131      c      continue
132      c      go to 100
133
134      c
135      c      Y Z slice.
136      c      Reads in the data and calls the routines to draw profile
137      c      and contour plots.
138      c      do 70 ii=imin,imax
139      c      call datain (ii,ii,jmin,jmax)
140      c      call prof3d (istep,restf,label,nlabel,ii)
141      c      continue
142      c      go to 100
143
144      c
145      c      X Z slice.
146      c      Reads in the data and calls routines to plot profile
147      c      and contour plots.
148      c      do 90 jj=jmin,jmax
149      c      call datain (imin,imax,ii,kmin)
150      c      call prof3d (istep,restf,label,nlabel,jj)
151      c      continue
152      c      continue
153
154      c      Close dump
155      c      call filefc
156      c      continue
157      c      go to 260
158
159      c
160      c      Plot station time histories
161      c      .....
162      c      continue
163      c      read (5,buistry)
164      c      kout=0
165      c      kfile=0
166      c      do 210 if=1,nfiles
167      c      read dump (file name and read data from dump file
168      c      read (1,270,end 210) read

```

```

169 8R call tstat (tstep, incore, costf, label)
170 8V call tfilec
171 90. kfile kstatf
172 C
173 C
174 91 determine number of characters in label.
175 92 nlabel=0
176 93 do 130 i=1,40
177 94 nlabel=nlabel+1
178 95 if (label(i:i) eq '$') go to 140
179 96 continue
180 97 continue
181 C
182 C
183 97. time
184 98 do 150 i=1,nxx
185 99 tim(i)=time(i)
186 100 k=kcount
187 101 patch time histories.
188 102 k=kcount
189 103 do 200 ksta=ista, jsta
190 104 istat=ksta
191 105 go to (160, 180, 160), ichose
192 C
193 C
194 104. static pressure.
195 105 do 170 i=1,nxx
196 106 pstat(i,k,istat)=prs(i,ksta)
197 107 if (ichose.eq.1) go to 210
198 C
199 C
200 107. dynamic pressure.
201 108 do 190 i=1,nxx
202 109 dstat(i,k,istat)=0.5*rhs(i,ksta)+vls(i,ksta)+vls(i,ksta)
203 110 continue
204 111 kount=knxx
205 C
206 C
207 111. plot time histories.
208 112 do 250 ksta=ista, jsta
209 113 istat=ksta
210 114 xstat=xstksta
211 115 ystat=ysksta
212 116 zstat=zsksta
213 117 length=l
214 118 time=tim(k)
215 119 if (ladj) call adjust (tim,psta(1,istat),kt,(max,length)
216 120 go to (220,230,220), ichose
217 C
218 C
219 120. static pressure.
220 121 upstat=loc(psta)
221 122 separate calls are made to label axes with DISSIPA and NCAR
222 123 if (diplot) then
223 124 call xname ('TIME (SEC)',10)
224 125 call yname ('PRESSURE (DYN/CM**2)',22)
225 126 else
226 127 call aunit(timtime, sep $,22)pressure, dyns/cm**2$,1,1,0,0)
227 128 endif
228 129 locall=1
229 130 go to 240

```

```

225 c
226 c
227 c
228 c
229 c
230 c
231 c
232 c
233 c
234 c
235 c
236 c
237 c
238 c
239 c
240 c
241 c
242 c
243 c
244 c
245 c
246 c
247 c
248 c
249 c
250 c
251 c
252 c
253 c
254 c
255 c
256 c
257 c
258 c
259 c
260 c
261 c
262 c
263 c
264 c
265 c
266 c
267 c
268 c
269 c
270 c
271 c
272 c
273 c
274 c
275 c
276 c
277 c
278 c
279 c
280 c
dynamic pressure.
ipstar=loc(dsta)
if (displa) then
  call xname ('TIME (SEC)',10)
  call yname ('DYNAMIC PRESSURE (DYNES/CM**2)',30)
else
  call annot(1,1,0,0)
  call annot(1,1,0,0)
endif
icall=2
c
c generate plots.
c Set up the titles and axes for the plots using DISSPLA or NCAR.
240 if (displa) then
  imx=imax(length,star(1,istat),1)
  call page (10.,11.)
  call area2d (6.,8.)
  encode (30,280,glab) star(imx,istat),tim(imx)
  call headin (glab,30,1,5,4)
  encode (32,300,glab) kfile,restf
  call headin (glab,32,1,5,4)
  encode (48,310,glab) ksta,xstat,ystat,zstat
  call headin (glab,48,1,5,4)
  encode (nlabel,290,glab) label(1:nlabel)
  if (darkhd) then
    call swissm
    call shdchr (90,1,0,003,1)
  endif
  call headin (glab,nlabel,2,1)
  call dispfr (tim,star(1,istat),length)
  if (icall.eq.1.and.iclose.eq.3) go to 230
else
  imx=imax(length,star(1,istat),1)
  encode (30,280,glab) star(imx,istat),tim(imx)
  call pwrtt (700,990,glab,30,1,0,0)
  encode (32,300,glab) kfile,restf
  call pwrtt (550,30,glab,32,1,0,0)
  encode (48,310,glab) ksta,xstat,ystat,zstat
  call pwrtt (550,50,glab,48,1,0,0)
  encode (nlabel,290,glab) label(1:nlabel)
  call ezyy (tim,star(1,istat),length,glab)
  if (icall.eq.1.and.iclose.eq.3) go to 230
endif
250 continue
c
c terminate graphics.
260 call qdone
c
c close files
275 close (5)
276 close (6)
277 close (7)
278 stop
c
c 270 format (17,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
      SHORT VECTOR LOOP BEGINS AT SIQ. NO.
      VECTOR LOOP BEGINS AT SIQ. NO.
      VECTOR LOOP BEGINS AT SIQ. NO.
      VECTOR LOOP BEGINS AT SIQ. NO.
      57, P= 551b
      97, P= 1013b
      104, P= 1047a
      107, P= 1070b

```

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

NUMRE

10 UNDEF**	43L	40E
10A530B	40L	
10RURDF**	40I	
20 5A4C	44I	42J
30 UNDEF**	58L	57E
30AUNDF**	57L	
30RURDF**	57I	
40 601b	60I	59J
50 UNDEF**	64L	60E
50A610B	60I	
50R635a	60I	59J
60 635c	66I	66F
70 URDF**	70L	
70A644C	66I	
70R670d	66I	
80 671b	72I	59J
90 UNDEF**	76L	72E
90A700b	72I	
90B725a	72I	
100 725a	77I	71J
100A551b	46I	46I
100B727d	46I	
110 730d	79L	37R
110A500.1	46L	36E
110B733c	36L	
120 734a	81I	35J
130 UNDF**	95I	92E
130A777a	92I	
130RURDF**	92I	
140 1013b	96I	94J
150 URDF**	98I	97E
150A1023a	97I	
150B1031b	97I	
160 1047a	104I	103J/2
170 URDF**	105L	104E
170A1057d	104I	
170B1066d	104I	
180 1070b	107L	103J
190 URDF**	108L	107I
190A1109b	107I	
190B1112a	107L	
200 UNDF**	109L	101E
200A1037a	101I	
200B1114d	101I	
210 1114d	110L	106J
210A744C	86L	R7R
210B1121b	86I	86I
220 1150c	120I	
230 1174c	165J	129I
240 1220b	137L	119J
		128J

250 UNDEF** 1671 111F
 250A1124B 111L
 250R1527C 111I
 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
 XXX02 473b 30I
 XXX03 474b 32L
 XXX04 546c 45I
 XXX05 577d 59W
 XXX06 1045c 103W
 XXX07 1142c 118L
 XXX08 1147a 119W
 XXX09 1165b 121I
 XXX10 1173b 124L
 XXX11 1211b 130I
 XXX12 1217b 133L
 XXX13 1367d 137I
 XXX14 1350c 148L
 XXX15 1524d 155I

(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
\$CIS	EXTERNAL			170U	169U	
\$FTA	EXTERNAL			163U	147U	143U
\$EFF	EXTERNAL			163U	159U	147U 143U
\$LTI	EXTERNAL			163U	161U	147U 143U
\$FV	EXTERNAL			161U/4	159U	145U 143U
\$OPW	EXTERNAL			27U	157U/2	141U/2
\$RIA	EXTERNAL			87U	26U	
\$RFI	EXTERNAL			87U	37U	
\$RFI	EXTERNAL			87U	37U	
\$RPI	EXTERNAL			82U	37U	
\$STOP	EXTERNAL			172U	28U	
ADJUST	EXTERNAL			118U		
ANOTAT	EXTERNAL			134U		
AREA2D	EXTERNAL			140U		
26 COLOUR	VARIABLE		GRFCOM	22D	20S	150
CONV3D	EXTERNAL			75U	63U	63U
34 DARKHD	VARIABLE		GRFCOM	148U	22D	20S 16D
DATAIN	EXTERNAL			73U	67U	61U
31 DISPLA	VARIABLE		GRFCOM	134U	130U	121U 30U 22D 20S 150
DISPRF	EXTERNAL			150U		
30 DMAX	VARIABLE		GRFCOM	22D	20S	20S
27 DMIN	VARIABLE		GRFCOM	22D	20S	20S
147 D51A	P	2DIM APPRAY		108S	108S	110
FZY	EXTERNAL			163U		

LINE#	OPER	TEXT	SYMBOL	ADDRESS	SYMBOL	ADDRESS	SYMBOL	ADDRESS	SYMBOL	ADDRESS
25	LINE#	1	VARIABLE	56S	22D	15D				
100	LINE#	1	LINE# FUNCT	129U	120U					
274	LINE#	1	VARIABLE	58U	56U	54U	53U	52U	51U	48U
				47U	46I					
				35U	20S	100	7D			
0	LS1AT		VARIABLE							
	MAIN		ENTRY							
	MIDMP		PARAMETER	65	11P/3	65				
	MDXX		PARAMETER	12P		19U				
7	MPLOT		1DIM ARRAY	58S	22D					
230	MSECT		1DIM ARRAY	47U	20S	17D				
0	MSELC		VARIABLE	59P	47S	22D				
156	MWICHI		2DIM ARRAY	58U	20S	17D				
	MXX		PARAMETER	9P/2	8P/3	55				
				27P	4S	3D				
143	NAMEI		VARIABLE	23D	20S					
0	NBU		VARIABLE	86N	36N	20S				
264	NFILES		VARIABLE	163U	163P	152P	147U	147P	93U	91S
272	NLABEL		VARIABLE	74P	69P	68P	63P	62P	41U	39S
				24S/2	19P/3	18P/4	17P/3	15P	14S	
263	NSLICI		PARAMETER	55U	46N	20S				
	NSTA		PARAMETER	12P	11P/2	9P/4	8P/3	5S		
11	NXX		VARIABLE	110U	107N	104N	99U	97N	100	
	PAGE		EXTERNAL	139U						
142	PRNIF		VARIABLE	26P	4S	3D				
	PROGRAM		EXTERNAL	74U	68U	62U				
4	PRS		2DIM ARRAY	105U	100	9D				
146	PSTA		2DIM ARRAY	120P	118P	105S	11D			
	PWRIT		EXTERNAL	162U	160U	158U				
141	READF		VARIABLE	25P	4S	3D				
	READIN		EXTERNAL	45U						
144	RESTF		VARIABLE	159U	143U	88P	87S	75P	74P	68P
				62P	38P	37S	3D			
				108U	38U	8D				
1	RHS		EXTERNAL	150U	157U	156P	153P	141U	138P	12D
	SHDCHR		EXTERNAL	164P	162U	158U				
1	STAR		2DIM ARRAY	149U						
	SWFS:M		EXTERNAL	164P	157U	153P	141U	118P	117U	11D
145	TIM		1DIM ARRAY	98U	100	9D				
313	TMAX		VARIABLE	108U/2	100	8D				
5	TIME		1DIM ARRAY	131U	122U					
2	VLS		EXTERNAL	117U	100	9D				
	XNAME		EXTERNAL	161U	145U	113S				
307	XS1AT		1DIM ARRAY	132U	123U					
	YNAME		EXTERNAL	114U	100	9D				
310	YS1AT		1DIM ARRAY	161U	145U	114S				
10	ZS		VARIABLE	115U	100	9D				
311	ZS1AT		1DIM ARRAY	161U	145U	115S				

TABLE OF EXTERNAL NAMES

SYMBOL	ADDRESS	SYMBOL	ADDRESS
\$CIS	170U	EXTERNAL	169U
\$EIA	167U	EXTERNAL	147U

Abbreviation	Description	Line	Source	Line	Source
EXTERNAL		1610	1590	1470	1410
EXTERNAL		1610	1590	1470	1410
EXTERNAL		1610/4	1570/2	1450/4	1430
EXTERNAL		260	250		
EXTERNAL		270			
EXTERNAL		370			
EXTERNAL		370			
EXTERNAL		370			
EXTERNAL		280			
EXTERNAL		1720			
EXTERNAL		1180			
EXTERNAL		1340			
EXTERNAL		1400			
EXTERNAL		750	630		
EXTERNAL		730	610		
EXTERNAL		1530			
EXTERNAL		1640			
EXTERNAL		890			
EXTERNAL		1680			
EXTERNAL		290			
EXTERNAL		1520	1440	1420	
EXTERNAL		1560			
EXTERNAL		310			
EXTERNAL		330			
EXTERNAL		1390			
EXTERNAL		740			
EXTERNAL		10			
EXTERNAL		1620			
EXTERNAL		450			
EXTERNAL		880			
EXTERNAL		1500			
EXTERNAL		1490			
EXTERNAL		1310			
EXTERNAL		1320			

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

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

TABLE OF PARAMETERS ENCOUNTERED

MDMP	= 1	MDXX	= 1
MXZ	= 1	NPAR	= 3
NSIA	= 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 LP	57	58		INLINE
50 KK	60	64	610d	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
240 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-STAT5	35-GRFCOM	1-NBUR
B SAVE:	31				
I SAVE:	4				
CONSTANTS:	132				
VARIABLES:	165				
TEMPORARIES:	535				
CODE:	1225				
TOTAL:	2336				

STATIC SPACE (IN OCTAL)

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

```

286 1. C      subroutine conv3d (istep,restf,label,nlabel,lplane)
287 C
288 C      graphics for FAS3D.
289 C      Conv3d plots, at the present time, 7 distinct graphs. The graphs to
290 C      plotted have a non zero value in their positions in the array mplot.
291 C      The values mplot(8) through mplot(14) refer to the 7 plots in this
292 C      subroutine.
293 C      The plots are, in order, a contour plot of renormalized density,
294 C      a contour plot of energy/volume, a contour plot of pressure (absolute)
295 C      a contour plot of mach number, a velocity vector plot, a plot of trace
296 C      particles used in the calculation, and a template grid to show the real
297 C      spacing of the irregular grid used in the calculations.
298 C      Fick3d can handle both regularly and irregularly spaced grids. If
299 C      an irregularly spaced grid is used, the flag Interp will be set,
300 C      and a call will be made to the subroutine conint to interpolate onto an
301 C      regular grid.
302 C      Also, contour plots can be done in color. If colour is specified,
303 C      the plots will be drawn by the routine colcon. Otherwise, they will be
304 C      drawn within conv3d itself, using either DISSPLA or NCAR calls, and
305 C      calling on the shorter routing disccon for DISSPLA.
306 C      Warning: a large array in blank common is used by the DISSPLA
307 C      routine CONMAK. Do not use blank common for anything without allowing
308 C      for that block.
309 C
310 C      array dimensions
311 C      parameter (nmnx=72, nmny=33, nmnz=88)
312 C      parameter (nmnxp=nmnx+1, nmny2=nmny+1, nmnz2=nmnz+1)
313 C      parameter (nmnx2=nmnx+2, nmny2=nmny+2, nmnz2=nmnz+2)
314 C      parameter (ndm1=72, ndm2=88)
315 C      real vx(ndm1,ndm2), vy(ndm1,ndm2), vz(ndm1,ndm2)
316 C      real eq(ndm1,ndm2)
317 C      real rho(ndm1,ndm2), rho(ndm1,ndm2), rhh(ndm1,ndm2)
318 C      common /plvar/ vx, vy, vz, erg, rho, rha, rhh
319 C      -- fixed arrays of hydrodynamic data
320 C      parameter (nspec=1, nsum=4*nspec)
321 C      real ecamb(nmz2), pcamb(nmz2), ecamb(nmz2)
322 C      real sumpl(nsum,nmz)
323 C      real rcmn(nmz2), gravz(nmz2), gravz(nmz2), fsky(nmz2)
324 C
325 C      common /hydro/ rcamb, pcamb, ecamb, sumpl, rcmn, gravz,
326 C      1 gravz, sum4, shrink, fsky, ratio
327 C      grid arrays
328 C      real xcor(nmz), ycor(nmz), zcor(nmz)
329 C      real twodx(nmz), twody2(nmz2), twodz2(nmz2)
330 C      real dtodx(nmz), dtody2(nmz2), dtodz2(nmz2)
331 C
332 C      common /qdvat/ xcor, ycor, zcor, twodx, twody2, twodz2, dtodx
333 C      1 , dtody2, dtodz2
334 C      global variables
335 C      common /qlover/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
336 C      1 , nstep, llix, lily, lllz
337 C      scratch space
338 C      parameter (myz2d=nmz2*nmz2, nvar=4*nspec)
339 C      parameter (ndm12=(ndm1+2)*(ndm2+2))
340 C      parameter (ndm1p2=ndm1*ndm2)
341 C      real dmv(ndm12,nvar), dmy(ndm12)

```

```

342 common /holder/ space(ndm1,ndm2),valmin,vaireg,
343       space2(ndm1,ndm2)
344 common /loccont/ dumv, dumy
345
346 real art(ndm1,ndm2), beta(ndm1), vcor(ndm2)
347 real h(ndm1p2,2), v(ndm1p2,2)
348 real art(ndm1,ndm2), av(ndm1,ndm2)
349 pointer (ipar,arc), (ipicor,hcor), (ipvcor,vcor)
350 pointer (iparb,arb), (iparv,arv)
351 pointer (iparrh,arh), (iparrh,maxh)
352 pointer (iparrv,arv), (iparrv,maxv)
353
354 local declarations
355 real pre(ndm1,ndm2), mac(ndm1,ndm2), gam(ndm1,ndm2)
356 real vint(ndm1), scr1(ndm1,ndm2), scr2(ndm1,ndm2)
357 equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
358 equivalence (scr1,dumv(1,4)), (scr2,dumv(1,5)), (eint,dumy)
359 real pscr(ndm1), pscr(ndm1), gscr(ndm1)
360 real sscr(ndm1), sscr(ndm1), sscr(ndm1)
361 equivalence (pscr,scr1), (pscr,scr1(1,2)), (pscr,scr1(1,3))
362 equivalence (sscr,scr1(1,4)), (sscr,scr1(1,5)), (sscr,scr2)
363
364 c
365 c The particles are set up for two bursts only; changing the number of
366 c bursts could cause an error in data statements for ichar and imark.
367
368 parameter (nop=70,nburst=2)
369 real x(nop,nburst), zp(nop,nburst)
370 integer nopp(nburst), nch(nop,nburst)
371 common /nbur/ xp,zp,nopp,nch
372
373 integer mplot(14)
374 character *40 label
375 character *8 restf
376 integer ichar(nburst), imark(nburst)
377 real glab1(7), glab2(7), glab3(7), glab4(7)
378 logical ltmp, colow, displa, darkhd
379 data ksh, ksv /30,30/
380 namelist /qfdat/ mslice,imin,imax,jmin,jmax,kmin,kmax,mplot
381       ,ltmp,colow
382 common /qfcom/ mslice,imin,imax,jmin,jmax,kmin,kmax,
383 mplot,ltmp,colow,imin,dmax,displa,ll,jl,darkhd
384
385 c
386 c first print labels.
387 encode (label,250,glab1) label(1:ntlabel)
388 encode (40,260,glab2) ltmp,istep,restf
389
390 c
391 c set window sizes.
392 go to (10,20,30), mslice
393
394 c
395 c set up the pointers to specify the coordinates of the given slice
396 c X Y slice
397
398 c
399 c
400 c
401 c
402 c
403 c
404 c
405 c
406 c
407 c
408 c
409 c
410 c
411 c
412 c
413 c
414 c
415 c
416 c
417 c
418 c
419 c
420 c
421 c
422 c
423 c
424 c
425 c
426 c
427 c
428 c
429 c
430 c
431 c
432 c
433 c
434 c
435 c
436 c
437 c
438 c
439 c
440 c
441 c
442 c
443 c
444 c
445 c
446 c
447 c
448 c
449 c
450 c
451 c
452 c
453 c
454 c
455 c
456 c
457 c
458 c
459 c
460 c
461 c
462 c
463 c
464 c
465 c
466 c
467 c
468 c
469 c
470 c
471 c
472 c
473 c
474 c
475 c
476 c
477 c
478 c
479 c
480 c
481 c
482 c
483 c
484 c
485 c
486 c
487 c
488 c
489 c
490 c
491 c
492 c
493 c
494 c
495 c
496 c
497 c

```

```

398      ipminv=loc(jmin)
399      ipmaxv=loc(jmax)
400      ipacor=loc(xcor)
401      ipvcor=loc(ycor)
402      iparh=loc(rvz)
403      iparv=loc(rvz)
404      if (displa) then
405          encode (28,270,glab3) lplane,zcor(lplane)/100000.
406      else
407          encode (32,275,glab3) lplane,zcor(lplane)
408      endif
409      go to 40
410
411      C
412      C
413      C
414      C
415      C
416      C
417      C
418      C
419      C
420      C
421      C
422      C
423      C
424      C
425      C
426      C
427      C
428      C
429      C
430      C
431      C
432      C
433      C
434      C
435      C
436      C
437      C
438      C
439      C
440      C
441      C
442      C
443      C
444      C
445      C
446      C
447      C
448      C
449      C
450      C
451      C
452      C
453      C

```

Y Z slice.

X-Z slice.

define plot window.

what section of the plane is to be drawn?

```

454      n1 = lv / 11
455      n1j = nt / n1
456      kh = th
457      kv = tv
458      if (linterp) then
459          delh = hmax / hmin
460          delv = vmax / vmin
461          kh = ndm1
462          kv = delh / float(kh / 1)
463          kv = float(delv / div)
464          if (kv .le. ndm2) go to 45
465          kv = ndm2
466          div = delv / float(kv / 1)
467          kh = float(dh / div)
468      endif
469      call wsize (hmin,hmax,vmin,vmax,hlo,hhi,vlo,vhi)
470
471      c
472      c generate contour and vector plots.
473      c
474      c ispec=1
475      c do 230 ip=8,14
476      c for each plot, set up the page and titles.
477      c The DISPLA commands set the type of print, the size of the page,
478      c the size of the plot area, frame the axes and label them. write the
479      c titles, and prepare for drawing the plots. the NCAR commands draw
480      c the titles, set the size of the subplot area, frame it and draw
481      c tickmarks.
482      c
483      c 50
484      c if (display) then
485      c   if (darkhd) then
486      c     call swissm
487      c     call slschr (90.,1.0 (003.1)
488      c     endif
489      c     call page (10.,11.)
490      c     call nbfchr
491      c     call area2d (8.*(hmax-hmin)/(vmax-vmin),8 )
492      c     call height (.16)
493      c     call hwidth (glab1,nlabel,1,1,6,3)
494      c     call hwidth (glab2,n,1,6,3)
495      c     call hwidth (glab3,n,1,6,3)
496      c     call xtime (' ',1)
497      c     call ytime (' ',1)
498      c     call hbfm (.02)
499      c     call frame
500      c     ystep = (vmax - vmin)/8.
501      c     xstep = ystep
502      c     call graf (hmin/100000.,xstep,hmax/100000.,
503      c     vmin/100000.,ystep,vmax/100000.)
504      c     flo 0
505      c     fhi 0
506      c     else
507      c     call pwt1 (500,1010,glab1,nlabel,1,0,0)
508      c     call pwt1 (350,980,glab2,40,1,0,0)
509      c     call pwt1 (500,950,glab3,28,1,0,0)
510      c     call sct (hlo,hhi,vlo,vhi,hmin,hmax,vmin,vmax,1)
511      c     call pwt1 (1,0,1,0)
512      c     flo 0

```

```

510      fh=0.
511      endif
512      C
513      go to (60,100,'10,140,170,400,420), ip-7
514      C
515      renormalized (111) density contours.
516      these plots are scaled w/ respect to the ambient density. Three
517      species of density can be drawn (from ispec). These are usually
518      total density, dust density, and explosive density. Usually only
519      species 1 or species 1 and 2 are used.
520      60      go to (70,80,90), ispec
521      C      first, assign the pointer
522      70      ipart=loc(space2)
523      do 72 nv,minv,maxv
524      iscr(nv) =vmgt((rcamb(ipart),rcamb(nv,1),mslice,eq,1)
525      do 72 nh,minh,maxh
526      space2(nh,nv)=rho(nh,nv)/rscr(nv)
527      space2(nh,nv)=vmgt(1.0,space2(nh,nv),space2(nh,nv),qt,0.95
528      1      .and. rho(nh,nv),11.1.05)
529      encode (18,300,glab4)
530      C      label the plot
531      if (display) then
532      tlenq = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
533      call message (glab4,17,tlenq,'75)
534      else
535      call pwrite (800,980,glab4,18,1.0,0)
536      endif
537      go to 160
538      C      first, assign the pointer (species 2)
539      80      ipart=loc(rha)
540      encode (20,310,glab4) ispec
541      C      label the plot
542      if (display) then
543      tlenq = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
544      call message (glab4,19,tlenq,'75)
545      else
546      call pwrite (800,980,glab4,20,1.0,0)
547      endif
548      flo=dmin
549      fhi=dmax
550      go to 160
551      C      first, assign the pointer (species 3)
552      90      ipart=loc(rhb)
553      encode (20,310,glab) ispec
554      C      label the plot
555      if (display) then
556      tlenq = (hmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
557      call message (glab4,19,tlenq,'75)
558      else
559      call pwrite (800,980,glab4,20,1.0,0)
560      endif
561      go to 160
562      C
563      C      energy contours.
564      C      first, assign pointer to the energy array
565      100      ipart=loc(eng)

```

```

566      encode (20,320,glab4)
567      label the plot
568      if (displa) then
569          tlenq = (lmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,19)/2.
570          call messaq (glab4,19,tlenq, .75)
571      else
572          call pwrnt (800,980,glab4,20,1.0,0)
573      endif
574      go to 160
575
576      c
577      c pressure contours.
578      c pressure is calculated in the equations of state routine eos
579      c fvs4 assign pointer to pressure array pre
580      c
581      encode (23,330,glab4)
582      label the plot
583      if (displa) then
584          tlenq = (lmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,22)/2.
585          call messaq (glab4,22,tlenq, .75)
586      else
587          call pwrnt (800,980,glab4,23,1.0,0)
588      endif
589      do 130 nv=mnv,maxv
590      do 120 nh=mnh,maxh
591          erq(nh,nv) = 0.5*(rvx(nh,nv)+rvx(nh,nv)+rvy(nh,nv)+rvy
592          (nh,nv)+vz(nh,nv)+vz(nh,nv))/rho(nh,nv)
593          call eos (mach,mnht,1,spec,rho(mnh,nv),eint(mnh),gam(mnh
594          ,nv),pre(mnh,nv),pscr(mnh),qscr(mnh),scra(mnh),scrb(mnh)
595          ,scre(mnh),scre(mnh),rba(mnh,nv),rbb(mnh,nv))
596      continue
597      go to 160
598
599      c
600      c mach number contours.
601      c first assign pointer to mach number array mac
602      c
603      encode (11,340,glab4)
604      label the plot
605      if (displa) then
606          tlenq = (lmax-hmin)/(vmax-vmin)*4.0 - xmess(glab4,11)/2.
607          call messaq (glab4,11,tlenq, .75)
608      else
609          call pwrnt (800,980,glab4,13,1.0,0)
610      endif
611      calculate pressure from eos to use in calculating mach number
612      do 140 nv=mnv,maxv
613      do 135 nh=mnh,maxh
614          erq(nh,nv) = 0.5*(rvx(nh,nv)+rvx(nh,nv)+rvy(nh,nv)+rvy
615          (nh,nv)+vz(nh,nv)+vz(nh,nv))/rho(nh,nv)
616          call eos (mach,mnht,1,spec,rho(mnh,nv),eint(mnh),gam(mnh
617          ,nv),pre(mnh,nv),pscr(mnh),qscr(mnh),scra(mnh),scrb(mnh)
618          ,scre(mnh),scre(mnh),rba(mnh,nv),rbb(mnh,nv))
619      continue
620      go to 160
621
622      c
623      c
624      c
625      c
626      c
627      c
628      c
629      c
630      c
631      c
632      c
633      c
634      c
635      c
636      c
637      c
638      c
639      c
640      c
641      c
642      c
643      c
644      c
645      c
646      c
647      c
648      c
649      c
650      c
651      c
652      c
653      c
654      c
655      c
656      c
657      c
658      c
659      c
660      c
661      c
662      c
663      c
664      c
665      c
666      c
667      c
668      c
669      c
670      c
671      c
672      c
673      c
674      c
675      c
676      c
677      c
678      c
679      c
680      c
681      c
682      c
683      c
684      c
685      c
686      c
687      c
688      c
689      c
690      c
691      c
692      c
693      c
694      c
695      c
696      c
697      c
698      c
699      c
700      c
701      c
702      c
703      c
704      c
705      c
706      c
707      c
708      c
709      c
710      c
711      c
712      c
713      c
714      c
715      c
716      c
717      c
718      c
719      c
720      c
721      c
722      c
723      c
724      c
725      c
726      c
727      c
728      c
729      c
730      c
731      c
732      c
733      c
734      c
735      c
736      c
737      c
738      c
739      c
740      c
741      c
742      c
743      c
744      c
745      c
746      c
747      c
748      c
749      c
750      c
751      c
752      c
753      c
754      c
755      c
756      c
757      c
758      c
759      c
760      c
761      c
762      c
763      c
764      c
765      c
766      c
767      c
768      c
769      c
770      c
771      c
772      c
773      c
774      c
775      c
776      c
777      c
778      c
779      c
780      c
781      c
782      c
783      c
784      c
785      c
786      c
787      c
788      c
789      c
790      c
791      c
792      c
793      c
794      c
795      c
796      c
797      c
798      c
799      c
800      c
801      c
802      c
803      c
804      c
805      c
806      c
807      c
808      c
809      c
810      c
811      c
812      c
813      c
814      c
815      c
816      c
817      c
818      c
819      c
820      c
821      c
822      c
823      c
824      c
825      c
826      c
827      c
828      c
829      c
830      c
831      c
832      c
833      c
834      c
835      c
836      c
837      c
838      c
839      c
840      c
841      c
842      c
843      c
844      c
845      c
846      c
847      c
848      c
849      c
850      c
851      c
852      c
853      c
854      c
855      c
856      c
857      c
858      c
859      c
860      c
861      c
862      c
863      c
864      c
865      c
866      c
867      c
868      c
869      c
870      c
871      c
872      c
873      c
874      c
875      c
876      c
877      c
878      c
879      c
880      c
881      c
882      c
883      c
884      c
885      c
886      c
887      c
888      c
889      c
890      c
891      c
892      c
893      c
894      c
895      c
896      c
897      c
898      c
899      c
900      c
901      c
902      c
903      c
904      c
905      c
906      c
907      c
908      c
909      c
910      c
911      c
912      c
913      c
914      c
915      c
916      c
917      c
918      c
919      c
920      c
921      c
922      c
923      c
924      c
925      c
926      c
927      c
928      c
929      c
930      c
931      c
932      c
933      c
934      c
935      c
936      c
937      c
938      c
939      c
940      c
941      c
942      c
943      c
944      c
945      c
946      c
947      c
948      c
949      c
950      c
951      c
952      c
953      c
954      c
955      c
956      c
957      c
958      c
959      c
960      c
961      c
962      c
963      c
964      c
965      c
966      c
967      c
968      c
969      c
970      c
971      c
972      c
973      c
974      c
975      c
976      c
977      c
978      c
979      c
980      c
981      c
982      c
983      c
984      c
985      c
986      c
987      c
988      c
989      c
990      c
991      c
992      c
993      c
994      c
995      c
996      c
997      c
998      c
999      c
1000     c

```


AD-A193 152

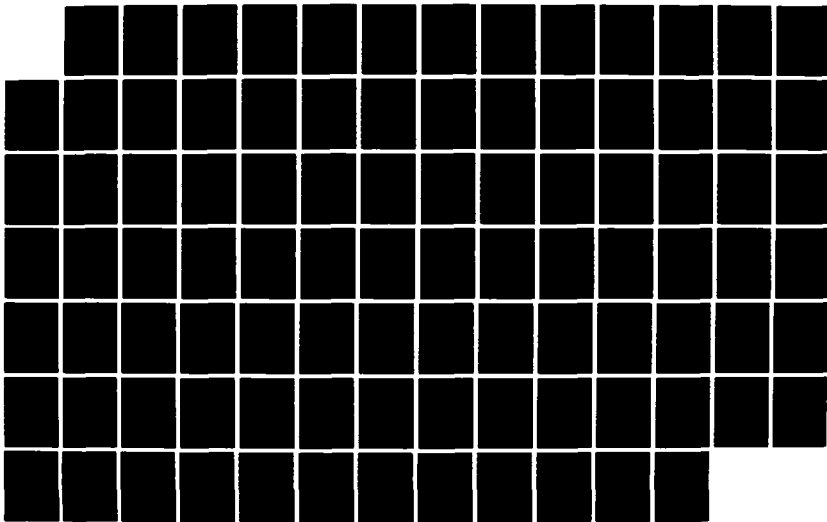
NUMERICAL MODELING OF AIRBLAST(U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA M A FRY JUN 87
SAIC-87/1701 N00014-86-C-2197

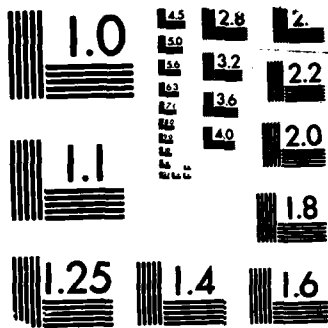
3/3

UNCLASSIFIED

F/G 19/11

NN





MICROCOPY RESOLUTION TEST CHART
JRF-40 U.S. STANDARDS-1963-A

```

622      237.      go to 160
623      238.      continue
624
625      C
626      C
627      C
628      C
629      C
630      C
631      239.      if (display) then
632      240.          if (limp) then
633      241.              call count (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
634      242.                  lv,scr1,kh,kv)
635      243.              if (colour) then
636      244.                  call colcon (scr1,l,l,kh,kv)
637      245.              else
638      246.                  call discon (scr1,ndm1,space,kh,kv)
639      247.              endif
640      248.              else
641      249.                  if (colour) then
642      250.                      call colcon (arr,minh,minv,kh,kv)
643      251.                  else
644      252.                      call discon (arr(minh,minv),ndm1,space,kh,kv)
645      253.                  endif
646      254.              endif
647      255.              else
648      256.                  if (limp) then
649      257.                      call count (arr(minh,minv),hcor(minh),vcor(minv),ndm1,lh,
650      258.                          lv,scr1,kh,kv)
651      259.                      if (colour) then
652      260.                          call colcon (scr1,l,l,kh,kv)
653      261.                          else
654      262.                              call conrec (scr1,ndm1,kh,kv,flc,flh,0,1,0,0)
655      263.                              endif
656      264.                          else
657      265.                              if (colour) then
658      266.                                  call colcon (arr,minh,minv,kh,kv)
659      267.                              else
660      268.                                  call conrec(arr(minh,minv),ndm1,kh,kv,flc,flh,0,-1,0,0)
661      269.                                  endif
662      270.                              endif
663.                              call frame
664.                              endif
665.                              go back and draw the other species of density, if any
666.                              ispec=spec1
667.                              if (ispec.le.nspec.and.ip-7.eq.1) go to 50
668.                              go to 230
669.
670.      C
671.      C
672.      C
673.      C
674.      274.      170      encode (19,350,glab4)
675.      275.      C
676.      276.      label the plot
677.      277.      if (display) then
678.          plot (tmax,tmin)/(vmax-vmin)*4.0, x=spst(glab4,17)/2

```

```

678 call messag (glab4,17,tleng,-.75)
679 else
680 call pwrnt (R00,980,glab4,18,1,0,0)
681 endif
682
683 c
684 c first calculate velocities from momenta.
685 do 180 iv=1,ny,maxv
686 do 180 nh=1,ni,maxh
687   pre(nh,iv)=1.0/rho(nh,iv)
688   arv(nh,iv)=arh(nh,iv)*pre(nh,iv)
689   arv(nh,iv)=arv(nh,iv)*pre(nh,iv)
690
691 c plot vectors. interpolate onto uniform grid if lintp=.true.
692 velmax=0.0
693 velmin=1.e50
694 if (lintp) then
695   call count (arh(minh,minv),hcor(minh),vcor(minv),ndmi,1h,lv
696   ,scr1,ksh,ksv)
697   call count (arv(minh,minv),hcor(minh),vcor(minv),ndmi,1h,lv
698   ,scr2,ksh,ksv)
699   do 200 nv=1,ksv
700     pre(nv,iv)=scr1(nh,iv)+scr2(nh,iv)+scr2(nh,iv)
701     velmax=amax1(velmax,pre1(smax(ksh,pre1(iv),1),iv))
702     velmin=amin1(velmin,pre1(smin(ksh,pre1(iv),1),iv))
703   continue
704   velmax=sp1(velmax)
705   velmin=sp1(velmin)
706   velmin=velmin*0.05*(velmax-velmin)
707 c draw the velocity vectors. Only NCAR is made use of (scaled to fit
708 c the DISPLA axes, if necessary), as only NCAR has a velocity plotting
709 c routine. libcar is called to initialize NCAR graphics, set to scale
710 c the plot, if displa=.true.
711 if (displa) then
712   call libcar
713   xa=(1.0-.8*(hmax-hmin))/(vmax-vmin)/2.+0.025
714   xb=1.05 xa
715   call set (xa,xb,110,838,hmin,hmax,vmin,vmax,1)
716   call velct (scr1,ndmi,scr2,ndmi,ksh,ksv,velmin,velmax,-1,
717   0,0,0,0)
718 else
719   call velct (scr1,ndmi,scr2,ndmi,ksh,ksv,velmin,velmax,-1,
720   0,0,0,0)
721 endif
722
723 c
724 do 220 iv=1,ny,minvkv 1
725 do 210 nh=1,ni,minhkv 1
726   pre(nh,iv)=arh(nh,iv)*arv(nh,iv)+arv(nh,iv)
727   velmax=amax1(velmax,pre1(smax(kh,pre1(nh,iv),1)*minh-1,iv))
728   velmin=amin1(velmin,pre1(smin(kh,pre1(nh,iv),1)*minh-1,iv))
729 continue
730 velmax=sp1(velmax)
731 velmin=sp1(velmin)
732 velmin=velmin*0.05*(velmax-velmin)
733 c plot the vector plot (as described above), using uninterpolated grid.
734 if (displa) then

```

```

734      call jbwcar
735      xa(1,0) = .8*(hmax-hmin)/(vmax-vmin)/2. + .025
736      xb=1.05 xa
737      call set (xa,xb,110.,B38,hmin,hmax,vmin,vmax,1)
738      call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
739      ,velmin,velmax,1,0,0,0)
740      else
741      call velvct (arh(minh,minv),ndm1,arv(minh,minv),ndm1,kh,kv
742      ,velmin,velmax,-1,0,0,0)
743      endif
744      call framem
745      if (displa) call endpl (0)
746      go to 230
747      return
748      332.
749
750      c
751      c
752      c
753      c
754      c
755      c
756      c
757      c
758      c
759      c
760      c
761      c
762      c
763      c
764      c
765      c
766      c
767      c
768      c
769      c
770      c
771      c
772      c
773      c
774      c
775      c
776      c
777      c
778      c
779      c
780      c
781      c
782      c
783      c
784      c
785      c
786      c
787      c
788      c
789      c

```

using either NCAR or DJSSPIA, the plot is labeled, and markers are placed at points showing the positions of the trace particles. A different marker is used for each burst, from the arrays lchar or imark, respectively.

if (displa) then
 tlenq = (lmax-hmin)/(vmax-vmin)*4.0 - xmess(qlab4,10)/2.
 call messag (qlab4,10,tlenq,0.75)
 do 405 k=1,nbu
 call marker (imark(k))
 do 405 ljk = 1, nopp(k)
 xp(ljk,k) = 0.00001 * xp(ljk,k)
 zp(ljk,k) = 0.00001 * zp(ljk,k)
 continue
 call curve (xp(1,k),zp(1,k),nopp(k),-1)
 continue
 call endpl(0)
 else
 call writ (800,980,qlab4,11,1,0,0)
 write (6,1021)
 do 410 k=1,nbu
 do 410 l=1,nopp(k)
 call points(xp(l,k),zp(l,k),lchar(k),0)
 write (6,1022) l,xp(l,k),zp(l,k)
 continue
 call framem
 endif
 go to 230

grid
 this draws the irregular grid on which the original calculation was done. this serves to show where the resolution was finest, and where the most interpolation was done, on the contour plots.

```

790 360. h(i,1) = hcor(minh + 1, i - 1)
791 361. h(i,2) = h(i,1)
792 362. 421 continue
793 363. C do 422 j = 1, nj
794 364. j = j + 1
795 365. v(j,j,1) = vcor(minv + j, j - 1)
796 366. v(j,j,2) = v(j,j,1)
797 367. 422 continue
798 368. C do 423 i = 1, ni
799 369. v(i,1) = v(minf,1)
800 370. v(i,2) = v(nf,1)
801 371. 423 continue
802 372. C do 424 j = n(i,1), nj
803 373. h(j,1) = h(i,1)
804 374. h(j,2) = h(nf,1)
805 375. 424 continue
806 376. C if (nj .lt. nchmlp2) then
807 377. do 425 j = nj + 1, nchmlp2
808 378. v(j,1) = v(nf,1)
809 379. v(j,2) = v(nf,2)
810 380. h(j,1) = h(nf,1)
811 381. h(j,2) = h(nf,2)
812 382. 425 continue
813 383. endif
814 384. C draw the grid itself
815 385. if (displa) then
816 386. do 426 j = 1, nchmlp2
817 387. space(1,1) = h(j,1)/100000.
818 388. space(2,1) = h(j,2)/100000.
819 389. space(1,2) = v(j,1)/100000.
820 390. space(2,2) = v(j,2)/100000.
821 391. call curve (space(1,1),space(1,2),2,0)
822 392. continue
823 393. call cmpl (0)
824 394. else
825 395. call ablat (th, th1, 1, 1, 0, 0)
826 396. call aspl ( win , , 1 )
827 397. call aspl ( grid/left , , hlo )
828 398. call aspl ( grid/right , , hhi )
829 399. call aspl ( grid/bottom , , vlo )
830 400. call aspl ( grid/top , , vhi )
831 401. call aspl ( x/minimum , , h(1,1) )
832 402. call aspl ( x/maximum , , h(nf,1) )
833 403. call aspl ( y/minimum , , v(nf,1) )
834 404. call aspl ( y/maximum , , v(nf,1) )
835 405. call aspl ( x/hice , , 0 )
836 406. call aspl ( y/hice , , 0 )
837 407. call aspl ( row , , 2 )
838 408. call rzmxy ( h, v, nchmlp2, nchmlp2, 2, 0 )
839 409. endif
840 410. C

```

```

846      409.  210      continue
847      410.          return
848
849      C
850      C   this entry is a fossilized routine.  Implemented but not used in the
851      C   current version of the code.  Ignore it, or better still, rip it out.
852      C   if you have the time and inclination.
853      C   entry gr(intlpt)
854
855      C -----
856      C   reads in data for in-line graphics.
857      C -----
858      411.          read (5,qfdat)
859      412.          write (6,qfdat)
860      413.          if (mslice.eq.1) lplane=kmmin
861      414.          if (mslice.eq.2) lplane=lmn
862      415.          if (mslice.eq.3) lplane=lmn
863      416.          lpl=lplane
864      417.          do 240 nv=1,ndm2
865      418.             rho(nv,nv)=cvmgt(rcamb(lpl+1),ecamb(nv+1),mslice.eq.1)
866      419.             vx(nv,nv)=0.0
867      420.             vy(nv,nv)=0.0
868      421.             vz(nv,nv)=0.0
869      422.             eq(nv,nv)=cvmgt(ecamb(lpl+1),ecamb(nv+1),mslice.eq.1)
870      423.             call gplot (lhw,4lplot,12)
871      424.             if (displa) then
872      425.                call libdisp
873      426.             else
874      427.                call libcar
875      428.             endif
876      429.             return
877      430.
878      431.          entry gpreid
879
880      C -----
881      C   terminates NCAR or DISSPLA for in-line graphics.
882      C -----
883      432.          call gpreu
884      433.          return
885
886      C
887      C   formats, and other important goodies, for the titles and labels.
888      C -----
889      434.          format ('time=',f5.1,' sec. step',15,' dump ',a)
890      435.          format ('X Y plane ',13,' at z=',f5.1,' km$')
891      436.          format ('X Y plane ',13,' at z=',e9.3,' km$')
892      437.          format ('Y Z plane ',13,' at x=',f5.1,' km$')
893      438.          format ('Y Z plane ',13,' at x=',e9.3,' km$')
894      439.          format ('X Z plane ',13,' at y=',f5.1,' km$')
895      440.          format ('X Z plane ',13,' at y=',e9.3,' km$')
896      441.          format ('density (g/cm**3)$')
897      442.          format ('density ',11,' (g/cm**3)$')
898      443.          format ('energy (ergs/cm**3)$')
899      444.          format ('pressure (dynes/cm**2)$')
900      445.          format ('mach number$')
901      446.          format ('vorticity (cm/sec)$')
902      447.          format ('particle$')
903      448.          format ('part ',14,'$')
904      449.          format ('part ',14,'$')
905      450.          format ('part ',14,'$')
906      451.          format ('part ',14,'$')

```

902 CONV3D 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. DEPENDENCY INVOLVING ARRAY "XP" IN SEQUENCE NUMBER 353
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "XP" IN SEQUENCE NUMBER 353
 EXPLANATION: ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION
 O AT SEQUENCE NUMBER 352. DEPENDENCY INVOLVING ARRAY "ZP" IN SEQUENCE NUMBER 353
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "ZP" IN SEQUENCE NUMBER 353
 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. DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 O AT SEQUENCE NUMBER 369. DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 370
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D VECTOR LOOP BEGINS AT SEQ. NO. 368. P= 4266a
 CONV3D VECTOR LOOP BEGINS AT SEQ. NO. 368. P= 4266a

O AT SEQUENCE NUMBER 373. DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 O AT SEQUENCE NUMBER 373. DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 374
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 O AT SEQUENCE NUMBER 378. DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 379
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 379
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 O AT SEQUENCE NUMBER 378. DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 379
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "V" IN SEQUENCE NUMBER 379
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS

O AT SEQUENCE NUMBER 379. DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 380
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 380
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 O AT SEQUENCE NUMBER 379. DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 380
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "H" IN SEQUENCE NUMBER 380
 EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
 CONV3D VECTOR LOOP BEGINS AT SEQ. NO. 377. P= 4340b
 CONV3D VECTOR LOOP BEGINS AT SEQ. NO. 377. P= 4340b

O AT SEQUENCE NUMBER 390. DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 386
 PRNAME CONV3D COMMENT - DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 386
 EXPLANATION: ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION

0 AT SEQUENCE NUMBER - 390. DEPENDENCY INVOLVING ARRAY "SPACE" IN SEQUENCE NUMBER 387
 PRNAME CONV3D COMMENT -
 EXPLANATION: ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION P=0250305C

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

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

NUMBER USE SOURCE PROGRAM REFERENCES

10	1727C	61L	60J
20	1778C	76I	60J
30	2043C	91I	60J
40	2111A	105I	90J
45	2162D	128I	123J
50	2177B	272J	75J
60	2360B	163I	162J
70	2366B	164L	163J
72	UNDEF**	169L	167E
72A	2374A	165L	165E
72R	2431D	165L	
72C	2415B	167L	
72D	2427A	167L	
80	2467D	178I	163J
90	2533R	189L	163J
100	2574B	198L	162J
110	2634A	207I	162J
120	UNDEF**	217L	216E
120A	2707d	216L	
120H	2731A	216I	
130	UNDEF**	219L	215E
130A	2677B	215L	
130R	2771D	215I	
140	2772B	221I	162J
145	UNDEF**	231I	230E
145A	3046A	230I	
145B	3067B	230I	229F
146	UNDEF**	233I	
146A	3035C	229I	
146R	3130A	229L	
150	UNDEF**	236I	234F
150A	3134A	234L	
150H	3174A	234L	
150X	3144C	235I	
150D	3171D	235I	
160	3174C	238I	220J
170	3361C	274I	206J
180	UNDEF**	285I	197J
180A	3423A	281I	281E
180B	3455B	281I	
180C	3436C	282L	
180H	3452C	282L	
190	UNDEF**	293L	292E
190A	3535C	292I	
190R	3546D	292I	
200	UNDEF**	296L	291E
200A	3526B	291I	
200R	3572B	291I	
210	UNDEF**	312L	311E
210A	3675C	311L	
			177J
			188J
			287E
			297E
			311E

210R3705d	3111				
220 UNDEF**	3151	310F			
220A3661b	3101				
220R3736C	3101				
230 4533b	4091	357J	331J	273J	131J 130E
230A2175J	1301				
230RUNDEF**	1301				
240 UNDEF**	424L	419E	418E		
240A4575b	418L				
240RUNDEF**	418L				
240C4611b	4191				
240RUNDEF**	4191				
250 FN	4351	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	444L	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**	3451	338E			
405A4076b	338L				
405R4136b	338L				
406 UNDEF**	3431	340F			
406A4113b	3401				
406B4174b	341L				
410 UNDEF**	3541	351E			
410A4160a	3501				
410R4216C	350L				
410C4165b	351L				
41004213d	351L				
420 4220a	358L	162J			
421 UNDEF**	362L	359E			
421A4232a	359L				
421B4241d	359L				
422 UNDEF**	3671	363E			
422A4253d	3631				
422R4266a	3631				
423 UNDEF**	371L	368E			
423A4302d	368L				
423B4324a	368L				
424 UNDEF**	375L	372E			
424A4332d	372L				
424B4340b	372L				
425 UNDEF**	3821	377F			
425A4355C	377L				
425R4405a	377L				
426 UNDEF**	391L	385F			

426A4411C	385I
426RUNDEF..	385L
1021 FN	450I
1022 FN	451I
00002 1726a	60W
00003 1762d	70I
00004 1775a	72I
00005 2030d	85I
00006 2043a	87I
00007 2076d	100I
00008 2111a	102I
00009 2162b	117I
00010 2302b	132I
00011 2206d	133I
00012 2350a	153L
00013 2354d	162W
00014 2364d	163W
00015 2457d	171I
00016 2467b	174I
00017 2521a	180I
00018 2530c	183I
00019 2564b	191I
00020 2573d	194I
00021 2624a	200I
00022 2633c	203I
00023 2663d	209I
00024 2673b	212I
00025 3022a	223I
00026 3031c	226I
00027 3260c	239I
00028 3236b	240L
00029 3227d	242I
00030 3235d	244L
00031 3260a	247I
00032 3245d	248I
00033 3260a	250I
00034 3354a	254I
00035 3325b	255I
00036 3312d	257L
00037 3324d	259I
00038 3353a	262I
00039 3334d	263I
00040 3353a	265I
00041 3407c	275L
00042 3417a	278L
00043 3654c	288I
00044 3640c	300I
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

349W
353W

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 382L
 00065 4405a 382E
 00066 UNDEF** 382E
 00067 4376a 382E

(SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (LN=FORMAT NUMBER, UNDEF**=UNDEFINED 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		
\$EFF	EXTERNAL			334U	274U	179U 190U 170U 103U
\$EFL	EXTERNAL			101U	88U 73U	59U 59U 170U 103U
\$FFV	EXTERNAL			334U	274U 208U	199U 71U 190U 170U 103U
\$RNL	EXTERNAL			101U	88U 73U	59U 59U 73U/2 71U/2 59U/2
\$WFF	EXTERNAL			190U	103U/2 101U/2	88U/2
\$WFI	EXTERNAL			412U		
\$WV	EXTERNAL			353U	349U	
\$WNL	EXTERNAL			353U/3		
ASSET1	EXTERNAL			406U	404U 403U	401U 400U 399U 398U
AMAX1	EXTERNAL			397U		
AMINI	EXTERNAL			395U		
ANDJAT	EXTERNAL			313U		
ARFA2D	EXTERNAL			314U		
4 ARR	R	2DIM ARRAY	POINTEE	139U		
1 ARR	R	2DIM ARRAY	POINTEE	326P	284P	284U 284S 30P 28D
5 ARR	R	2DIM ARRAY	POINTEE	266P	264P 251P	249P 26D 26D
COLCON	I	VARIABLE	GRFCOM	326P	312U/2 289P	285U 285S 30P 28D
CONJNT	EXTERNAL			264U	249U 243U	
CONREC	EXTERNAL			263U	257U 242U	
1645 CONV3D	ENTRY			290U	289U 241U	
CURVE	EXTERNAL			432D	10/2	
CVMGT	EXTERNAL			390U	169U	
34 DARKIN	L	VARIABLE	GRFCOM	424U	166U	
1560 DELH	R	VARIABLE		133U	52D	
1561 DELV	R	VARIABLE		126P	118S	
1562 DIV	R	VARIABLE		125U	119S	
DISCON	EXTERNAL			126U	122P 125S	
				251U	122U 121S	

CONV3D	PAGE 18	OF ACDELPOK5.LV-7	08/18/86	MX-V	12-07:27	C11140 (05/16/86)	PAGE 30						
31	DISPIA	I	VARIABLE	GRFCOM	4260	3840	3350	3300	3190	3000	2750	2390	2230
					2090	2000	1910	1900	1710	1320	1000	850	700
30	DMAX	R	VARIABLE	GRFCOM	560	520							
27	DMIN	R	VARIABLE	GRFCOM	1870	560							
264	FCAMB	R	2D EQ. ARRAY	INDFIX	1860	560							
101024	FINI	R	2D EQ. ARRAY	LOCOM1	424P/2	2315	218P	2175	360	340			
	ENDPL	R	EXTERNAL		3920	3460	3300						
	EOS	R	EXTERNAL		2320	2180							
45100	FRG	R	2DIM ARRAY	PLTVAR	4245	2310	2170	198P	90	70			
	E7MXV	R	EXTERNAL		4070								
1574	FHI	R	VARIABLE		266P	260P	1875	1605	1525				
1573	FLO	R	VARIABLE		266P	260P	1865	1595	1515				
	FLOAT	R	INLINE FUNCT.		1250	1210							
	FRAME	R	EXTERNAL		1470								
	FRAMFN	R	EXTERNAL		3550								
32010	GAM	R	2D EQ. ARRAY	LOCOM1	2360	232P	218P	350	330				
	GDONE	R	EXTERNAL		4330								
1600	GLAB	R	VARIABLE		1905								
1513	GLAB1	R	2D EQ. ARRAY		154P	141P	585	510					
1522	GLAB2	R	2D EQ. ARRAY		155P	142P	595	510					
1531	GLAB3	R	2D EQ. ARRAY		156P	143P	1035	1015	885	865	735	715	510
1540	GLAB4	R	2D EQ. ARRAY		348P	337P	336P	3345	279P	277P	276P	2745	227P
		R	2D EQ. ARRAY		225P	224P	225P	213P	211P	210P	2085	204P	202P
		R	2D EQ. ARRAY		201P	199S	195P	193P	192P	184P	182P	181P	179S
		R	EXTERNAL		175P	173P	172P	1705	1705				
		R	EXTERNAL		4250								
		R	EXTERNAL		1500								
1612	GRFAT	R	NAMELIST		413P	412P	555						
4644	GRFEND	R	ENTRY		4320/2								
4542	GRFIN	R	ENTRY		4110/2								
47234	GSCR	R	2D EQ. ARRAY	LOCOM1	232P	218P	330	370	3860	3810	3815	3800	3805
276	H	R	2D EQ. ARRAY		407P	401P	400P	3870	3610	3615	3605	270	260
		R	2D EQ. ARRAY		3740	3745	3735	3730	3615	3615	3605	290	280
		R	2D EQ. ARRAY		3600	290P	289P	256P	241P	1070	1060	290	280
		R	2D EQ. ARRAY		1430	1420	1410						
2	HOUR	R	2D EQ. ARRAY	POINTEE	1400								
	HEIGHT	R	EXTERNAL		157P	157P	128P	128P					
1564	IRH	R	VARIABLE		397P	157P	128P	128P					
1563	HLO	R	VARIABLE		396P	157P	128P	128P					
1	HMAX	R	VARIABLE	ROUND	3360	323P	3210	304P	3020	2760	2740	2100	2010
		R	VARIABLE		1920	1810	1720	157P	150P	1390	128P	1180	1075
		R	VARIABLE	ROUND	570								
		R	VARIABLE	ROUND	3360	323P	3210	304P	3020	2760	2740	2100	2010
		R	VARIABLE		1920	1810	1720	157P	150P	1390	128P	1180	1065
		R	VARIABLE		570								
1607	I	R	VARIABLE		3700	3690	3680	3610/2	3600/2	3590	3530/3	3520/2	3510
1507	ICMAR	R	2D EQ. ARRAY		352P	545	500						
1606	IJK	R	VARIABLE		3420/2	3410/2	3400						
32	IL	R	VARIABLE	GRFCOM	3600	1120	560						
1511	IMARK	R	2D EQ. ARRAY		339P	545	500						
2	IMAX	R	VARIABLE	GRFCOM	93P	63P	560	555					
1	IMIN	R	VARIABLE	GRFCOM	4150	62P	62P	560	555				
	INT	R	INLINE FUNCT.		1260	1220							
1570	IP	R	VARIABLE		2720	162P	1300	1300					
1501	IPARR	R	VARIABLE		985	835	300P	300P					
1476	IPARR	R	VARIABLE		2715	2075	1985	1805	1785	1645	290P		

1502	IPARV	+	VARIABLE	995	845	695	30P					
1477	IPK OR	+	VARIABLE	965	R15	665	29P					
1504	IPMAX 1	+	VARIABLE	935	785	635	31P					
1506	IPMAXV	+	VARIABLE	955	805	655	32P					
1503	IPMIN1	+	VARIABLE	925	775	625	31P					
1505	IPMINV	+	VARIABLE	945	795	645	32P					
1500	IPVCR	+	VARIABLE	975	825	675	29P					
	ISMAX	I	EXTERNAL	3130	2940							
	ISMIN	I	EXTERNAL	3140	2950							
1567	ISPEC	I	VARIABLE	2720	2710	2715	190U	179U	163P	1295	379U	378U
1	ISTEP	I	VARIABLE	59U	ID							
1610	J	I	VARIABLE	389U	387U	387U	386U	3851	381U	380U	379U	
		I	VARIABLE	3771	374U	373U	3721	365U	364U	3631		
1611	JJ	I	VARIABLE	366U/2	365U	3645						
33	JL	I	VARIABLE	365U	113U	56D						
4	JMAX	I	VARIABLE	78P	65P	56D	555					
3	JMIN	I	VARIABLE	416U	77P	64P	56D					
1605	K	I	VARIABLE	353U/2	352U/3	351N	3501	344U/3	342U/2	341U/2	340N	339U
		I	VARIABLE	3381	324P	314P	313P	311N	266P	264P	260P	258P
1556	KH	I	VARIABLE	256P	251P	249P	245P	243P	241P	2365	121P	120S
		I	VARIABLE	1155	80P	56D	545					
6	KMAX	I	VARIABLE	95P	94P	79P	55S					
5	KMIN	I	VARIABLE	414U	305P	295P	284P	292N	290P	289P	535	
1547	KSH	I	VARIABLE	307P	307P	291N	290P	289P	535			
1550	KSV	I	VARIABLE	307P	307P	291N	290P	289P	535			
1557	KV	I	VARIABLE	326P	324P	310N	256P	264P	256P	251P	251P	
		I	VARIABLE	249P	243P	243P	241P	125P	1245	123U	1225	1165
3	LABEL	CH	VARIABLE	58U	48D	11	241P	115U	112U	1105		
1551	LH	I	VARIABLE	290P	289P	256P	241P					
		I	VARIABLE	427U	320U	301U	117U	56D	55S	52D		
	LIBRISP	I	EXTERNAL	429U	255U	240U	178U	164U	164U	99U	97U	
	LIBNCAR	L	EXTERNAL	288U	207U	198U	189U	184U	84U	83U	81U	
25	LINTP	L	VARIABLE	221U	96U	94U	93U	92U	84U	83U	81U	
	LOC	I	INLINE FUNCT	ROU	79U	78U	77U	69U	68U	67U	65U	
		I	VARIABLE	64U	63U	62U						
6	LPI	I	VARIABLE	424U	420U	4175	411D					
5	LPLANE	I	VARIABLE	417U	4165	4155	4145	166U	103U/2	101U/2	88U/2	86U/2
		I	VARIABLE	73U/2	71U/2	11						
1552	L V	I	VARIABLE	290P	289P	256P	241P	116U	113U	1115		
15004	MAC	R	2D EQ APPRAY	2365	221P	35D	33D					
	MAIN.	I	ENTRY	339U	282N	31P						
7	MAX1	I	VARIABLE	282N	235N	232P	230N	218P	216N	167N	110U	107U
		I	EXTERNAL	31P	234N	229N	215N	165N	111U	109U	32P	
11	MAXV	I	VARIABLE	281N	277U	225U	211U	202U	193U	182U	173U	
	MESSA'S	I	EXTERNAL	360U	326U/2	324U/2	314U/2	313U/2	311N/2	290U/2	289U/2	282N
6	MIN1	I	VARIABLE	266U	256U/2	256U/2	251U	249P	241U/2	235N	232U/2	230N
		I	VARIABLE	218U/2	216N	167N	106U	106U	31P			
10	MINV	I	VARIABLE	365U	326U/2	324U/2	310U/2	290U/2	289U/2	281N	266U	264P
		I	VARIABLE	256U/2	251U	249P	241U/2	234N	229N	211N	165N	111U
		I	VARIABLE	108U	32P							
7	MPIOT	I	IDIM APPRAY	131U	56D	555	47D					
0	M5 LIC	I	VARIABLE	424P	420P	416U	415U	166P	60P	56D	555	

CONV3D	PAGE 20	ON ACDEFI PQRSUVXZ	OR/IR/RG	MX-V	12-07-27	C11145 (05/16/86)	PAGE 32
O	NRU	I	VARIABLE				
	NRURST	I	PARAMETER				
	NRM1	I	PARAMETER				
				350N	46D		
				50P/2	44P/2		
				419N	324P/2	415	
				256P	241P	305P/2	290P
				28P/2	24P/2	8P/3	38P/3
				23P/2	215	7P	6P/3
				407P/2	385N	376U	225
				418N	124U	34P/2	28P/2
				7P	6P/3	55	
1576	NI	I	VARIABLE				
				424U	423U	421U	312U/5
				292I	284U/3	282I	236U/3
				217U/9	216I	169U/4	167I
1553	NI	I	VARIABLE				
				402U	401U	372N	364U
				112S			359N
1555	NIJ	I	VARIABLE				
				403U	381U	379U	377N
				114S			372N
1554	NJ	I	VARIABLE				
4	NIABFL	I	VARIABLE				
	NNX	I	VARIABLE				
	NNX2	I	PARAMETER				
	NNXP	I	PARAMETER				
	NRIV	I	PARAMETER				
	NNY2	I	PARAMETER				
	NNYV	I	PARAMETER				
	NNZ	I	PARAMETER				
	NNZ2	I	PARAMETER				
	NNZP	I	PARAMETER				
	NNZ2D	I	PARAMETER				
	NRIRDR	I	EXTERNAL				
	NOP	I	PARAMETER				
430	NOPP	I	IDIM ARRAY				
	NPARTL	I	PARAMETER				
	NSPEC	I	PARAMETER				
	NSUM	I	PARAMETER				
1575	NV	I	VARIABLE				
				43P/2	415	45D	44D
				344P	340N		
				232P	218P	10S	
				10S			
				424U/2	423U	421U	420U/2
				310I	295U/2	294U/2	291I
				236U/3	234I	232U/5	231U/R
				168U/3	166U/2	165I	229I
				23P	20S		
				137U			
				158U			
				352U			
				314P/2	312S	295P/2	294P/2
				236U	232P	207P	35D
				232P	218P	37U	33D
				348U	279U	213U	204U
				155U	154U		
				420P/2	166P/2	11D	
				59U	49D		
				232P	218P	9D	8D
				232P	218P	9D	8D
				470S	283U	212P	231U
				9D	8D		
				232P	218P	166S	39D
				421S	236U/2	231U/2	98P
				422S	236U/2	231U/2	83P
47124	P5CR	R	ID EQ. ARRAY				
	PWRIT	R	EXTERNAL				
				23P			
				137U			
				158U			
				352U			
				314P/2	312S	295P/2	294P/2
				236U	232P	207P	35D
				232P	218P	37U	33D
				348U	279U	213U	204U
				155U	154U		
				420P/2	166P/2	11D	
				59U	49D		
				232P	218P	9D	8D
				232P	218P	9D	8D
				470S	283U	212P	231U
				9D	8D		
				232P	218P	166S	39D
				421S	236U/2	231U/2	98P
				422S	236U/2	231U/2	83P
47014	RSCR	R	ID EQ. ARRAY				
	RVX	R	EXTERNAL				
	RVY	R	EXTERNAL				
				23P			
				137U			
				158U			
				352U			
				314P/2	312S	295P/2	294P/2
				236U	232P	207P	35D
				232P	218P	37U	33D
				348U	279U	213U	204U
				155U	154U		
				420P/2	166P/2	11D	
				59U	49D		
				232P	218P	9D	8D
				232P	218P	9D	8D
				470S	283U	212P	231U
				9D	8D		
				232P	218P	166S	39D
				421S	236U/2	231U/2	98P
				422S	236U/2	231U/2	83P
47014	RSCR	R	ID EQ. ARRAY				
	RVX	R	EXTERNAL				
	RVY	R	EXTERNAL				
				23P			
				137U			
				158U			
				352U			
				314P/2	312S	295P/2	294P/2
				236U	232P	207P	35D
				232P	218P	37U	33D
				348U	279U	213U	204U
				155U	154U		
				420P/2	166P/2	11D	
				59U	49D		
				232P	218P	9D	8D
				232P	218P	9D	8D
				470S	283U	212P	231U
				9D	8D		
				232P	218P	166S	39D
				421S	236U/2	231U/2	98P
				422S	236U/2	231U/2	83P

30600 RVZ	R	2D DIM ARRAY	PLIVAR	4235	236U/2	211U/2	217U/2	99P	R4P	'00	6D
47014 SCR1	R	2D EQ. ARRAY	LOCUM1	307P	305P	297U/2	289P	260P	258P	295P	245P
64020 SCR2	R	2D EQ. ARRAY	LOCUM1	241P	40N/2	39U/3	36D	34D	36D	34D	243P
47344 SCR3	R	1D EQ. ARRAY	LOCUM1	307P	218P	40P	38P	400			
47454 SCR3	R	1D EQ. ARRAY	LOCUM1	232P	218P	40P	38D				
64070 SCR3	R	1D EQ. ARRAY	LOCUM1	232P	218P	40P	38D				
SFT1	R	EXTERNAL		323U	304U	157U					
SHDCIR	R	EXTERNAL		135U							
O SPACE	R	2D DIM ARRAY	HOLDER	390P/2	389S	387S	386S	386S	251P	243P	24D
14302 SPAC2	R	2D DIM ARRAY	HOLDER	169P/2	168S	164P	24D				
SMJ55M	R	EXTERNAL		317U	298U	297U	236U				
TIMEFRM	R	EXTERNAL		134U							
3 TIME	R	VARIABLE	GLOVAR	59U	19D						
1577 TLENG	R	VARIABLE		337P	336S	277P	276S	225P	224S	211P	210S
776 V	R	2D DIM ARRAY		201S	193P	192S	182P	173P	173P	172S	202P
3 VCOR	R	1D DIM ARRAY	POINTEE	407P	403P	389U	389U	388U	379U	379S	378S
1601 VELMAX	R	VARIABLE		370U	370S	369U	369S	366U	366S	365S	27D
1602 VELMIN	R	VARIABLE		365U	290P	289P	241P	241P	109U	109U	26D
VELVCT	R	EXTERNAL		326P	324P	318U	316P	316S	313P	313S	305P
1566 VIE	R	VARIABLE		326P	324P	299U	294P	294S	286S	314P	307P
1565 VLO	R	VARIABLE		305P	289U/2	299S	298P	298S	317S	295S	287S
3 VMAX	R	VARIABLE	ROUND	398P	324U	307U	305U				
2 VMIN	R	VARIABLE	ROUND	398P	157P	128P					
W5IZE	R	EXTERNAL		398P	157P	128P					
1603 XA	R	VARIABLE		398P	157P	128P					
1604 XB	R	VARIABLE		336U	323P	321U	304P	302U	276U	224U	210U
O XCOR	R	1D DIM ARRAY	GRDVAR	192U	181U	172U	157P	150P	148U	139U	128P
XMF55	R	EXTERNAL		109S	57D						
XNAME	R	EXTERNAL		336U	323P	321U	304P	302U	276U	224U	210U
O XP	R	2D DIM ARRAY	PARTL	108S	57D						
1572 XSTEP	R	VARIABLE		128U							
111 YCOR	R	1D DIM ARRAY	GRDVAR	323P	322S	304P	303S	303U	302S		
YNAME	R	EXTERNAL		96P	88U	86U	66P	18D	15D		
1571 YSTEP	R	VARIABLE		336U	276U	224U	210U	201U	192U	181U	172U
152 ZCOR	R	1D DIM ARRAY	GRDVAR	144U	352P	344P	341U	341S	45D	43D	
214 ZP	R	2D DIM ARRAY	PARTL	353U	149S	81P	67P	18D	15D		

TABLE OF EXTERNAL NAMES

\$EFA	R	EXTERNAL		59U	58U	222U	208U	199U	190U	179U	170U	103U
\$FFF	R	EXTERNAL		334U	274U	86U	73U	71U	59U	58U	170U	103U
\$EFT	R	EXTERNAL		101U	88U	274U	208U	199U	190U	179U	170U	103U
\$FTV	R	EXTERNAL		101U	88U	274U	208U	199U	190U	179U	170U	103U
\$FTV	R	EXTERNAL		190U	179U	103U/2	101U/2	88U/2	86U/2	73U/2	71U/2	59U/2

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

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

TABLE OF PARAMETERS ENCOUNTERED

NRURST	=	2
NDM12	=	6660
NDM2	=	RR
NNX2	=	74
NNXV	=	33
NNXP	=	34
NNZ2	=	90
NNZ2D	=	3150
NPARTL	=	422
NSUM	=	5

NDM1	=	72
NDM1P2	=	160
NNX	=	72
NNXP	=	73
NNY2	=	35
NNZ	=	RR
NNZP	=	89
NOP	=	70
NSPEC	=	1
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 NI	167	169	2415b	12
130 NV	215	219	2677b	73
120 NI	216	217	2707d	22
146 NV	229	233	3035c	73
145 NI	230	231	3046a	21
150 NV	234	236	3174a	40
150 NI	235	236	3144c	25
180 NV	281	285	3423a	32
180 NI	282	285	3436c	14
200 NV	291	296	3526b	44
190 NI	292	293	3535c	11
220 NV	310	315	3668b	56
210 NI	311	312	3675c	10
405 K	338	345	4076b	40
405 L,K	340	343	4117b	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	4372d	6
425 J	377	382	4355c	14

426 J 391 411C 16
 240 NV 418 424 4575D 34
 240 NI 419 424 4611D 15

BLOCK NAMES AND LENGTHS IN OCTAL

4655 CONV3D 65 #1B 1243 #CL 1116 GREDVAR
 14-GLDVAR 30602-HOLDER 11601D LOCOM1 35-GRF.COM
 4-HOUR 646 PART1 1741-INDFIX 1-NBRJR

STATIC SPACE (IN OCTAL)

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

```

903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946

1      Subroutine disprf (arrx,arry,length)
2      real arrx(length), arry(length)
3
4      * This Subroutine plots profile graphs from the input data.
5      * It is called to do profiles on part of an array, using
6      * DISPLA calls to the routine curve.
7
8      C
9      C
10     C
11     C
12     C
13     C
14     C
15     C
16     C
17     C
18     C
19     C
20     C
21     C
22     C
23     C
24     C
25     C
26     C
27     C
28     C
29     C
30     C
31     C
32     C
33     C
34     C
35     C
36     C
37     C
38     C
39     C
40     C
41     C
42     C
43     C
44     C
45     C
46     C
47     C
48     C
49     C
50     C
51     C
52     C
53     C
54     C
55     C
56     C
57     C
58     C
59     C
60     C
61     C
62     C
63     C
64     C
65     C
66     C
67     C
68     C
69     C
70     C
71     C
72     C
73     C
74     C
75     C
76     C
77     C
78     C
79     C
80     C
81     C
82     C
83     C
84     C
85     C
86     C
87     C
88     C
89     C
90     C
91     C
92     C
93     C
94     C
95     C
96     C
97     C
98     C
99     C
100    C
101    C
102    C
103    C
104    C
105    C
106    C
107    C
108    C
109    C
110    C
111    C
112    C
113    C
114    C
115    C
116    C
117    C
118    C
119    C
120    C
121    C
122    C
123    C
124    C
125    C
126    C
127    C
128    C
129    C
130    C
131    C
132    C
133    C
134    C
135    C
136    C
137    C
138    C
139    C
140    C
141    C
142    C
143    C
144    C
145    C
146    C
147    C
148    C
149    C
150    C
151    C
152    C
153    C
154    C
155    C
156    C
157    C
158    C
159    C
160    C
161    C
162    C
163    C
164    C
165    C
166    C
167    C
168    C
169    C
170    C
171    C
172    C
173    C
174    C
175    C
176    C
177    C
178    C
179    C
180    C
181    C
182    C
183    C
184    C
185    C
186    C
187    C
188    C
189    C
190    C
191    C
192    C
193    C
194    C
195    C
196    C
197    C
198    C
199    C
200    C
201    C
202    C
203    C
204    C
205    C
206    C
207    C
208    C
209    C
210    C
211    C
212    C
213    C
214    C
215    C
216    C
217    C
218    C
219    C
220    C
221    C
222    C
223    C
224    C
225    C
226    C
227    C
228    C
229    C
230    C
231    C
232    C
233    C
234    C
235    C
236    C
237    C
238    C
239    C
240    C
241    C
242    C
243    C
244    C
245    C
246    C
247    C
248    C
249    C
250    C
251    C
252    C
253    C
254    C
255    C
256    C
257    C
258    C
259    C
260    C
261    C
262    C
263    C
264    C
265    C
266    C
267    C
268    C
269    C
270    C
271    C
272    C
273    C
274    C
275    C
276    C
277    C
278    C
279    C
280    C
281    C
282    C
283    C
284    C
285    C
286    C
287    C
288    C
289    C
290    C
291    C
292    C
293    C
294    C
295    C
296    C
297    C
298    C
299    C
300    C
301    C
302    C
303    C
304    C
305    C
306    C
307    C
308    C
309    C
310    C
311    C
312    C
313    C
314    C
315    C
316    C
317    C
318    C
319    C
320    C
321    C
322    C
323    C
324    C
325    C
326    C
327    C
328    C
329    C
330    C
331    C
332    C
333    C
334    C
335    C
336    C
337    C
338    C
339    C
340    C
341    C
342    C
343    C
344    C
345    C
346    C
347    C
348    C
349    C
350    C
351    C
352    C
353    C
354    C
355    C
356    C
357    C
358    C
359    C
360    C
361    C
362    C
363    C
364    C
365    C
366    C
367    C
368    C
369    C
370    C
371    C
372    C
373    C
374    C
375    C
376    C
377    C
378    C
379    C
380    C
381    C
382    C
383    C
384    C
385    C
386    C
387    C
388    C
389    C
390    C
391    C
392    C
393    C
394    C
395    C
396    C
397    C
398    C
399    C
400    C
401    C
402    C
403    C
404    C
405    C
406    C
407    C
408    C
409    C
410    C
411    C
412    C
413    C
414    C
415    C
416    C
417    C
418    C
419    C
420    C
421    C
422    C
423    C
424    C
425    C
426    C
427    C
428    C
429    C
430    C
431    C
432    C
433    C
434    C
435    C
436    C
437    C
438    C
439    C
440    C
441    C
442    C
443    C
444    C
445    C
446    C
447    C
448    C
449    C
450    C
451    C
452    C
453    C
454    C
455    C
456    C
457    C
458    C
459    C
460    C
461    C
462    C
463    C
464    C
465    C
466    C
467    C
468    C
469    C
470    C
471    C
472    C
473    C
474    C
475    C
476    C
477    C
478    C
479    C
480    C
481    C
482    C
483    C
484    C
485    C
486    C
487    C
488    C
489    C
490    C
491    C
492    C
493    C
494    C
495    C
496    C
497    C
498    C
499    C
500    C
501    C
502    C
503    C
504    C
505    C
506    C
507    C
508    C
509    C
510    C
511    C
512    C
513    C
514    C
515    C
516    C
517    C
518    C
519    C
520    C
521    C
522    C
523    C
524    C
525    C
526    C
527    C
528    C
529    C
530    C
531    C
532    C
533    C
534    C
535    C
536    C
537    C
538    C
539    C
540    C
541    C
542    C
543    C
544    C
545    C
546    C
547    C
548    C
549    C
550    C
551    C
552    C
553    C
554    C
555    C
556    C
557    C
558    C
559    C
560    C
561    C
562    C
563    C
564    C
565    C
566    C
567    C
568    C
569    C
570    C
571    C
572    C
573    C
574    C
575    C
576    C
577    C
578    C
579    C
580    C
581    C
582    C
583    C
584    C
585    C
586    C
587    C
588    C
589    C
590    C
591    C
592    C
593    C
594    C
595    C
596    C
597    C
598    C
599    C
600    C
601    C
602    C
603    C
604    C
605    C
606    C
607    C
608    C
609    C
610    C
611    C
612    C
613    C
614    C
615    C
616    C
617    C
618    C
619    C
620    C
621    C
622    C
623    C
624    C
625    C
626    C
627    C
628    C
629    C
630    C
631    C
632    C
633    C
634    C
635    C
636    C
637    C
638    C
639    C
640    C
641    C
642    C
643    C
644    C
645    C
646    C
647    C
648    C
649    C
650    C
651    C
652    C
653    C
654    C
655    C
656    C
657    C
658    C
659    C
660    C
661    C
662    C
663    C
664    C
665    C
666    C
667    C
668    C
669    C
670    C
671    C
672    C
673    C
674    C
675    C
676    C
677    C
678    C
679    C
680    C
681    C
682    C
683    C
684    C
685    C
686    C
687    C
688    C
689    C
690    C
691    C
692    C
693    C
694    C
695    C
696    C
697    C
698    C
699    C
700    C
701    C
702    C
703    C
704    C
705    C
706    C
707    C
708    C
709    C
710    C
711    C
712    C
713    C
714    C
715    C
716    C
717    C
718    C
719    C
720    C
721    C
722    C
723    C
724    C
725    C
726    C
727    C
728    C
729    C
730    C
731    C
732    C
733    C
734    C
735    C
736    C
737    C
738    C
739    C
740    C
741    C
742    C
743    C
744    C
745    C
746    C
747    C
748    C
749    C
750    C
751    C
752    C
753    C
754    C
755    C
756    C
757    C
758    C
759    C
760    C
761    C
762    C
763    C
764    C
765    C
766    C
767    C
768    C
769    C
770    C
771    C
772    C
773    C
774    C
775    C
776    C
777    C
778    C
779    C
780    C
781    C
782    C
783    C
784    C
785    C
786    C
787    C
788    C
789    C
790    C
791    C
792    C
793    C
794    C
795    C
796    C
797    C
798    C
799    C
800    C
801    C
802    C
803    C
804    C
805    C
806    C
807    C
808    C
809    C
810    C
811    C
812    C
813    C
814    C
815    C
816    C
817    C
818    C
819    C
820    C
821    C
822    C
823    C
824    C
825    C
826    C
827    C
828    C
829    C
830    C
831    C
832    C
833    C
834    C
835    C
836    C
837    C
838    C
839    C
840    C
841    C
842    C
843    C
844    C
845    C
846    C
847    C
848    C
849    C
850    C
851    C
852    C
853    C
854    C
855    C
856    C
857    C
858    C
859    C
860    C
861    C
862    C
863    C
864    C
865    C
866    C
867    C
868    C
869    C
870    C
871    C
872    C
873    C
874    C
875    C
876    C
877    C
878    C
879    C
880    C
881    C
882    C
883    C
884    C
885    C
886    C
887    C
888    C
889    C
890    C
891    C
892    C
893    C
894    C
895    C
896    C
897    C
898    C
899    C
900    C
901    C
902    C
903    C
904    C
905    C
906    C
907    C
908    C
909    C
910    C
911    C
912    C
913    C
914    C
915    C
916    C
917    C
918    C
919    C
920    C
921    C
922    C
923    C
924    C
925    C
926    C
927    C
928    C
929    C
930    C
931    C
932    C
933    C
934    C
935    C
936    C
937    C
938    C
939    C
940    C
941    C
942    C
943    C
944    C
945    C
946    C
947    C
948    C
949    C
950    C
951    C
952    C
953    C
954    C
955    C
956    C
957    C
958    C
959    C
960    C
961    C
962    C
963    C
964    C
965    C
966    C
967    C
968    C
969    C
970    C
971    C
972    C
973    C
974    C
975    C
976    C
977    C
978    C
979    C
980    C
981    C
982    C
983    C
984    C
985    C
986    C
987    C
988    C
989    C
990    C
991    C
992    C
993    C
994    C
995    C
996    C
997    C
998    C
999    C
1000    C

```

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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF ** 121 71
 10A36d 71
 10B55a 71
 00002 UNDEF ** 81
 00003 UNDEF ** 91
 00004 UNDEF ** 10L
 00005 UNDEF ** 11L
 00006 60a 131
 00007 61d 151
 00008 64d 18L
 00009 66c 201
 00010 74a 23L

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

TABLE OF NAME'S 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	RU/2	4U	2D	1U
2	ARRY	R	1DIM ARRAY	DUM. ARG.	30P	11U/2	QU/2	6U	2P	1U
16	DISPRF		ENTRY		30U					
	FNJPI		EXTERNAL							
	FRAME		EXTERNAL		31U					
	GRAF		EXTERNAL		28U					
14	ICDUNI	I	VARIABLE		29U					
3	LENGTH	I	VARIABLE	DUM. ARG.	11U/2	10U/2	QU/2	RU/2		71.
	MAIN		ENTRY		30P	7N	2P/2	1U		
	TIKERM		EXTERNAL		27U					
10	XMAX	R	VARIABLE		29P	10S	10I	3S		
11	XMIN	R	VARIABLE		29P	8S	RU	4S		
12	YMAX	R	VARIABLE		29P	24S	23U	16U	14U	13U
					11U	5S			19U	19S
13	YMIN	R	VARIABLE		29P	25S	23U/2	21U	21S	18U
					9U	6S				9S

TABLE OF EXTERNAL NAME'S

CURVE EXTERNAL
 ENDP1 EXTERNAL
 FRAME EXTERNAL
 GRAF EXTERNAL
 TIKERM EXTERNAL

ABBREVIATIONS USED ABOVE (THOSE ARE KEYS TO THE SOURCE LISTING LINE NUMBER)

- A USED IN FORTRAN ASSIGN STATEMENT
- D DEFINED IN DECLARATIVE STATEMENT
- F STATEMENT NUMBER FINISHING A DO LOOP
- I INDEX OF A DO OR LIMITED DO LOOP
- J STATEMENT NUMBER USED IN TRANSFER
- L SOURCE LINE OF A STATEMENT NUMBER
- N NAME USED AS A DO LOOP PARAMETER
- P USED IN CALL/TUNE CALL OR ARRAY OFF
- R FORMAT USED IN A READ STATEMENT
- S STORED SO CONTINIS MAY BE CHANGED
- U NAME USED IN EXECUTABLE STATEMENT
- W FORMAT USED IN A WRITE STATEMENT
- X DEFINED OR DECLARED BUT NOT USED
- Z TFN 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-DISPRT 23-#IR 21-#CL

STATIC SPACE (IN OCTAL)	LENGTH
B SAVE	16
L SAVE	5
CONSTANTS	6
VARIABLES	7
TEMPORARIES	21
CODE	103
TOTAL	164

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

NUMBER USE SOURCE PROGRAM REFERENCES

5	UNDEF**	5L	4F
5A35b		4I	
5R43d		4L	
10	UNDEF**	15L	10F
10A57d		10I	
10R76a		10I	
00002	UNDEF**	11L	
00003	UNDEF**	12I	
00004	UNDEF**	13I	
00005	UNDEF**	14I	
00006	101a	16L	
00007	103d	18I	
00008	103d	18I	
00009	106d	21L	
00010	111c	23L	
00011	111c	23L	
00012	117a	26I	

(LSN=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	IDIM ARRAY	DUM. ARG.	5U
2	ARRY	R	IDIM ARRAY	DUM. ARG.	33P
	CURVE		EXTERNAL		33U
17	DISIPL		ENTRY		10/2
	ENDPL		EXTERNAL		34U
	FRAME		EXTERNAL		31U
	GRAF		EXTERNAL		32U
11	ICOUNT	I	VARIABLE		14U/2
3	LENGTH	I	VARIABLE	DUM. ARG.	33P
	MAIN		ENTRY		
4	SPACE	R	IDIM ARRAY	DUM. ARG.	33P
	TIKTRM		EXTERNAL		30U
12	XMAX	R	VARIABLE		32P
13	XMIN	R	VARIABLE		32P
14	YMAX	R	VARIABLE		14S
15	YMIN	R	VARIABLE		32P
					12S
					13S
					11S
					27S
					14U
					8S
					28S
					26U/2
					9S
					7U
					11U/2
					13U/2
					10W
					13U/2
					10I
					2P/2
					5U/2
					4I
					10
					2P
					8U
					9U
					12U/2
					14U/2
					14U/2
					12U/2
					4N
					3P
					11U/2
					7U
					11U/2
					13U
					6S
					11U
					7S
					19U
					18U
					19S
					17U
					17S
					16U
					24U
					24S
					23U
					22U
					22S
					21U
					33U
					34U

TABLE OF EXTERNAL NAMES

CURVE	EXTERNAL
ENDPL	EXTERNAL

FRAME EXTERNAL 31U
 GRAY EXTERNAL 32U
 THKERA EXTERNAL 30U

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

A USED IN FORTRAN ASSIGN STATEMENT P USED IN CALL/UNCL 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 IMP LID 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-DISTIP 30-#IR 21 #CL

STATIC SPACE (IN OCTAL)

B SAVE: 23
 T SAVE: 5
 CONSTANTS: 7
 VARIABLES: 7
 TEMPORARIES: 21
 CODE: 125
 TOTAL: 214

993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
DISCON

```

1.      subroutine discon (conarr,mdim,cnhold,kh,kv)
2.      real conarr (mdim,kv), cnhold (kh,kv)
3.      common work (5000)
C .....
C * This Subroutine Draws contour plots using DISPLA calls.
C * plotting the specified part of the array
C .....
C
4.      do 10 jcount=1,kh
5.      do 10 icount=1,kv
6.      cnhold(icount,jcount)=conarr(icount,jcount)
7.      10 continue
C
8.      set up the contour levels
9.      call hconak (cnhold,kh,kv,'scale')
C
10.     specify the appearance of the lines, and their labels
11.     call conlin (0,'solid','labels',1,3)
12.     call conang (60.)
13.     call conlth (0.06)
C
14.     draw the contour lines
15.     call concur (1,'labels','draw')
16.     call endpl(0)
17.     return
      end

```

VECTOR LOOP BEGINS AT SFQ. NO. 5. P= 35d

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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF... 71 5F 4E
 10A35d 41
 10B62C 41
 10C51b 51
 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

3	BCOMON		EXTERNAL		RU
	CONAN'S	R	2DIM ARRAY	DUM. ARG.	9P 10
	CONARR	R	2DIM ARRAY	DUM. ARG.	6S 2P 10
	CONLIN		EXTERNAL		6U 2D 10
	CONMAK		EXTERNAL		11U
	CONTHN		EXTERNAL		9U
	CONTUR		EXTERNAL		13U
	DISCON		EXTERNAL		14U
	ENDPI		ENTRY		1D/2
	HEIGHT		EXTERNAL		15U
	ICOUNT	I	EXTERNAL		10U
	JCOUNT	I	VARIABLE		6U/2 41
	KH	I	VARIABLE		5T 5I
	KV	I	VARIABLE	DUM. ARG.	9P 4N 2P 10
	MAIN	I	VARIABLE	DUM. ARG.	9P 2P/2 10
	ENTRY		ENTRY		

TABLE OF EXTERNAL NAMES

BCOMON	EXTERNAL	RU
CONAN'S	EXTERNAL	12U
CONARR	EXTERNAL	11U
CONLIN	EXTERNAL	9U
CONMAK	EXTERNAL	13U
CONTHN	EXTERNAL	14U
CONTUR	EXTERNAL	15U
ENDPL	EXTERNAL	10U
HEIGHT	EXTERNAL	

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

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

TABLE OF LOOPS ENCOUNTERED

LABEL INDEX	FROM	TO	ADDRESS	LENGTH
10 ICOUNT	4	7	35d	26
10 JCOUNT	5	7	51b	6

BLOCK NAMES AND LENGTHS IN OCTAL

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

STATIC SPACE (IN OCTAL)

R SAVE	27
I SAVE	2
CONSTANTS	17
VARIABLES	4
TEMPORARIES	31
CODE	101

TOTAL : 206

```

1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075

      C      subroutine prof3d (istop,restf,label,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 linp and
      C      the routine conint. In DISPLA, the curves are actually plotted in the
      C      routine distpl, rather than in prof3d itself.
      C
      C      array dimensions
      C
      C      parameter (nmnx=72, nmy=33, nnoz=88)
      C      parameter (nmnxp=nmnx+1, nmypp=nmy+1, nmnp=nmnz+1)
      C      parameter (nmnz2=nmnz+2, nmy2=nmy+2, nmnz2=nmnz+2)
      C
      C      plot variables
      C
      C      parameter (ndm1=72, ndm2=88)
      C      real vx(ndm1,ndm2), vy(ndm1,ndm2), vz(ndm1,ndm2)
      C      real rho(ndm1,ndm2)
      C      real rho(ndm1,ndm2), rho(ndm1,ndm2), rho(ndm1,ndm2)
      C      common /plivar/ vx, vy, vz, erg, rho, rho, rho, rhb
      C
      C      grid arrays
      C
      C      real xcor(nmnp), ycor(nmy), zcor(nnoz)
      C      real twdx(nmnp), twdy2(nmy2), twdz2(nmz2)
      C      real dtorx(nmnp), dtordy2(nmy2), dtordz2(nmz2)
      C
      C      common /holder/ space(ndm1,ndm2), valmin, valmax,
      C      space2(ndm1,ndm2)
      C
      C      common /qrdvar/ xcor, ycor, zcor, twdx, twdy2, twdz2, dtordx,
      C      dtordy2, dtordz2
      C
      C      common /hydfix/ rcamb,pcamb
      C
      C      common /glover/ ddim, dmax, cour, dt, dth, rlos, nfile
      C      , nstep, llix, lily, lllz
      C
      C      scratch space
      C      parameter (mny2d=mny2, nmz2, nspec=1, ivar=4, nspec)
      C      parameter (ndm12=(ndm1+2)*(ndm2+2))
      C      real dumv(ndm12, nvar), dumy(ndm12)
      C      common /trcom1/ dumv, dumy
      C
      C      pointers
      C      real arr(ndm1,ndm2), hcor(ndm1), vcor(ndm2)
      C      pointer (iparr,arr), (ipcor,hcor), (ipvcor,vcor)
      C      pointer (ipminh,minh), (ipmaxh,maxh)
      C      pointer (ipminv,minv), (ipmaxv,maxv)
      C
      C      local declarations
      C
      C      parameter (nhv=88)
      C      real pcor(ndm1,ndm2), mac(ndm1,ndm2), qcor(ndm1,ndm2)
      C      real dpr(ndm1,ndm2), out(nhv), scrh(nhv)
  
```

```

1076 20. equivalence (pre,dumv), (mac,dumv(1,2)), (gam,dumv(1,3))
1077 30. equivalence (dpr,dumv(1,4)), (eint,dumv(1,5)), (scrh,dumy)
1078 31. real user (ndim), pscr(ndim), qscr(ndim)
1079 32. real scr(ndim), scrb(ndim), scr(ndim)
1080 33. equivalence (pscr,mac), (pscr,mac(1,2)), (qscr,mac(1,3))
1081 34. equivalence (scr,mac(1,4)), (scr,mac(1,5)), (scr,mac(1,6))
1082 35. integer mplot(14)
1083 36. character *40 label
1084 37. character *8 restf
1085 38. real glab1(7), glab2(7), glab3(7)
1086 39. logical displa, darkhd
1087 40. common /qfcom/ mslice, lmin, lmax, jmin, jmax, kmin, kmax,
1088 1 mplot, lminp, colour, dmin, dmax, displa, jl, jl, darkhd
1089
1090 C
1091 C first print labels, use encode to set up the titles
1092 encode (llabel,3(0,glab1) llabel(1:nlablel)
1093 encode (4(0,320,glab2) time,istep,restf
1094 C
1095 C set window sizes,
1096 go to (10,20,30), mslice
1097 C
1098 C X-Y slice
1099 10 continue
1100 ipminh=loc(jmin)
1101 ipmaxh=loc(jmax)
1102 ipminy=loc(jmin)
1103 ipmaxy=loc(jmax)
1104 iphcor=loc(zcor)
1105 ipvcor=loc(zcor)
1106 go to 40
1107 C
1108 C Y-Z slice,
1109 20 continue
1110 ipminh=loc(jmin)
1111 ipmaxh=loc(jmax)
1112 ipminy=loc(kmin)
1113 ipmaxy=loc(kmax)
1114 iphcor=loc(zcor)
1115 ipvcor=loc(zcor)
1116 go to 40
1117 C
1118 C X-Z slice,
1119 30 continue
1120 ipminh=loc(lmin)
1121 ipmaxh=loc(lmax)
1122 ipminy=loc(kmin)
1123 ipmaxy=loc(kmax)
1124 iphcor=loc(xcor)
1125 ipvcor=loc(zcor)
1126 40 continue
1127 C
1128 C plot profiles,
1129 (spec=1
1130 do 240 ip=1,7
1131 if (mplot(ip) eq 0) go to 240

```

```

1132 C make DISPLA calls to set the page size, print type, plot area,
1133 C and draw the titles, or REAR calls to write the titles.
1134 C 50 if (displa) then
1135 C call page (10.,11.)
1136 C call nohdr
1137 C call area2d (6.,8.)
1138 C if (darkhd) then
1139 C call stchr (90.,1.,003,1)
1140 C call swissm
1141 C endif
1142 C call height (.16)
1143 C call headin (glab1,lab1-1,1.5,3)
1144 C call headin (glab2,25,1.5,3)
1145 C else
1146 C call pwrit (550.50,glab1,lab1,2,0,0)
1147 C call pwrit (550.25,glab2,40,1,0,0)
1148 C endif
1149 C
1150 C go to (50,100,110,130,1000,1010,1020), ip
1151 C
1152 C density profile
1153 C plots up to 3 species of absolute density, usually total density (1),
1154 C dust density (2), and explosive density (3).
1155 C 60 go to (70,80,90), ispec
1156 C label the axes
1157 C 70 if (displa) then
1158 C call yname ('distance (km)',13)
1159 C call yname ('density (g/cm**3)',17)
1160 C 91 else
1161 C call annotat (14hdistance, km, $.15hdensity, gm/cc$,1,1,0,0)
1162 C endif
1163 C set pointer to array
1164 C iparr=loc(rho)
1165 C go to 160
1166 C
1167 C label the axes
1168 C 80 if (displa) then
1169 C call yname ('distance (km)',13)
1170 C call yname ('density 2 (g/cm**3)',19)
1171 C 99 else
1172 C call annotat (14hdistance, km, $.17hdensity 2, gm/cc$,1,1,0,0)
1173 C endif
1174 C set pointer to array
1175 C iparr=loc(rho)
1176 C go to 160
1177 C
1178 C label the axes
1179 C 90 if (displa) then
1180 C call yname ('distance (km)',13)
1181 C call yname ('density 3 (g/cm**3)',19)
1182 C 107 else
1183 C call annotat (14hdistance, km, $.17hdensity 3, gm/cc$,1,1,0,0)
1184 C endif
1185 C set pointer to array
1186 C iparr=loc(rho)
1187 C go to 160
1188 C
1189 C energy profile
1190 C

```



```

1188 C
1189 C label the axes
1190 C 100
1191 C if (displa) then
1192 C   call xname ('distance (km)', 13)
1193 C   call yname ('energy (ergs/cm**3)', 19)
1194 C   else
1195 C     call axotat (14)distance, km, $.16)energy, ergs/cm**3, 1, 1, 0, 0)
1196 C   endif
1197 C   set pointer to array
1198 C   iparr=loc(iparr)
1199 C   go to 160
1200 C
1201 C dynamic pressure profile.
1202 C label the axes
1203 C 110
1204 C if (displa) then
1205 C   call xname ('distance (km)', 13)
1206 C   call yname ('dynamic pressure (dynes/cm**2)', 30)
1207 C   else
1208 C     call axotat(14)distance, km, $.
1209 C     30)dynamic pressure, dynes/cm**2$, 1, 1, 0, 0)
1210 C   endif
1211 C   set pointer to array
1212 C   iparr=loc(iparr)
1213 C   calculate the dynamic pressure from momenta and density
1214 C   do 120 iv minv,maxv
1215 C     dpr=(nh,iv)-0.5*(rvx(nh,iv)+rvy(nh,iv)+rvz(nh,iv)+rho(nh,iv)
1216 C       *rvz(nh,iv)+rvz(nh,iv))/rho(nh,iv)
1217 C     go to 160
1218 C
1219 C pressure profile.
1220 C this is a profile of absolute pressure.
1221 C label the axes
1222 C 130
1223 C if (displa) then
1224 C   call xname ('distance (km)', 13)
1225 C   call yname ('pressure (dynes/cm**2)', 22)
1226 C   else
1227 C     call axotat (14)distance, km, $.22)pressure, dynes/cm**2$,
1228 C     1, 1, 0, 0)
1229 C   endif
1230 C   set pointer to array
1231 C   iparr=loc(iparr)
1232 C   calculate the dynamic pressure, for use in calculating pressure
1233 C   do 135 iv minv,maxv
1234 C     do 135 nb minb,maxb
1235 C       dpr=(nb,iv)-0.5*(rvx(nb,iv)+rvy(nb,iv)+rvz(nb,iv)+rho(nb,iv)
1236 C         *rvz(nb,iv)+rvz(nb,iv))/rho(nb,iv)
1237 C       calculate the pressure, using equations of state subroutine eos
1238 C       do 150 nv minv,maxv
1239 C         eint(nb,iv)=erg(nb,iv)-dpr(nb,iv)
1240 C         call eos (maxb-minb+1, nspec, rho(minh,nv), eint(minh), gam(minh)
1241 C           ,nv, p(minh,nv), rsc(minh), pscr(minh), qscr(minh), scra(minh)
1242 C           ,scrb(minh), scrc(minh), cha(minh,nv), chb(minh,nv))
1243 C         continue
1244 C       2
1245 C     150
1246 C   go to 160

```

```

1244 C
1245 C Renormalize density -- with respect to ambient density
1246 C label the axes
1247 1000 if (display) then
1248   call xname ('distance (km)', 13)
1249   call yname ('renormalized density', 22)
1250 else
1251   call aunitat (14, distance, cm, $)
1252   2threnormalized density$, 1, 1, 0, 0)
1253 endif
1254 C set pointer to array
1255   ipar=loc(space2)
1256 C renormalize the density with the ambient
1257 do 1002 jv=mnv,maxv
1258   rscr(jv)=cvmgt(rcomb(1plane),rcamb(jjv+1),mslice,eq, 1)
1259 do 1002 ih=minh,maxh
1260   space2(ih,jv)=rho(ih,jv)/rscr(jjv)
1261 continue
1262 go to 160
1263
1264 C Renormalized pressure
1265 C label the axes
1266 1010 if (display) then
1267   call xname ('distance (km)', 13)
1268   call yname ('renormalized pressure', 21)
1269 else
1270   call aunitat (14, distance, cm, $)
1271   2threnormalized pressure$, 1, 1, 0, 0)
1272 endif
1273
1274 C set pointer to the array
1275   ipar=loc(space2)
1276 C calculate dynamic pressure to use in calculating pressure
1277 do 1011 iv=mnv,maxv
1278   do 1011 nh=minh,maxh
1279     dpr(nh,nv)=0.5*(vrx(nh,nv)*vrx(nh,nv)+rvy(nh,nv)*rvy(nh,nv)
1280     +rvz(nh,nv)*rvz(nh,nv))/rho(nh,nv)
1281     calculate pressure (absolute) from equations of state routine eos
1282     do 1013 iv=mnv,maxv
1283       do 1012 nh=minh,maxh
1284         eint(nh)=erg(nh,nv)-dpr(nh,nv)
1285         call eos (maxh-minh+1, nspec, rho(minh,nv), eint(minh), gam(minh)
1286         ,nv), p1(minh,nv), rscr(minh), pscr(minh), dscr(minh), scra(minh)
1287         ,scrh(minh), scrc(minh), cha(minh,nv), cha(minh,nv))
1288         continue
1289 C renormalize density with the ambient
1290 do 1015 jv=mnv,maxv
1291   pscr(jv)=cvmgt(pcomb(1plane),pcamb(jjv+1),mslice,eq, 1)
1292 do 1015 ih=minh,maxh
1293   space2(ih,jv)=p1(ih,jv)/pscr(jjv)
1294 continue
1295 go to 160
1296
1297 C Z velocity profile
1298 C label the axes
1299 1020 if (display) then
1300   call xname ('distance (km)', 13)

```

```

1300      call vcomp ('z velocity (cm/sec)', 19)
1301      else
1302      call arotal (14, distance, cm, $)
1303      19hz velocity, cm/sec$, 1, 1, 0, 0)
1304      endif
1305      c set pointer to array
1306      iparr = loc(space2)
1307      calculate z velocity from z momentum and density
1308      do 1022 jv-minv, maxv
1309      do 1022 ih-minh, maxh
1310      space2(iih, jv) = rvz(iih, jv) / rho(iih, jv)
1311      continue
1312      go to 160
1313      c
1314      160 mpabs = labs(mplot(ip))
1315      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 DISPLA,
1321      c the actual drawing is done in the routine displ, which converts from
1322      c centimeters to kilometers and plots the proper portion of the grid.
1323      c X-Y slice.
1324      170 if (displa) then
1325      if (mplot(ip).gt.0) then
1326      encode(31, 250, glab3) mpabs, vcor(mpabs) / 100000.
1327      1plane, zcor(1plane) / 100000.
1328      call headin (glab3, 30, 1.5, 3)
1329      call displ (hcor(minh), arr(minh, mpabs), maxh, minht, space)
1330      else
1331      encode(31, 260, glab3) mpabs, hcor(mpabs) / 100000.
1332      1plane, zcor(1plane) / 100000.
1333      call headin (glab3, 30, 1.5, 3)
1334      do 180 iv-minv, maxv
1335      scrh(iv) = arr(mpabs, iv)
1336      call displ (vcor(minv), scrh(minv), maxv-minvt, space)
1337      endif
1338      else
1339      if (mplot(ip).gt.0) then
1340      encode(39, 255, glab3) mpabs, vcor(mpabs), 1plane, zcor(1plane)
1341      call pwr it (550, 1010, glab3, 39, 1, 0, 0)
1342      call ezxy (hcor(minh), arr(minh, mpabs), maxh, minht, th$)
1343      else
1344      encode(39, 265, glab3) mpabs, hcor(mpabs), 1plane, zcor(1plane)
1345      call pwr it (550, 1010, glab3, 39, 1, 0, 0)
1346      do 185 iv-minv, maxv
1347      scrh(iv) = arr(mpabs, iv)
1348      call ezxy (vcor(minv), scrh(minv), maxv-minvt, th$)
1349      endif
1350      endif
1351      go to 230
1352      c
1353      c Y Z slice.
1354      190 if (displa) then
1355      if (mplot(ip).gt.0) then

```

```

1356   encode(31,270,glab3)mpabs,vcor(mpabs)/100000.
1357   lplane,vcor(lplane)/100000.
1358   call headin (glab3,30,1.5,3)
1359   call distpl (hcor(minh),arr(minh,mpabs),maxh-minh+1,space)
1360   else
1361   encode(31,280,glab3)mpabs,hcor(mpabs)/100000.
1362   lplane,vcor(lplane)/100000.
1363   do 200 mv=minv,maxv
1364   scrh(nv)-arr(mpabs,nv)
1365   call distpl (vcor(minv),scrh(minv),maxv-minv+1,space)
1366   endif
1367   else
1368   if (plot(ip).gt.0) then
1369   encode(39,275,glab3)mpabs,vcor(mpabs),lplane,vcor(lplane)
1370   call pwrit (550,1010,glab3,39,1.0,0)
1371   call ezxy (hcor(minh),arr(minh,mpabs),maxh-minh+1,th$)
1372   else
1373   encode(39,285,glab3)mpabs,hcor(mpabs),lplane,vcor(lplane)
1374   call pwrit (550,1010,glab3,39,1.0,0)
1375   do 205 mv=minv,maxv
1376   scrh(nv)-arr(mpabs,nv)
1377   call ezxy (vcor(minv),scrh(minv),maxv-minv+1,th$)
1378   endif
1379   endif
1380   go to 230
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411

```

C X Z slice

210

220

225

C

```

1412 1412 go back and plot the other species of density. if any
1413 230      ispec = spec(i)
1414 273      if (ispec.le.nspec) and (p.eq. 1) go to 50
1415 274      continue
1416 275      return
1417
1418
1419      format statements for the titles of these plots
1420 276 format ('j.v.kz.z=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1421 277 format ('i.x.kz.z=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1422 278 format ('kz.z.i.x.x=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1423 279 format ('j.v.i.x.x=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1424 280 format ('kz.z.j.v.y=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1425 281 format ('i.x.i.j.v.y=', i3., ' ', f5.1, ' ', ix, i3., ' ', f5.1, '$')
1426 282 format (a)
1427 283 format ('j.v.kz.z=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1428 284 format ('i.x.kz.z=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1429 285 format ('kz.z.i.x.x=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1430 286 format ('j.v.i.x.x=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1431 287 format ('kz.z.j.v.y=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1432 288 format ('i.x.i.j.v.y=', i3., ' ', e9.3, ' ', ix, i3., ' ', e9.3, '$')
1433 289 format ('time=', f5.1, ' sec, step', i5., ' dump', a)
1434 290      end
1435      VECTOR LOOP BEGINS AT SEQ. NO. 128. P= 1031d
1436      VECTOR LOOP BEGINS AT SEQ. NO. 129. P= 1113d
1437      VECTOR LOOP BEGINS AT SEQ. NO. 142. P= 1152c
1438      VECTOR LOOP BEGINS AT SEQ. NO. 156. P= 1265b
1439      VECTOR LOOP BEGINS AT SEQ. NO. 168. P= 1345d
1440      VECTOR LOOP BEGINS AT SEQ. NO. 171. P= 1404c
1441      VECTOR LOOP BEGINS AT SEQ. NO. 177. P= 1474a
1442      VECTOR LOOP BEGINS AT SEQ. NO. 189. P= 1554c
1443      VECTOR LOOP BEGINS AT SEQ. NO. 203. P= 1612c
1444      VECTOR LOOP BEGINS AT SEQ. NO. 215. P= 1742b
1445      VECTOR LOOP BEGINS AT SEQ. NO. 229. P= 2071c
1446      VECTOR LOOP BEGINS AT SEQ. NO. 241. P= 2221b
1447      VECTOR LOOP BEGINS AT SEQ. NO. 255. P= 2350c
1448      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	71I	
60 660a	87I	86J	
70 666a	88I	87J	
80 711b	96I	87J	
90 734C	104L	87J	
100 757I	112I	86J	
110 1003a	120I	86J	
120 UNDEF**	129I	128F	127F
120A1031d	127I		
120R1064C	127I		
120C1042b	128I		
120R1061d	128I		
130 1065a	131I	86J	
135 UNDEF**	140I	139F	138E
135A113d	138I		
135R1146C	138I		
135C1124b	139I		
135D1143d	139I		
140 UNDEF**	143I		142E
140A1163d	142I		
140B1175b	142L		
150 UNDEF**	145I		141E
150A1152C	141L		
150R1236a	141I		
160 1601d	193I	192J	180J 159J 146J 130J 111J 103J
	95J		
170 1612C	195I	194J	
180 UNDEF**	204I	203F	
180A1721b	203I		
180R1727b	203I		
185 UNDEF**	216I	215F	
185A2050C	215L		
185R2096C	215I		
190 2071C	221I	194J	
200 UNDEF**	230I	229F	
200A2200b	229I		
200R2206b	229I	241F	
205 UNDEF**	242I		
205A2327C	241I		
205R2335C	241I		
210 2350C	247I	194J	
220 UNDEF**	256I	255E	
220A2457b	255I		
220R2465b	255I		
225 UNDEF**	268I	267F	
225A2606C	267I		

225R2614c	267L		
230 2627a	272I		
240 2633b	274I		
240A573a	69I	246J	220J
240RUNDIT**	69I	70I	69I
250 FN	276I	197R	
255 FN	283I	209R	
260 FN	277I	201R	
265 FN	284L	213R	
270 FN	278I	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**	158I	156I	154E
1002A1265b	154L		
1002B1316c	154I		
1002C1304b	156L		
1002D1313d	156I		
1010 1317a	160L	86J	167F
1011 UNDEF**	169I	168F	
1011A1343i	167I		
1011R1400c	167L		
1011C1356b	168L		
1011D1375d	168L		
1012 UNDEF**	172I	171F	
1012A1416b	171I		
1012R1427b	171L		
1013 UNDEF**	174I	170F	
1013A1404c	170I		
1013R1470a	170I		
1015 UNDEF**	179L	177I	175F
1015A1474a	175I		
1015B1525b	175I		
1015C1513a	177L		
1015D1522c	177I		
1020 1525d	181L	86J	
1022 UNDEF**	131L	189F	188E
1022A1554c	188I		
1022B1601b	188L		
1022C1565a	189L		
1022D1576c	189L		
0XX002 527a	43W		
0XX003 631b	71I		
0XX004 613a	75L		
0XX005 650a	82L		
0XX006 654c	86W		
0XX007 664c	87W		
0XX008 701a	88L		
0XX009 707a	91L		

00010	724b	961
00011	732b	991
00012	747c	1041
00013	755c	1071
00014	772d	1121
00015	10X0b	1151
00016	1016a	1201
00017	1024a	1231
00018	1100a	1311
00019	1106a	1341
00020	1251c	1471
00021	1257c	1501
00022	1332a	1601
00023	1340a	1631
00024	1540b	1811
00025	1546a	1841
00026	1611a	194W
00027	1742b	195L
00028	1660c	196L
00029	1741d	200L
00030	2071a	2071
00031	2007c	208L
00032	2071a	2121
00033	2221b	2211
00034	2137c	222L
00035	2220d	226L
00036	2350a	2331
00037	2266c	234L
00038	2350a	2381
00039	2500b	247L
00040	2416c	248L
00041	2477d	252L
00042	2627a	259L
00043	2545c	260L
00044	2627a	2641

(SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
 (FN-FORMAT NUMBER, UNDEF-UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENUMERATED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS NAME	TYPE	MAIN USAGE	BLOCK	SOURCE	PROGRAM REFERENCES
\$FFA	EXTERNAL			42U	41U
\$FF	EXTERNAL			265U	261U
				209U	201U
\$FF1	EXTERNAL			265U	261U
				209U	201U
\$FFV	EXTERNAL			265U/4	261U/4
				209U/4	201U/4
AND1A1	EXTERNAL			185U	151U
AREA2D	EXTERNAL			74U	
1 ARR	R	2DIM ARRAY	POINTEE	268U	263P
				211P	204U
CVMGT		INLINE FUNCT.		176U	155U
				239U	239U
				41U	41U
				239U	239U
				41U	41U
				239U/4	239U/4
				239U/4	239U/4
				108U	100U
				116U	100U
				237P	225P
				230U	225P
				242U	22D
				251P	23P
				194p	23P
				256U	251P
				194p	23P
				151U	135U
				164U	164U
				263P	263P
				204U	204U
				176U	155U
				257U	257U
				197U	197U
				197U	197U
				253U/4	253U/4
				197U/4	197U/4
				42U/2	42U/2
				124U	124U
				242U	242U
				22D	22D
				235U	235U
				227U	227U
				223U	223U
				223U/4	223U/4
				100U	92U
				230U	216U

NO	PARM	UNIT	DESCRIPTION	DATA	UNIT	DESCRIPTION	DATA	UNIT	DESCRIPTION	DATA		
34	DARK01	I	VARIABLE	75U	40D	39D	181U	160U	147U	131U	120U	112U
31	DISPLA	I	VARIABLE	247U	221U	195U	71U	40U	39D			
				104U	96U	88U	225U	205U	198U			
				257U	251U	231U	140S	129S	126P	30D	28D	
47014	DPR	R	2D EQ. ARRAY	172U	169S	143U	140S	129S	126P			
64020	FINI	R	1D EQ. ARRAY	173P	172S	144P	143S	30D	28D			
				173U	144U	144U						
45100	ERG	R	201M ARRAY	172U	143U	118P	9D	7D				
				269U	263U	243U	237U	217U	211U			
32010	GAM	R	EX1EPNAL	173P	144P	79D	77U					
413	GLAB1	R	2D EQ. ARRAY	83P	80P	41S	38D					
422	GLAB2	R	1D1M ARRAY	84P	81P	42S	38D					
431	GLAB3	R	1D1M ARRAY	266P	265S	262P	261S	254P	253S	250P	249S	240P
				210P	209S	202P	201S	198P	197S	38D	38D	213S
15224	GSCR	R	1D EQ. ARRAY	173P	144P	33D	31D	239U	237P	227U	225P	213U
2	HCOR	R	1D1M ARRAY	265U	263P	253U	251P	22D				
				211P	201U	199P	23P	22D				
				254U	250U	228U	224U	202U	198U	81U	80U	
				79U								
				190U/3	189I	178U/2	177I	157U/2	156I			
				62P	46P	40D						
445	I101	I	VARIABLE	61P	45P	40D						
2	IMAX	I	VARIABLE	273U	260U	248U	234U	222U	208U	196U	193U	86P
1	IMIN	I	VARIABLE	70U	69I							
441	IP	I	VARIABLE	187S	166S	153S	137S	126S	118S	110S	102S	94S
404	IPARP	*I	VARIABLE	23P								
405	IPICOR	*I	VARIABLE	65S	57S	49S	23P					
410	IPMAX1	*I	VARIABLE	62S	54S	46S	24P					
412	IPMAXV	*I	VARIABLE	64S	56S	48S	25P					
407	IPMIN4	*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				
				42U	1D							
444	J1V	I	VARIABLE	190U/3	188I	178U/3	176U/2	175I	157U/3	155U/2	154I	
4	JMAX	I	VARIABLE	54P	48P	40D						
3	JMIN	I	VARIABLE	53P	47P	40D						
6	KMAX	I	VARIABLE	64P	56P	40D						
5	KMIN	I	VARIABLE	63P	55P	40D						
3	LABEL	CH	VARIABLE	41U	36D	1D						
				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	269U/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			
				263P	251P	237P	225P	211P	198P	189N	177N	173P
				171N	168N	156N	144P	139N	139N	129N	129N	217P
7	MAXV	I	VARIABLE	215N	205P	203N	188N	175N	170N	167N	154N	141N
				138N	127N	25P						
4	MIN1	I	VARIABLE	263U/3	251U/3	237U/3	225U/3	211U/3	198U/3	189N	177N	173U/?
				171N	168N	156N	144U/?	142N	139N	139N	129N	24P
6	MINV	I	VARIABLE	269U/3	267N	257U	255N	243P	241N	231P	229N	217U/?
				269U/3	267N	257U	255N	243U/3	241N	231U/3	229N	217U/3

```

446 MPARS I VARIABLE
    215N 205U/3 203N 188N 175N 170N 167N 154N 141N
    13RN 127N 25P
    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 1935
    260U 248U 244U 238U 193P 70U 400
    35D
    194P 176P 155P 43P 40D
    32P/3 31P/3 28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3
    55
    20P/2 195
    28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3 55
    172U/3 1711 169U/R 1681 143U/3 1421 140U/R 1391 129U/R
    1281
    26S
    83P/2 26S
    83P 80P 41U 41P 1D
    25
    11P 10P 35
    10P 25
    12P 11P 45
    35
    10P 25
    15P/2 12P 11P 45
    185
    73U
    273U 173P 144P 185
    268U/2 2671 256U/2 2551 242U/2 2411 230U/2 2291
    2151 204U/2 2031 173U/5 172U/2 1701 167U/R 1671 144U/5
    143U/2 1411 140U/R 1381 129U/R 1271
    20P 185
    72U
    176P/2 16D
    178U 173P 144P 137P 29D 27D
    1D/2
    176S 173P 144P 33D 31D 83U
    265U 240U 236U 214U 210U
    155P/2 16D 15D
    42U 37D 1D
    173P 144P 102P 9D 8D
    173P 144P 110P 9D 8D
    190U 173P 169U 157U 144P 140U 124U 94P 9D
    8D
    173P 157U 1555 144P 33D 31D
    169U/2 140U/2 129U/2 9D 6D
    169U/2 140U/2 129U/2 9D 6D
    190U 169U/2 140U/2 129U/2 9D 6D
    173P 144P 34D 37D
    173P 144P 34D 32D
    173P 144P 34D 32D
    259P 2685 257P 2565 243P 231P 230S 217P
    2165 205P 30D
    76U
    257P 251P 205P 199P 13D
    190S 187P 1783 166P 1575 153P 11D
  
```

```

443 NH1 I VARIABLE
    215N 205U/3 203N 188N 175N 170N 167N 154N 141N
    13RN 127N 25P
    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 1935
    260U 248U 244U 238U 193P 70U 400
    35D
    194P 176P 155P 43P 40D
    32P/3 31P/3 28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3
    55
    20P/2 195
    28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3 55
    172U/3 1711 169U/R 1681 143U/3 1421 140U/R 1391 129U/R
    1281
    26S
    83P/2 26S
    83P 80P 41U 41P 1D
    25
    11P 10P 35
    10P 25
    12P 11P 45
    35
    10P 25
    15P/2 12P 11P 45
    185
    73U
    273U 173P 144P 185
    268U/2 2671 256U/2 2551 242U/2 2411 230U/2 2291
    2151 204U/2 2031 173U/5 172U/2 1701 167U/R 1671 144U/5
    143U/2 1411 140U/R 1381 129U/R 1271
    20P 185
    72U
    176P/2 16D
    178U 173P 144P 137P 29D 27D
    1D/2
    176S 173P 144P 33D 31D 83U
    265U 240U 236U 214U 210U
    155P/2 16D 15D
    42U 37D 1D
    173P 144P 102P 9D 8D
    173P 144P 110P 9D 8D
    190U 173P 169U 157U 144P 140U 124U 94P 9D
    8D
    173P 157U 1555 144P 33D 31D
    169U/2 140U/2 129U/2 9D 6D
    169U/2 140U/2 129U/2 9D 6D
    190U 169U/2 140U/2 129U/2 9D 6D
    173P 144P 34D 37D
    173P 144P 34D 32D
    173P 144P 34D 32D
    259P 2685 257P 2565 243P 231P 230S 217P
    2165 205P 30D
    76U
    257P 251P 205P 199P 13D
    190S 187P 1783 166P 1575 153P 11D
  
```

```

    215N 205U/3 203N 188N 175N 170N 167N 154N 141N
    13RN 127N 25P
    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 1935
    260U 248U 244U 238U 193P 70U 400
    35D
    194P 176P 155P 43P 40D
    32P/3 31P/3 28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3
    55
    20P/2 195
    28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3 55
    172U/3 1711 169U/R 1681 143U/3 1421 140U/R 1391 129U/R
    1281
    26S
    83P/2 26S
    83P 80P 41U 41P 1D
    25
    11P 10P 35
    10P 25
    12P 11P 45
    35
    10P 25
    15P/2 12P 11P 45
    185
    73U
    273U 173P 144P 185
    268U/2 2671 256U/2 2551 242U/2 2411 230U/2 2291
    2151 204U/2 2031 173U/5 172U/2 1701 167U/R 1671 144U/5
    143U/2 1411 140U/R 1381 129U/R 1271
    20P 185
    72U
    176P/2 16D
    178U 173P 144P 137P 29D 27D
    1D/2
    176S 173P 144P 33D 31D 83U
    265U 240U 236U 214U 210U
    155P/2 16D 15D
    42U 37D 1D
    173P 144P 102P 9D 8D
    173P 144P 110P 9D 8D
    190U 173P 169U 157U 144P 140U 124U 94P 9D
    8D
    173P 157U 1555 144P 33D 31D
    169U/2 140U/2 129U/2 9D 6D
    169U/2 140U/2 129U/2 9D 6D
    190U 169U/2 140U/2 129U/2 9D 6D
    173P 144P 34D 37D
    173P 144P 34D 32D
    173P 144P 34D 32D
    259P 2685 257P 2565 243P 231P 230S 217P
    2165 205P 30D
    76U
    257P 251P 205P 199P 13D
    190S 187P 1783 166P 1575 153P 11D
  
```

```

    215N 205U/3 203N 188N 175N 170N 167N 154N 141N
    13RN 127N 25P
    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 1935
    260U 248U 244U 238U 193P 70U 400
    35D
    194P 176P 155P 43P 40D
    32P/3 31P/3 28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3
    55
    20P/2 195
    28P 27P/3 22P/2 13P/2 8P/3 7P 6P/3 55
    172U/3 1711 169U/R 1681 143U/3 1421 140U/R 1391 129U/R
    1281
    26S
    83P/2 26S
    83P 80P 41U 41P 1D
    25
    11P 10P 35
    10P 25
    12P 11P 45
    35
    10P 25
    15P/2 12P 11P 45
    185
    73U
    273U 173P 144P 185
    268U/2 2671 256U/2 2551 242U/2 2411 230U/2 2291
    2151 204U/2 2031 173U/5 172U/2 1701 167U/R 1671 144U/5
    143U/2 1411 140U/R 1381 129U/R 1271
    20P 185
    72U
    176P/2 16D
    178U 173P 144P 137P 29D 27D
    1D/2
    176S 173P 144P 33D 31D 83U
    265U 240U 236U 214U 210U
    155P/2 16D 15D
    42U 37D 1D
    173P 144P 102P 9D 8D
    173P 144P 110P 9D 8D
    190U 173P 169U 157U 144P 140U 124U 94P 9D
    8D
    173P 157U 1555 144P 33D 31D
    169U/2 140U/2 129U/2 9D 6D
    169U/2 140U/2 129U/2 9D 6D
    190U 169U/2 140U/2 129U/2 9D 6D
    173P 144P 34D 37D
    173P 144P 34D 32D
    173P 144P 34D 32D
    259P 2685 257P 2565 243P 231P 230S 217P
    2165 205P 30D
    76U
    257P 251P 205P 199P 13D
    190S 187P 1783 166P 1575 153P 11D
  
```

SWISSIA	EXTERNAL	77U
3 TIME	R VARIABT	42U
3 YCOR	R 1DIM ARRAY	261U 205P
O XCOR	R 1DIM ARRAY	49P
XNAME	EXTERNAL	14U
111 YCOR	R 1DIM ARRAY	182U 265U
152 ZCOR	R 1DIM ARRAY	183U 239U 58P

TABLE OF EXTERNAL NAMES

\$EFA	EXTERNAL	42U	253U	249U	239U	235U	227U	223U	213U
\$EFF	EXTERNAL	265U	209U	197U	42U	41U			
\$EFI	EXTERNAL	265U	261U	253U	249U	239U	235U	227U	213U
\$EFV	EXTERNAL	265U/4	261U/4	253U/4	249U/4	239U/4	235U/4	227U/4	213U/4
AROTAT	EXTERNAL	209U/4	201U/4	197U/4	42U/2	124U	116U	108U	92U
AREA2D	EXTERNAL	185U	164U	151U	135U				
DISIPL	EXTERNAL	74U							
EOS	EXTERNAL	257U	251U	231U	225U	205U	199U		
FZXY	EXTERNAL	173U	144U						
HEADIN	EXTERNAL	269U	263U	243U	237U	217U	211U	81U	80U
HEIGHT	EXTERNAL	254U	250U	228U	224U	202U	198U		
NOBRDR	EXTERNAL	79U							
PAGE	EXTERNAL	73U							
PWRIT	EXTERNAL	266U	262U	240U	236U	214U	210U	83U	
SHDCHR	EXTERNAL	76U							
SWISSM	EXTERNAL	182U	161U	148U	132U	121U	113U	105U	89U
XNAME	EXTERNAL	183U	162U	149U	133U	122U	114U	106U	90U
YNAME	EXTERNAL								

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

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

NDM1	= 72
NDM2	= 88
NDM3	= 72
NDM4	= 73
NDM5	= 35
NDM6	= 88
NDM7	= 89

NDM12	= 6660
NDM13	= 88
NDM14	= 74
NDM15	= 33
NDM16	= 34
NDM17	= 90
NDM18	= 3150

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 JUV	154	158	1265b	32
1002 ITH	156	158	1304b	7
1011 NV	167	169	1345d	34
1011 NH	168	169	1346b	17
1013 NV	170	174	1404c	64
1012 NH	171	172	1416b	11
1015 JUV	175	179	1474a	31
1015 ITH	177	179	1513a	7
1022 JUV	188	191	1554c	25
1022 ITH	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	2600c	6

BLOCK NAMES AND LENGTHS IN OCTAL

2641-PROFID	57 #1R	705 #CL	30602-HOLDER	1116-GRDVAR
264-HYDFIX	14 GLOVAR	116030 IDCOM1	126500 PLTVAR	35 GRFCOM

STATIC SPACE (IN OCTAL)

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

```

1434 1.      subroutine fileio (ifetch,nf)
1435 c
1436 c      performs disk reads using the CFILIR random access I/O routine
1437 c      RDABSF. checks for I/O completion before returning to calling
1438 c      program.
1439 c
1440 c      array dimensions
1441 c      parameter (nmx=72, nmy=33, nmz=88)
1442 c      parameter (mxyz2d=(nmy*2)*(nmz*2))
1443 c      parameter (hydro arrays)
1444 c      parameter (nspec=1, nvar=4+nspec, npl=1)
1445 c      parameter (jkval=nvar+mxyz2d)
1446 c      real dvar(mxyz2d,nvar,npl)
1447 c
1448 c      common /hydro/ dvar
1449 c      real var(mxyz2d,nvar,npl)
1450 c      pointer (ipvar,var)
1451 c      logical ifetch, lstore
1452 c      character *8 file
1453 c
1454 c      set pointer
1455 c      ipvar=loc(dvar)
1456 c
1457 c      perform I/O.
1458 c      call rdabst (lunit,var(1,1,nf),jkvar,(ifetch 1),jkvar)
1459 c      return
1460 c
1461 c      entry fileio(file,lun,nsiz)
1462 c
1463 c      entry FILEIO opens the disk file.
1464 c      the arguments:
1465 c      file - a real variable containing the hollerith string filename
1466 c      lun  - logical unit number
1467 c      nsiz - file size (words)
1468 c
1469 c      lunit=lun
1470 c      nsize=nsiz
1471 c
1472 c      open the disk file.
1473 c      call assign (lunit,file,0)
1474 c      call fawait (lunit,1)
1475 c      call famsiz (lunit,nsize)
1476 c      return
1477 c
1478 c      entry filefc
1479 c
1480 c      entry FILEFC closes the disk file after ensuring I/O completion.
1481 c
1482 c      confirm that all previous disk I/O requests have been completed
1483 c      if (unit(lunit)) 10,30,40)
1484 c
1485 c      close the file.
1486 c      continue
1487 c      call close (lunit)
1488 c      return
1489 c

```

```

1490          entry fileld
1491          C
1492          C entry FILED destroys the scratch disk I/O file.
1493          C
1494          28 call drabst (unit,file,0,terr)
1495          29 write (6,70) terr
1496          30 return
1497          C
1498          entry fileck
1499          C
1500          C entry FILECK waits for all pending I/O requests to be completed
1501          C before returning to the calling program.
1502          C
1503          32 if (unit(lunit)) 20,30,40
1504          C
1505          33 return
1506          C
1507          C error messages.
1508          34 write (6,50) lunit
1509          35 stop
1510          C
1511          36 write (6,60) lunit
1512          37 stop
1513          C
1514          38 50 format ('/ EOF or (DI) condition present for unit ',lunit - ',(3)
1515          39 60 format ('/// I/O error occurred for unit ',lunit - ',(3)
1516          40 70 format ('//I/O error from subroutine DRABSF',(3)
1517          41 end
    
```

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

NUMBER USE SOURCE PROGRAM REFERENCES

10	UNDEF**	24L	23I	23J
20	UNDEF**	33L	32I	32J
30	217C	34I	32J	23J
40	230a	36L	32J	23J
50	FN	38I	34W	
60	FN	39I	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
\$WFI		EXTERNAL		36U	29U
\$WFV		EXTERNAL		36U	29U
ASSIGN		EXTERNAL		18U	
CLOSE		EXTERNAL		25U	
DRAWST		EXTERNAL		28U	
O DVAR	R	3DIM ARRAY	HYDVAR	12P	7D
FAMS1Z		EXTERNAL		20U	
FAMWAT		EXTERNAL		19U	
3 FILE	CH	VARIABLE	DUM.ARG.	28P	18P
204 FILECK		ENTRY		31D/2	
136 FILEFC		ENTRY		22D/2	
155 FILEFD		ENTRY		27D/2	
102 FILEFE		ENTRY		15D/2	
50 FILEFI		ENTRY		31D	15D
46 IERR	I	VARIABLE		27D	22D
1 JFETCH	I	VARIABLE	DUM.ARG.	28P	1D
41 D'VAR	I	VARIABLE		13U	9P
JKVAR	I	PARAMETER		13U	13P
LOC	I	INLINE FUNCT		12U	
4 LUN	I	VARIABLE	DUM.ARG.	16U	15D
44 L'UNIT	I	VARIABLE		36U	34U
				16S	13P
MAIN.		ENTRY		13U	1D
2 NF	I	VARIABLE	DUM.ARG.	25	
NNJX	I	PARAMETER		25	
NNNY	I	PARAMETER		25	
NNNZ	I	PARAMETER		25	
NNY2D	I	PARAMETER		8P	35
NPL	I	PARAMETER		8P	45
5 NS1Z	I	VARIABLE	DUM.ARG.	17U	15D
45 NS1ZE	I	VARIABLE		20P	17S
NSPEC	I	PARAMETER		45	

1	VAR	R	3DIM	ARRAY	POINTIF	8P	6P	45
	NVAR							
	RDARS							
	UNIT							
	1							
	2							
	3							
	4							
	5							
	6							
	7							
	8							
	9							
	10							
	11							
	12							
	13							
	14							
	15							
	16							
	17							
	18							
	19							
	20							
	21							
	22							
	23							
	24							
	25							
	26							
	27							
	28							
	29							
	30							
	31							
	32							
	33							
	34							
	35							
	36							
	37							
	38							
	39							
	40							
	41							
	42							
	43							
	44							
	45							
	46							
	47							
	48							
	49							
	50							
	51							
	52							
	53							
	54							
	55							
	56							
	57							
	58							
	59							
	60							
	61							
	62							
	63							
	64							
	65							
	66							
	67							
	68							
	69							
	70							
	71							
	72							
	73							
	74							
	75							
	76							
	77							
	78							
	79							
	80							
	81							
	82							
	83							
	84							
	85							
	86							
	87							
	88							
	89							
	90							
	91							
	92							
	93							
	94							
	95							
	96							
	97							
	98							
	99							
	100							

TABLE OF EXTERNAL NAMES

\$STOP	EXTERNAL	37U	35U
\$WFF	EXTERNAL	36U	29U
\$WFI	EXTERNAL	36U	29U
\$WV	EXTERNAL	36U	29U
ASSIGN	EXTERNAL	18U	
CLOSE	EXTERNAL	25U	
DRBSF	EXTERNAL	28U	
FAMSTZ	EXTERNAL	20U	
FAMWAIT	EXTERNAL	19U	
RDARS	EXTERNAL	13U	
UNIT	EXTERNAL	32U	23U

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

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

TABLE OF PARAMETERS ENCOUNTERED

AKVAR	=	15750	NNX	=	72
NNV	=	33	NNZ	=	8H
NNY22D	=	3150	NPL	=	1
NSPEC	=	1	NVAR	=	5

BLOCK NAMES AND LENGTHS IN OCTAL

241-FILED	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		


```

1518 1. subroutine pttarr (i1,j1,k1,nloc)
1519 c
1520 c fills plot arrays rvx,rvy,rvz,erg,rho,rha,rhb
1521 c
1522 c array dimensions
1523 parameter (nny=72, nny2=nny*3, nnyz=88)
1524 parameter (nny2d=nny2*nnyz)
1525 c
1526 c hydro arrays
1527 parameter (nnspec=1, nvar=4*nnspec, npl=1)
1528 real var(nny2d,nvar,npl)
1529 common /hydvar/ var
1530 c
1531 c fixed arrays of hydrodynamic data
1532 parameter (nsum=4*nnspec)
1533 real ecamb(nnyz), pcamb(nnyz), ecamb(nnyz)
1534 real sumpl(nsum,nnyz)
1535 real rcmbl(nnyz), gravz(nnyz), gravz(nnyz), fsky(nnyz)
1536 c
1537 c common /hydro/ rcmbl, pcamb, ecamb, sumpl, rcmbl, gravz,
1538 c gravz, sumpl, shrink, fsky, ratio
1539 c
1540 c plot variables
1541 parameter (ndm1=72, ndm2=88)
1542 real rvx(ndm1,ndm2), rvy(ndm1,ndm2), rvz(ndm1,ndm2)
1543 real erg(ndm1,ndm2)
1544 real rho(ndm1,ndm2), rha(ndm1,ndm2), rhb(ndm1,ndm2)
1545 common /plivar/ rvx,rvy,rvz,erg,rho,rha,rhb
1546 c
1547 c local declarations
1548 logical lintp, colour, displa, darkhd
1549 real mplot(14)
1550 common /grfcom/ mslice, imin, imax, jmin, jmax, kmin, kmax,
1551 mplot, lintp, colour, dmin, dmax, displa, i1, j1, darkhd
1552 c
1553 c get data along the appropriate slice.
1554 go to (10,90,170), mslice
1555 c
1556 c X-Y slice.
1557 do 20 j=jmin,jmax
1558 rvx(i1,j)=var(j+jk,1,nloc)
1559 rvy(i1,j)=var(j+jk,2,nloc)
1560 rvz(i1,j)=var(j+jk,3,nloc)
1561 erg(i1,j)=var(j+jk,4,nloc)
1562 go to (70,50,30), nspec
1563 rhb(i1,j)=cvngt(var(j+jk,7,nloc),0.0,var(j+jk,7,nloc),gt)
1564 do 40 j=jmin,jmax
1565 f.e 6*var(j+jk,5,nloc))
1566 rha(i1,j)=cvngt(var(j+jk,6,nloc),0.0,var(j+jk,6,nloc),gt)
1567 do 80 j=jmin,jmax
1568 f.e 6*var(j+jk,5,nloc))
1569 rhb(i1,j)=var(j+jk,5,nloc)
1570 go to 290
1571 c
1572 c Y-Z slice
1573 do 100 k=kmin,kmax
1574 do 100 j=jmin,jmax

```

```

1574 39.          rvz(j,k) var(j)jk,1,nloc)
1575 40.          rvz(j,k)-var(j)jk,2,nloc)
1576 41.          rvz(j,k) var(j)jk,3,nloc)
1577 42.          rvz(j,k) var(j)jk,4,nloc)
1578 43.          go to (150,130,110), nspec
1579 44.          do 120 k=kmin,kmax
1580 45.          jk=kenny2+1
1581 46.          do 120 j=jmin,jmax
1582 47.          rhh(j,k)-cvmgf(var(j)jk,7,nloc),0.0,var(j)jk,7,nloc) qt
1583 48.          t e 6*var(j)jk,5,nloc))
1584 49.          do 140 k=kmin,kmax
1585 50.          jk=kenny2+1
1586 51.          do 140 j=jmin,jmax
1587 52.          rha(j,k)-cvmgf(var(j)jk,6,nloc),0.0,var(j)jk,6,nloc) qt
1588 53.          t e 6*var(j)jk,5,nloc))
1589 54.          rha(j,k)-cvmgf(var(j)jk,6,nloc),0.0,var(j)jk,6,nloc) qt
1590 55.          dmin.and.var(j)jk,6,nloc),le.dmax)
1591 56.          do 160 k=kmin,kmax
1592 57.          jk=kenny2+1
1593 58.          do 160 j=jmin,jmax
1594 59.          rho(j,k) var(j)jk,5,nloc)
1595 60.          go to 250
1596 61.          c
1597 62.          c X-Z slice.
1598 63.          jk=jj+1
1599 64.          do 180 k=kmin,kmax
1600 65.          jk1-jk+kenny2
1601 66.          rvz(j1,k1)-var(jk1,1,nloc)
1602 67.          rvz(j1,k1)-var(jk1,2,nloc)
1603 68.          rvz(j1,k1)-var(jk1,3,nloc)
1604 69.          erg(j1,k1)-var(jk1,4,nloc)
1605 70.          go to (230,210,190), nspec
1606 71.          do 200 k=kmin,kmax
1607 72.          rhh(j1,k1)-cvmgf(var(jk+kenny2,7,nloc),0.0,
1608 73.          var(jk+kenny2,7,nloc) qt,1.e-6*var(jk+kenny2,5,nloc))
1609 74.          do 220 k=kmin,kmax
1610 75.          rha(j1,k1)-cvmgf(var(jk+kenny2,6,nloc),0.0,
1611 76.          var(jk+kenny2,6,nloc) qt,1.e-6*var(jk+kenny2,5,nloc))
1612 77.          rha(j1,k1)-cvmgf(var(jk+kenny2,6,nloc),0.0,
1613 78.          var(jk+kenny2,6,nloc) qt,dmin.and.var(jk+kenny2,6,nloc))
1614 79.          t e.dmax)
1615 80.          do 240 k=kmin,kmax
1616 81.          rho(j1,k1)-var(jk+kenny2,5,nloc)
1617 82.          continue
1618 83.          return
1619 84.          end
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 23. P= 27b
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 29. P= 75c
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 31. P= 123b
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 33. P= 151a
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 38. P= 175c
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 46. P= 247c
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 50. P= 304c
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 54. P= 341c
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 65. P= 422d
PLIARR VECTOR LOOP BEGINS AT SEQ. NO. 67. P= 451c

```

P11ARR

PAGE 1

001 ACDFEILPORSTVY7

08/18/86 MK V 12-07 21

CF11146 (05/16/86) PAGE 67

P11ARR

VICTOR 1000 REGINS AT SEQ NO.

69, P- 50000

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

NUMBER USE	SOURCE PROGRAM REFERENCES
10 27b	22I
20 UNDEF**	23I 23E
20A43C	23I
20R70a	23I
30 75c	28J 29F
40 UNDEF**	29I
40A110b	29I
40R123b	29I
50 123b	31I
60 UNDEF**	32I
60A136a	31I
60R151a	31I
70 151a	33I
80 UNDEF**	34I
80A162c	33I
80R172a	33I
90 172c	36I
100 UNDEF**	42I 36F
100A175c	36I
100R237a	36I
100K211d	38I
100D234b	38I
110 244c	44I
120 UNDEF**	47I 44F
120A247c	44I
120R301c	44I
120C265a	46I
120D276d	46I
130 301c	48I
140 UNDEF**	51I 48F
140A304c	48I
140R336c	48I
140C322a	50I
140R333d	50I
150 336c	52I
160 UNDEF**	55I 52F
160A341c	52I
160R370b	52I
160C356c	54I
160D365c	54I
170 370d	57I
180 UNDEF**	67I 58F
180A403c	58I
180R415b	58I
190 422d	65I
200 UNDEF**	66I 64J 65F
200A435c	65I
200R451c	65I
210 451c	67I 64J
220 UNDEF**	68I 67F

220A464b 67I
 220R500b 67I
 230 500b 69I 64J
 240 UNDEF ** 69F
 240A511d 69I
 240R521d 69I
 250 521d 71I 35J
 00002 25d 21W
 00003 74a 28W
 00004 243a 43W
 00005 421b 64W

(SN-STATEMENT NUMBER, GSN-GENERATED STATEMENT NUMBER)
 (LN-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
45100 ERG	CVMGT	INLINE FUNCT.			
1 II	R	2DIM ARRAY	PLTVAR	68U	51U 32U 30U
	I	VARIABLE	DUM.ARG.	63S	27S 17D 15D
	I	VARIABLE		70U	63U 62U 61U
	I	VARIABLE		30U	26U 24U 10
	I	VARIABLE		55U/2	54I 47U/4 40U/2
	I	VARIABLE		39U/2	38I 34U/2 31I 40U/2
	I	VARIABLE		26U/2	25U/2 32U/4 29I 27U/2
	I	VARIABLE	DUM.ARG.	57U	1D 31I
	I	VARIABLE		70U	68U/3 55U 51U/3 43S
	I	VARIABLE		47U/3	42U 41U 40U 39U 32U/3
	I	VARIABLE		30U/3	27U 26U 25U 22S
	I	VARIABLE		63U	62U 61U 60U
	I	VARIABLE	GRFCOM	54N	50N 38N 31N
	I	VARIABLE	GRFCOM	54N	50N 38N 31N
	I	VARIABLE		70U/2	69I 46N 441 23N 200
	I	VARIABLE		60U	59U 58I 55U 52I 61U
	I	VARIABLE		47U	45U 44I 37U 36I
	I	VARIABLE	DUM.ARG.	22U	1D 40U
	I	VARIABLE	GRFCOM	64N	67N 52N 20D
	I	VARIABLE	GRFCOM	69N	67N 52N 20D
	I	VARIABLE		21P	20D 48N 48N
	I	VARIABLE	GRFCOM	16P/3	15P 14P/3 11S
	I	PARAMETER		16P/3	15P 14P/3 13S
	I	PARAMETER		70U	68U/3 63U 61U 51U/3
	I	VARIABLE	DUM.ARG.	47U/3	42U 41U 40U 27U
	I	VARIABLE		26U	25U 24U 10
	I	PARAMETER		10P	2S 43U
	I	PARAMETER		2S	68U/3 59U 22U
	I	PARAMETER		70U	68U/3 59U 57U 53U 22U
	I	PARAMETER		3S	3S 37U 37U 22U
	I	PARAMETER		2S	9P/3 3S 45U
	I	PARAMETER		11P/4	9P/3 3S 45U
	I	PARAMETER		6P	4S 55U
	I	PARAMETER		6P	5S 43P
	I	PARAMETER		64P	43P 28P 5S

NSUM	PARAMETER	TOP	85
15	PLTARR	6P	55
75700	R1A	6R5	170
112200	R1B	6R5	170
61400	R1R	705	170
0	R1X	605	170
14300	R1Y	615	170
30600	R1Z	625	170
0	R2	70U	66U
		55U	51U
		34U	32U
		7D	6U
			62U
			61U
			41U
			26U
			27U
			30P/2
			47P/2
			60U
			39U
			24U

TABLE OF EXTERNAL NAMES

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

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

NDM1	= 72
NNX	= 72
NNY2	= 35
NNZ2	= 90
NPL	= 1
NSUM	= 5
NDM2	= 88
NNY	= 33
NN7	= 88
NNZ2D	= 3150
NSPEC	= 1
NVAR	= 5

TABLE OF LOOPS ENCOUNTERED

LABEL INDEX	FROM	TO	ADDRESS	LENGTH
20 J	23	27	43C	25
40 J	21	30	110D	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	308C	33
140 J	50	51	322a	11
160 K	52	55	341C	27

160 J	54	55	350c	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-PLIARR 42 #18 36606-11YDVAR 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

```

1620 1
1621 1
1622 C      subroutine restr (lstep, ncore, restf, label)
1623 C      read data from dump file for graphics.
1624 C      array dimensions
1625 2      parameter (nmw=72, nny=31, nmz=88)
1626 3      parameter (nmxp=nmw*1, nnyx=nyy*1, nmzr=nmz*1)
1627 4      parameter (nmw2=nmw*2, nny2=nyy*2, nmz2=nmz*2)
1628 5      parameter (nny2d=nyy2*nmz2)
1629 C      hydro arrays
1630 6      parameter (nspec=1, nvar=4*nspec, npl=1)
1631 7      parameter (jkl=nmw*nvar*nyy2d, jkll=nmw2*nvar*nyy2d)
1632 8      real dvar(nmw2,nvar,npl)
1633 9      common /hydrovar/ dvar
1634 10     real var(nny2d,nvar,npl)
1635 11     pointer (ipvar,var)
1636 C      fixed arrays of hydrodynamic data
1637 12     parameter (nsum=4*nspec, ix=7*nmw2*nmw*nsum*3)
1638 13     real ramb(nmw2), ecamb(nmw2), ecamb(nmw2)
1639 14     real sumpl(nsum,nmx)
1640 15     real rcmn(nmw2), gravz(nmw2), gravz(nmw2), fsky(nmw2)
1641 16     common /hydrofix/ ramb, ecamb, sumpl, rcmn, gravz.
1642 17     gravz, sum4, shrink, fsky, ratio
1643 C      grid arrays
1644 18     parameter (ngrd=2*(nmw*nyy*nmz))
1645 19     real xcor(nmxp), ycor(nnyx), zcor(nmz)
1646 20     real twdx(nmxp), twdy2(nnyx), twdz2(nmw2)
1647 21     real dtodx(nmxp), dtody2(nnyx), dtodz2(nmw2)
1648 22     common /gridvar/ xcor, ycor, zcor, twdx, twdy2, twdz2, dtodx
1649 23     , dtody2, dtodz2
1650 C      global variables
1651 24     parameter (ntime=9)
1652 25     common /glover/ dtmin, dtmax, cour, time, dt, dth, rlos, nfile
1653 26     , nstep, llx, lly, llz
1654 C      body indices
1655 27     parameter (nrbd=1, nrbdx=1, nrbdz=1)
1656 28     parameter (nbody=1+2*nrbd*(1+nrbdx*(1+nrbdz)))
1657 29     integer kmdb(nrbd), kmdb(nrbd)
1658 30     integer kmdb(nrbdx,nrbd), kmdb(nrbdx,nrbd)
1659 31     integer jmbd(nrbdz,nrbdx,nrbd), jmbd(nrbdz,nrbdx,nrbd)
1660 32     common /bdcdb/ nrbd, imrbd, kmdb, kmdb, kmdb, jmbd, jmbd
1661 33     , station arrays
1662 34     parameter (mxx=1, nsta=1)
1663 35     parameter (nstat=1+4*mxx*nsta+mxx*3*nsta+1)
1664 36     logical lstat
1665 37     real rts(mxx,nsta), vls(mxx,nsta), gms(mxx,nsta)
1666 38     real prs(mxx,nsta), tme(mxx,nsta), xs(nsta), ys(nsta), zs(nsta)
1667 39     common /stats/ lstat, rts, vls, gms, prs, tme, xs, ys, zs, nxx
1668 40     , particles
1669 41     parameter (nnp=70, nburst=2)
1670 42     parameter (npa=1+3*nnp*nburst)
1671 43     real z(nnp,nburst), z(nnp,nburst)
1672 44     integer nnp(nburst), nchp(nnp,nburst)
1673 45     ,
1674 46     ,
1675 47     ,

```



```

1674 C common /part1/ xp,yp,opp,achk
1677 C
1678 C     active grid boundaries
1679 C
1679 C parameter (nscr=6, nact=17)
1680 C logical /scan, /sides, /top, /botm, /rear, /front
1681 C common /scan/ /scan, /sides, /top, /botm, /rear, /front
1682 C common /active/ /x, /y, /z, /xlo, /xhi, /ylo, /yhi, /zlo, /zhi,
1683 C /kz, /kzb, /kzo, /ks1, /ks2, /kdl, /kd2, /kthrm
1684 C     local declarations
1685 C parameter (nlab=5, nch=8-nlab)
1686 C character *8 restf
1687 C character *40 label
1688 C logical incore
1689 C
1690 C parameter (nsize=jkallntimnlabnschnactngndffixnbody
1691 C /nstatpart1)
1692 C data /unit /16/, /dump /17/
1693 C
1694 C set pointer
1695 C ipvar = loc(ivar)
1696 C
1697 C open dump file.
1698 C call filefo (restf, /unit, nsize)
1699 C
1700 C first read all the data stored at the end of the file. the
1701 C conserved variables on the mmx2 planes are stored at the
1702 C beginning
1703 C read file number, step, time, timestep and label.
1704 C call rdabef (/unit, time, ntime, /kall)
1705 C call rdabef (/unit, label, nlab, /kall, ntime)
1706 C call fileck
1707 C
1708 C write messages.
1709 C write (6, 30) nfile, nstep, time
1710 C /step, nstep
1711 C
1712 C error exits.
1713 C if (/l1) ne, mmx, or /l1y, ne, mmx, or /l1z, ne, mmz) then
1714 C write (6, 40) /l1x, /l1y, /l1z, mmx, mmx, mmz, mmz
1715 C stop
1716 C endif
1717 C
1718 C check if this is a time history run.
1719 C if (/lstat) then
1720 C /lsize, /kallntimnlabnschnactngndf
1721 C go to 30
1722 C endif
1723 C
1724 C read active grid switches and boundary indices.
1725 C call rdabef (/unit, /scan, /sco, /kallntimnlab)
1726 C call rdabef (/unit, /x, /nact, /kallntimnlabnsco)
1727 C
1728 C read coordinate data.
1729 C call rdabef (/unit, /x, /nact, /kallntimnlabnschnact)
1730 C /lsize, /kallntimnlabnschnactngndf
1731 C

```

```

1732 C read fixed arrays of hydrodynamic data.
1733 C call rdabst (lunit,rcamb,nfix,ladrs)
1734 C
1735 C read station arrays.
1736 C call rdabst (lunit,lstat,lstat,ladrs,nfix)
1737 C
1738 C read body indices.
1739 C call rdabst (lunit,nbod,nbody,ladrs,nfix,nstat)
1740 C
1741 C read particles
1742 C
1743 C call rdabst (lunit,xp,npart,l,ladrs,nfix,nstat,nbody)
1744 C
1745 C return
1746 C
1747 C only readin
1748 C
1749 C read in all the planes for incore calculation.
1750 C
1751 C call rdabst (lunit,var,jkpl,0)
1752 C call filefc
1753 C return
1754 C
1755 C
1756 C entry datain(lmin,imax,jj,kk)
1757 C
1758 C fill the plot arrays.
1759 C
1760 C do 20 i=lmin,imax
1761 C if (.not.incure) then
1762 C call fileck
1763 C call fileio (i,1)
1764 C call fileck
1765 C endif
1766 C iloc=comp(i,1,incure)
1767 C call plane (i,jj,kk,iloc)
1768 C continue
1769 C return
1770 C
1771 C 30 format (ix,'reading file',i3,' at time, steps',e12.5,16)
1772 C 40 format (ix,'error: wrong restart file',ix,'dump file contains n
1773 C imax,nny,nmiz',314/ix,'fast3d expects nmx,nny,nmiz',314/ix,'ex
1774 C 20 execution terminated')
1775 C 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
 20A312C 75L
 20R331C 75L
 30 FN 85L 52W
 40 FN 86L 55W
 OXXX3 174d 54I
 OXXX3 174d 58I
 OXXX4 320b 76I

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

TABLE OF NAME'S ENCOUNTERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS NAME TYPE MAIN USAGE BLOCK SOURCE PROGRAM REFERENCES

\$STOP	EXTERNAL			56U	
\$WFF	EXTERNAL			55U	
\$WFI	EXTERNAL			52U	52U
\$WV	EXTERNAL			55U/6	52U/3
CVMG	INLINE FUNCT.			81U	
271 DATALN	ENTRY			74D/2	
O DVAR	R 3DIM ARRAY	HYDVAR		47P	9D
FILECK	EXTERNAL			79U	77U
FILEFC	EXTERNAL			72U	
FILEFO	EXTERNAL			48U	
FILEID	EXTERNAL			78U	
60 IADRS	I VARIABLE			68P	66P
61 I1	I VARIABLE			82P	65S
62 I1OC	I VARIABLE			82P	75I
6 IMAX	I VARIABLE			75N	
5 IMIN	I VARIABLE			74D	
2 INCOR1	I VARIABLE			75N	
55 IPVAR	*I VARIABLE			81P	44D
1 ISTEP	*I VARIABLE			475	11P
O IXR	I VARIABLE			535	1D
7 JJ	I VARIABLE			63P	40D
JKALL	I PARAMETER			82P	74D
JKPL	I PARAMETER			65U	64P
10 KK	I VARIABLE			71P	75
4 LABEL	CH VARIABLE			82P	1D
57 LDUMP	*I VARIABLE			50P	43D
11 L1LX	I VARIABLE			465	
12 L1LY	I VARIABLE			55U	54U
13 L1LZ	I VARIABLE			55U	54U
LOC	I INLINE FUNCT			55U	54U
O I SCAN	I VARIABLE	SCAN		47U	
O I STAT	L VARIABLE	STATS		62P	38D
				67P	31D
				63P	49P
				59U	50P
				62P	49P
				63P	49P
				64P	75

56	UNIT	I	VARIABLE	71P	68P	67P	60P	64P	63P	62P	50P	49P
	MAIN.			48P	46S							
	ENTRY											
	P'PARAMETER			30P/2	29P/3	26S						
	P'PARAMETER			65U	64U	63P	59U	37S				
	P'PARAMETER			24S	25S							
	P'PARAMETER			24S								
	P'PARAMETER			35P/2	34P/2	32S						
	P'PARAMETER			41S								
	VARIABLE		GLOVAR	52U	23D							
	P'PARAMETER			67U	66P	66P	12S					
	P'PARAMETER			65U	64P	59U	17S					
	P'PARAMETER			65U	64U	63U	62U	59U	50P	41S		
	P'PARAMETER			24S								
	P'PARAMETER			55U	54U	14P	2S					
	P'PARAMETER			4S								
	P'PARAMETER			20P	19P	18P	3S					
	P'PARAMETER			55U	54U	18P	2S					
	P'PARAMETER			20P	19P	4S						
	P'PARAMETER			3S								
	P'PARAMETER			55U	54U	13P	2S					
	P'PARAMETER			20P	19P	15P/4	13P/3	4S				
	P'PARAMETER			3S								
	P'PARAMETER			10P	RP	5S						
	P'PARAMETER			35P	34P/2	32S						
	P'PARAMETER			68P	33S							
	P'PARAMETER			10P	8P	6S						
	P'PARAMETER			65U	64U	63U	62P	59U	37S			
	P'PARAMETER			48P	45S							
	P'PARAMETER			6S								
	P'PARAMETER			30P/4	29P/3	26S						
	P'PARAMETER			68U	67P	27S						
	P'PARAMETER			53U	52U	23D						
	P'PARAMETER			14P	12S							
	P'PARAMETER			65U	64U	63U	62U	59U	50U	49P	22S	
	P'PARAMETER			10P	8P	6S						
	EXTERNAL			82U								
	EXTERNAL			66P	16D	13D	66U	64U	63U	62U	50U	49U
	EXTERNAL			71U	68U	67U						
	EXTERNAL			70P/2								
	EXTERNAL			48P	42U	1D						
	EXTERNAL			74U	70U	1D/2						
	EXTERNAL			52U	49P	23D						
	EXTERNAL			71U	11P	10D						
	EXTERNAL			64P	21D	18D						
	EXTERNAL			68P	36D	34D						
	EXTERNAL			56U	52U							
	EXTERNAL			55U	52U							
	EXTERNAL			55U/6	52U/3							
	EXTERNAL			73U	77U	51U						

TABLE OF EXTERNAL NAMES

\$STOP	EXTERNAL
\$WFF	EXTERNAL
\$WFI	EXTERNAL
\$WV	EXTERNAL
FILECK	EXTERNAL

EXTERNAL 72U
 EXTERNAL 48U
 EXTERNAL 78U
 EXTERNAL 82U
 EXTERNAL 71U
 PDABST 68U 67U 66U 64U 49U
 63U 62U 50U

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 ? IFN OR MORE REFERENCES TO SYMBOL

TABLE OF PARAMETERS ENCOUNTERED

JKALL	= 1165500	JKPL	= 1134000
MAX	= 1	NACT	= 17
NRDIX	= 1	NBODY	= 7
NRDZ	= 1	NBURST	= 2
PH	= 40	NFIX	= 993
RIGPD	= 392	NIAB	= 5
TNRD	= 1	NPJX	= 72
NNX2	= 74	NNXP	= 73
NNY	= 33	NNY2	= 35
NNYP	= 34	NNZ	= 88
NNZ2	= 90	NNZP	= 89
NNZ2D	= 3150	NOP	= 70
NPARTL	= 422	NPL	= 1
NSCN	= 6	NSIZE	= 1167361
NSPTC	= 1	NSJA	= 1
NSSTAT	= 10	NSUM	= 5
NSTM	= 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 RESTR	24-#1R		141 #C1	
14-GLOVAR	12 #1A15		646 PARTII	
STATIC SPACE (IN OCTAL)				
B SAVE			10	
I SAVE			4	
CONSTANTS			53	

1741-HYDFIX
 21-ACTIVE
 1116-GRDVAR

36606-HVDVAR
 6-SCAN

VARIABLES:	
TEMPORARIES:	10
CODE	151
	252
TOTAL:	522

```

1776 C      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 C      parameter (ndm1=72, ndm2=88, nr=5)
1783 C      real arr(ndm1,ndm2), hue(nr)
1784 C      common /holder/ space(ndm1,ndm2),valmin,valuig
1785 C      logical displa, darkhd
1786 C      common /grfcom/ mslice, iamn, imax, jamin, jmax, kmin, kmax,
1787 C      mplot, linterp, colour, dmin, dmax, displa, il, jl, darkhd
1788 C      data hue /3hred,4hblue,5hgreen,6hyellow,5hclear/
1789 C
1790 C      determine maximum and minimum values.
1791 C      arrmax=0.0
1792 C      arrmin=1.e50
1793 C      do 10 j=jmin,jmin+kv-1
1794 C          imx=imax+kh,arrrmin,j,1)imin-1
1795 C          arrmax=arr(imx,j)
1796 C          imj=jmin+kh,arrrmin,j,1)imin-1
1797 C          arrmin=arr(imj,j)
1798 C      continue
1799 C      drange=(arrmax-arrmin)/float(nr)
1800 C
1801 C      plot contours for each range.
1802 C      h55PLA plots the colour contours in the subroutine discol. NCAR
1803 C      draws the contour lines a few at a time, changing color between.
1804 C      if (displa) then
1805 C          valmin=arrmin
1806 C          valrig=drange
1807 C          call setc.ir ('RED')
1808 C          call discol (arr(imin,jmin),ndm1,space,kh,kv)
1809 C      else
1810 C          do 30 ir=1,nr
1811 C              flo=arrmin+float(ir-1)*drange
1812 C              fhj=arrmin+float(ir)*drange
1813 C              call flush
1814 C              call gcolor (hue(ir))
1815 C              call conrec (arr(imin,jmin),ndm1,kh,kv,flo,fhj,6,-1,0,0)
1816 C              call flush
1817 C          continue
1818 C      endif
1819 C      return
1820 C      end

```

O AT SEQUENCE NUMBER 11. DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 12
PRNAME COLCON COMMENT - ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION P=0250630D
EXPLANATION: ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION

O AT SEQUENCE NUMBER 11. DEPENDENCY INVOLVING ARRAY "ARR" IN SEQUENCE NUMBER 13
PRNAME COLCON COMMENT - ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION P=0250630D
EXPLANATION: ARRAY USED AS AN ARGUMENT TO A SUBROUTINE/FUNCTION

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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF** 151 10E
 10A50c 101
 10B77a 101
 30 UNDEF** 301 23L
 30A132d 23L
 30BUNDEF** 23L
 0XXX2 121c 17L
 0XXX3 161a 22L

(S)=STATEMENT NUMBER, G=N GENERATED STATEMENT NUMBER)
 (L)=FORMAT NUMBER, UNDEF=UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCIPHERED (ADDRESS FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS NAME TYPE MAIN USAGE BLOCK SOURCE PROGRAM REFERENCES

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 10
17 ARRMAX	R	VARIABLE			16U 85
20 ARRMIN	R	VARIABLE			24U 14P 14S 95
31 COLCON	R	ENTRY			10/2
CONREC	R	EXTERNAL			28U
DISCDL	R	EXTERNAL			21U
14 DISPLA	L	VARIABLE	GRFCOM		6D 5D 19U 165
24 DRANGE	R	VARIABLE			24U 245
27 FHI	R	VARIABLE			28P 245
26 FLO	R	VARIABLE			24U 16U
FLOAT	R	INLINE FUNCT.			24U 26U
FLUSH	R	EXTERNAL			29U
GCOLOR	R	EXTERNAL			27U
12 HUE	R	1DIM ARRAY			7S 3D 13U/2 11U/2 10
2 IMIN	I	VARIABLE	DUM.ARG.		21U 135
23 IMN	I	VARIABLE			14U 11S
22 IMX	I	VARIABLE			12U 25P 231
25 IR	I	VARIABLE			21U 11U
ISMAY	I	EXTERNAL			13U
ISMIN	I	EXTERNAL			14U
21 J	I	VARIABLE			10N/2 101
3 JMIN	I	VARIABLE	DUM.ARG.		28U 10N/2 10
4 K1	I	VARIABLE	DUM.ARG.		28P 11P 10
5 KV	I	VARIABLE	DUM.ARG.		28P 10N 10
MAIN.	R	ENTRY			4P 3P 25
NDM1	I	PARAMETER			21P 3P 25
NDM2	I	PARAMETER			4P 3P 25
NR	I	PARAMETER			3P 25
SETCLR	R	EXTERNAL			20U
SPACE	R	2DIM ARRAY	HOLDER		4D
14300 VALMIN	R	VARIABLE	HOLDER		4D

1430J VALRNS *R VARIABLE HOLDER 19S 4D

TABLE OF EXTERNAL NAMES

CONTR:	EXTERNAL	28U
DISCO:	EXTERNAL	21U
FLUSH:	EXTERNAL	24U
GCOLOR:	EXTERNAL	27U
ISMAX:	I EXTERNAL	11U
ISMIN:	I EXTERNAL	13U
SETCLR:	EXTERNAL	20U

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

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

TABLE OF PARAMETERS ENCOUNTERED

NDM1 = 72 NDM2 = 88
 NR = 5

TABLE OF LOOPS ENCOUNTERED

LABL INDEX	FROM	TO	ADDRESS	LENGTH
10 J	10	15	50C	27
30 IR	23	30	132B	27

BLOCK NAMES AND LENGTHS IN OCTAL

16A-COLCON 27-#IR 37 #CI 1430J-HOLDER 20-GRFCOM

STATIC SPACE (IN OCTAL)

B SAVE	23
I SAVE	4
CONSTANTS	10
VARIABLES	20
TEMPORARIES	37
CODE	134
TOTAL	252

```

1821      subroutine discol (arrin,mdim,cmhold,kh,kv)
1822      real arrin(mdim,kv),cmhold(kh,kv)
1823      common work(5000)
1824
1825      .....
1826      .....
1827      * This subroutine uses DISPLA graphics to draw contour
1828      * plots in color. The actual color calls are in the
1829      * user-defined subroutine MYCON.
1830      .....
1831      .....
1832      C Store the portion of the array to be plotted
1833      do 10 i=1,kh
1834      do 10 j=1,kv
1835      cmhold(i,j)=arrin(i,j)
1836      continue
1837      C Set up the contour lines
1838      call bcmon(5000)
1839      call comak (cmhold,kh,kv,'scale')
1840      * specify the characteristics of the lines and labels, including their
1841      * color (color) and other characteristics is set in the routine mycon.
1842      call conin (0,'MYCON','LABELS',1,3)
1843      call conang (60.)
1844      call conth(0.06)
1845      draw the contour lines
1846      call contour (1,'labels','draw')
1847      return
1848      end
1849
1850      VVECTOR LOOP BEGINS AT SEQ. NO. 5. P= 364
1851
DISCOL

```



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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF**	71	5F	4F
10A36d	41		
10B63C	41		
10C52D	5L		
10D60d	5L		

(LSN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (LN=FORMAT NUMBER, UNDLN=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.	CU	2U	1D
RCOMON		EXTERNAL		8U		
3 CNHDL	R	2DIM ARRAY	DUM.ARG.	9P	6S	2P 1D
CONANG		EXTERNAL		11U		
CONLIN		EXTERNAL		10U		
CONMAK		EXTERNAL		9U		
CONTHN		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

RCOMON	EXTERNAL	8U
CONANG	EXTERNAL	11U
CONLIN	EXTERNAL	10U
CONMAK	EXTERNAL	9U
CONTHN	EXTERNAL	12U
CONTUR	EXTERNAL	13U

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

- A USED IN FORTRAN ASSIGN STATEMENT
- D DEFINED IN DECLARATIVE STATEMENT
- E STATEMENT NUMBER ENDING A DO LOOP
- I INDEX OF A DO OR IMPLIED DO LOOP
- J STATEMENT NUMBER USED IN TRANSFER
- L SOURCE LINE OF A STATEMENT NUMBER
- N NAME USED AS A DO LOOP PARAMETER
- P USED IN CALL/FUNC CALL OR ARRAY DEF
- R FORMAT USED IN A READ STATEMENT
- S STORED SO CONTENTS MAY BE CHANGED
- U NAME USED IN EXECUTABLE STATEMENT
- W FORMAT USED IN A WRITE STATEMENT
- X DEFINED OR DECLARED BUT NOT USED
- ? 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	52B	6

BLOCK NAMES AND LENGTHS IN OCTAL

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

STATIC SPACE (IN OCTAL)

R SAVE	27
T SAVE	2
CONSTANTS	20
VARIABLES	4
TEMPORARIES	25
CODE	75

TOTAL : 177

```

1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870

```

1. subroutine mycon (array, iarray)
2. parameter (ndim1=72, ndim2=88, nr=5)
3. real array(2),hue(nr)
4. integer iarray(9)
5. common /holder/ space(ndim1,ndim2),valmin,valorng,
6. space2(ndim1,ndim2)
7. data hue /3hred,4hblue,5hgreen,6hyellow,5hwhite/
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
18.
19.
20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.
31.
32.
33.
34.
35.
36.
37.
38.
39.
40.
41.
42.
43.
44.
45.
46.
47.
48.
49.
50.
51.
52.
53.
54.
55.
56.
57.
58.
59.
60.
61.
62.
63.
64.
65.
66.
67.
68.
69.
70.
71.
72.
73.
74.
75.
76.
77.
78.
79.
80.
81.
82.
83.
84.
85.
86.
87.
88.
89.
90.
91.
92.
93.
94.
95.
96.
97.
98.
99.
100.
101.
102.
103.
104.
105.
106.
107.
108.
109.
110.
111.
112.
113.
114.
115.
116.
117.
118.
119.
120.
121.
122.
123.
124.
125.
126.
127.
128.
129.
130.
131.
132.
133.
134.
135.
136.
137.
138.
139.
140.
141.
142.
143.
144.
145.
146.
147.
148.
149.
150.
151.
152.
153.
154.
155.
156.
157.
158.
159.
160.
161.
162.
163.
164.
165.
166.
167.
168.
169.
170.
171.
172.
173.
174.
175.
176.
177.
178.
179.
180.
181.
182.
183.
184.
185.
186.
187.
188.
189.
190.
191.
192.
193.
194.
195.
196.
197.
198.
199.
200.
201.
202.
203.
204.
205.
206.
207.
208.
209.
210.
211.
212.
213.
214.
215.
216.
217.
218.
219.
220.
221.
222.
223.
224.
225.
226.
227.
228.
229.
230.
231.
232.
233.
234.
235.
236.
237.
238.
239.
240.
241.
242.
243.
244.
245.
246.
247.
248.
249.
250.
251.
252.
253.
254.
255.
256.
257.
258.
259.
260.
261.
262.
263.
264.
265.
266.
267.
268.
269.
270.
271.
272.
273.
274.
275.
276.
277.
278.
279.
280.
281.
282.
283.
284.
285.
286.
287.
288.
289.
290.
291.
292.
293.
294.
295.
296.
297.
298.
299.
300.
301.
302.
303.
304.
305.
306.
307.
308.
309.
310.
311.
312.
313.
314.
315.
316.
317.
318.
319.
320.
321.
322.
323.
324.
325.
326.
327.
328.
329.
330.
331.
332.
333.
334.
335.
336.
337.
338.
339.
340.
341.
342.
343.
344.
345.
346.
347.
348.
349.
350.
351.
352.
353.
354.
355.
356.
357.
358.
359.
360.
361.
362.
363.
364.
365.
366.
367.
368.
369.
370.
371.
372.
373.
374.
375.
376.
377.
378.
379.
380.
381.
382.
383.
384.
385.
386.
387.
388.
389.
390.
391.
392.
393.
394.
395.
396.
397.
398.
399.
400.
401.
402.
403.
404.
405.
406.
407.
408.
409.
410.
411.
412.
413.
414.
415.
416.
417.
418.
419.
420.
421.
422.
423.
424.
425.
426.
427.
428.
429.
430.
431.
432.
433.
434.
435.
436.
437.
438.
439.
440.
441.
442.
443.
444.
445.
446.
447.
448.
449.
450.
451.
452.
453.
454.
455.
456.
457.
458.
459.
460.
461.
462.
463.
464.
465.
466.
467.
468.
469.
470.
471.
472.
473.
474.
475.
476.
477.
478.
479.
480.
481.
482.
483.
484.
485.
486.
487.
488.
489.
490.
491.
492.
493.
494.
495.
496.
497.
498.
499.
500.
501.
502.
503.
504.
505.
506.
507.
508.
509.
510.
511.
512.
513.
514.
515.
516.
517.
518.
519.
520.
521.
522.
523.
524.
525.
526.
527.
528.
529.
530.
531.
532.
533.
534.
535.
536.
537.
538.
539.
540.
541.
542.
543.
544.
545.
546.
547.
548.
549.
550.
551.
552.
553.
554.
555.
556.
557.
558.
559.
560.
561.
562.
563.
564.
565.
566.
567.
568.
569.
570.
571.
572.
573.
574.
575.
576.
577.
578.
579.
580.
581.
582.
583.
584.
585.
586.
587.
588.
589.
590.
591.
592.
593.
594.
595.
596.
597.
598.
599.
600.
601.
602.
603.
604.
605.
606.
607.
608.
609.
610.
611.
612.
613.
614.
615.
616.
617.
618.
619.
620.
621.
622.
623.
624.
625.
626.
627.
628.
629.
630.
631.
632.
633.
634.
635.
636.
637.
638.
639.
640.
641.
642.
643.
644.
645.
646.
647.
648.
649.
650.
651.
652.
653.
654.
655.
656.
657.
658.
659.
660.
661.
662.
663.
664.
665.
666.
667.
668.
669.
670.
671.
672.
673.
674.
675.
676.
677.
678.
679.
680.
681.
682.
683.
684.
685.
686.
687.
688.
689.
690.
691.
692.
693.
694.
695.
696.
697.
698.
699.
700.
701.
702.
703.
704.
705.
706.
707.
708.
709.
710.
711.
712.
713.
714.
715.
716.
717.
718.
719.
720.
721.
722.
723.
724.
725.
726.
727.
728.
729.
730.
731.
732.
733.
734.
735.
736.
737.
738.
739.
740.
741.
742.
743.
744.
745.
746.
747.
748.
749.
750.
751.
752.
753.
754.
755.
756.
757.
758.
759.
760.
761.
762.
763.
764.
765.
766.
767.
768.
769.
770.
771.
772.
773.
774.
775.
776.
777.
778.
779.
780.
781.
782.
783.
784.
785.
786.
787.
788.
789.
790.
791.
792.
793.
794.
795.
796.
797.
798.
799.
800.
801.
802.
803.
804.
805.
806.
807.
808.
809.
810.
811.
812.
813.
814.
815.
816.
817.
818.
819.
820.
821.
822.
823.
824.
825.
826.
827.
828.
829.
830.
831.
832.
833.
834.
835.
836.
837.
838.
839.
840.
841.
842.
843.
844.
845.
846.
847.
848.
849.
850.
851.
852.
853.
854.
855.
856.
857.
858.
859.
860.
861.
862.
863.
864.
865.
866.
867.
868.
869.
870.
871.
872.
873.
874.
875.
876.
877.
878.
879.
880.
881.
882.
883.
884.
885.
886.
887.
888.
889.
890.
891.
892.
893.
894.
895.
896.
897.
898.
899.
900.
901.
902.
903.
904.
905.
906.
907.
908.
909.
910.
911.
912.
913.
914.
915.
916.
917.
918.
919.
920.
921.
922.
923.
924.
925.
926.
927.
928.
929.
930.
931.
932.
933.
934.
935.
936.
937.
938.
939.
940.
941.
942.
943.
944.
945.
946.
947.
948.
949.
950.
951.
952.
953.
954.
955.
956.
957.
958.
959.
960.
961.
962.
963.
964.
965.
966.
967.
968.
969.
970.
971.
972.
973.
974.
975.
976.
977.
978.
979.
980.
981.
982.
983.
984.
985.
986.
987.
988.
989.
990.
991.
992.
993.
994.
995.
996.
997.
998.
999.
1000.

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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF** 91 7E
 10A230 71
 10BUNDEF** 71
 00002 33b 8L
 (LN-STATEMENT NUMBER, GEN-GENERATED STATEMENT NUMBER)
 (LN-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

ADDRESS NAME	TYPE	MAIN USAGE	BLOCK	SOURCE PROGRAM REFERENCES
3 FLOAT	R	INLINE FUNCT.		
3 HUE	R	1DIM ARRAY		BU/2 3D
10 I	I	VARIABLE		65 71
12 MYCUN	I	ENTRY		10/2
NUM1	I	PARAMETER		5P/2 25
NUM2	I	PARAMETER		5P/2 25
NR	I	PARAMETER		8P/2 7N 3P 10
1 RARRAY	R	1DIM ARRAY	DUM.ARG.	
SETCLR	R	EXTERNAL		BU
14300 VALMIN	R	VARIABLE	HOLDER	BU/2 5D
14301 VALRN3	R	VARIABLE	HOLDER	BU/2 5D

TABLE OF EXTERNAL NAMES

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

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

TABLE OF PARAMETERS ENCOUNTERED

NUM1 = 72
 NR = 5
 NUM2 = 88

TABLE OF LOGS ENCOUNTED

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

BLOCK NAMES AND LENGTHS IN OCTAL

41 MYCON 15 #1B 2 #11 30602 HOLDER

STATIC SPACE (IN OCTAL)

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

TOTAL 60

```

1871      subroutine adjust (tim,array,n,tmax,length)
1872
1873
1874      c  chops away portions of time history that are not of interest.
1875
1876      real t(min), array(n)
1877
1878      length=n
1879      tmax=tim(n)
1880      amax=array(ismax(n,array,1))
1881      amin=array(1)+(amax-array(1))*0.1
1882
1883      c  chop to the right.
1884      imax=n
1885      do 10 i=n,1,-1
1886      if (array(i).gt.amin) go to 20
1887      continue
1888  10   go to 30
1889      imax=i
1890      tmax=tim(imax)
1891
1892      c  chop to the left.
1893      imin=1
1894      do 40 i=1,imax
1895      if (array(i).gt.amin) go to 50
1896      continue
1897  40   go to 60
1898      imin=i
1899      length=imax-imin+1
1900      do 70 i=imin,imax
1901      tim(i-imin)=tim(i)
1902      array(i-imin+1)=array(i)
1903      return
1904  70   end

```

0 AT SEQUENCE NUMBER 22. DEPENDENCY INVOLVING ARRAY "TIM" P-0250024C
PRNAME ADJUST COMMENT - DEPENDENCY INVOLVING ARRAY "TIM" P-0250024C
EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
0 AT SEQUENCE NUMBER 23. DEPENDENCY INVOLVING ARRAY "ARRAY" P-0250024C
PRNAME ADJUST COMMENT - DEPENDENCY INVOLVING ARRAY "ARRAY" P-0250024C
EXPLANATION: AMBIGUOUS OR CONFLICTING SUBSCRIPTS
ADJUST VECTOR LOOP BEGINS AT SEQ. NO. 21, P= 65rd
ADJUST CONDITIONAL VECTOR LOOP BEGINS AT SEQ. NO. 21, P= 65rd

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

NUMBER USE SOURCE PROGRAM REFERENCES

10 UNDEF** 10I 8F
 10A42C 8I
 10R47C 8I 9J
 20 50R 12I 11J
 30 51R 13I 15E
 40 UNDEF** 17L
 40A57b 15I
 40R64b 15I 16J
 50 64d 19I 18J
 60 65d 20I 21I
 70 UNDEF** 23I
 70A105d 21I
 70B133C 21I
 0XXX02 116a 23I
 0XXX03 133C 23E
 0XXX04 UNDEF** 23F
 0XXX05 124d 23E

(SN=STATEMENT NUMBER, GEN-GENERATED STATEMENT NUMBER)
 (LN=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			10/2	
5 AMAX	R VARIABLE			6U	55
6 AMIN	R VARIABLE			16U	9U
2 ARRAY	R 1DIM ARRAY	DUM.ARG.		23U	65
10 I	I VARIABLE			20U/2	16U
7 IMAX	I VARIABLE			21N	21I
11 IMIN	I VARIABLE			20U	15N
5 ISMAX	I EXTERNAL			22U	21N
5 LENGTH	*I VARIABLE	DUM.ARG.		23U	20U
3 N	I VARIABLE			5U	10
1 TIM	R 1DIM ARRAY	DUM.ARG.		20S	35
4 TMAX	*R VARIABLE	DUM.ARG.		7U	5P
				22U	13U
				13S	45
					1D
					2P/2
					1D
					3U
					2D
					5U
					12U
					2P
					9U
					8I

TABLE OF EXTERNAL NAMES

ISMAX I EXTERNAL 5U

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

- A USED IN FORTRAN ASSIGN STATEMENT
- D DEFINED IN DECLARATIVE STATEMENT
- F STATEMENT NUMBER ENDING A DO LOOP
- I INDEX OF A DO OR IMPLIED DO LOOP
- J STATEMENT NUMBER USED IN TRANSFER
- L SOURCE LINE OF A STATEMENT NUMBER
- N NAME USED AS A DO LOOP PARAMETER
- P USED IN CALL/FUNC CALL OR ARRAY DEF
- R FORMAT USED IN A READ STATEMENT
- S STORED SO CONTENTS MAY BE CHANGED
- U NAME USED IN EXECUTABLE STATEMENT
- W FORMAT USED IN A WRITE STATEMENT
- X DEFINED OR DECLARED BUT NOT USED
- ? 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 #18 4 -#C1

STATIC SPACE (IN OCTAL)

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

```

1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960

```

```

      subroutine count (arrin,xin,yin,nx,ix,iv,arout,mx,my)
c
c this subroutine interpolates values from an irregularly spaced
c mesh to an evenly spaced mesh in two dimensions.
c it is meant to be an interface to the NCAR contour plotter
c CONREC which works only on regularly spaced grids.
c
c parameter (msxy=90)
c real arrin(mx,my), arout(mx,my)
c real xin(ix), yin(my)
c common /bound/ xmin, xmax, ymin, ymax
c
c integer iin(mx), xfacp(mx)
c real xfacm(mx), xfacp(mx)
c
c dy=(ymax-ymin)/float(my-1)
c dz=(xmax-xmin)/float(mx-1)
c myt=my-1
c ixt=ix-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
c iimin=1
c do 30 iout=1,mx
c   xxmin(iout)=xin(i)
c   if (iout.gt.1) iimin=iin(iout-1)
c   do 10 j=imin,ixm1
c     if (x.gt.xin(j)) go to 10
c     iin(iout)=max0(1,j)
c     go to 20
c   continue
c   yin(iout)=1*xm1
c   i=ixm1(iout)
c   xfacm(iout)=(x-xin(i))/x(in(iout-1))-xin(i))
c   xfacp(iout)=1.0-xfacm(iout)
c   continue
c 30
c
c cycle over the y values
c
c jjin=1
c do 70 jout=1,my
c   y-ymin(jout-1)*dy
c
c find the interval in the yin array in which y resides
c
c jimin=1
c do 40 j=jjin,myml
c   if (y.gt.yin(j)) go to 40
c   jjin=max0(1,j-1)
c   go to 50
c 40
c continue
c jjin=myml
c continue
c
c compute scale factors for y direction
c

```

```

1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
CURINI

```

```

C
37. yfacm (v-y*in(jin))/(y*in(jin)+1) yin(jin)
38. yfacp=10-yfacm
C
C cycle over the x values
C
39. do 60 iout=1,mx
40.   in=iin(iout)
41.   arout(iout,iout)=yfacm*xfacm(iout)+arr*in(iout,iout)
1970   yfacm*xfacp(iout)+arr*in(iout,iout)+yfacp*xfacm(iout)+arr*in
1971   (iout,iout)+yfacp*xfacp(iout)+arr*in(iout,iout)
42.   60 continue
43.   70 continue
C
44.   return
45.   end
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	
10A	502d	16I			
10R	513a	16A			
20	514c	22I	19J		
30	UNDEF**	25I	13E		
30A	470b	13I			
30R	525c	13I			
40	546c	34I	31J	30E	
40A	541b	30I			
40B	551b	30I			
50	552b	36I	33J		
60	UNDEF**	42I	39E		
60A	573d	39I			
60R	636a	39I			
70	UNDEF**	43I	27E		
70A	531a	27I			
70B	640d	27I			
00002	476b	15I			
00003	606a	42E			
00004	605c	42E			
00005	616a	42E			
00006	615c	42I			
00007	622a	42E			
00008	621c	42E			
00009	626c	42I			
00010	626a	42E			

(SN=STATEMENT NUMBER, GEN=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	ARROUJ	R	2DIM ARRAY	DUM ARG. 41S 3P 1D
442	CONINT	R	ENTRY	1D/2 9S
422	DX	R	VARIABLE	28U 8S
421	DY	R	VARIABLE	9U 8U
	FLOAT	R	INLINE FUNCT	18U 17U 16I
430	I	I	VARIABLE	40U 22U 21S 18S 15U 6D
3	IIN	I	1DIM ARRAY	16N 15S 12S
425	JMIN	I	VARIABLE	41U/4 40S 23U/3 22S
431	JN	I	VARIABLE	41U/5 40U 39I 24U/2 23U 18U 15U/2
426	LOUT	I	VARIABLE	14U 13I
436	J	I	VARIABLE	32P 31U 30I
432	JIN	I	VARIABLE	41U/4 37U/3 35S 29U 26S
435	JMIN	I	VARIABLE	30N 29S

LINE	TEXT	41U	28U	27I
433	JOUT			
5	LX			
424	LXMI			
	MAIN			
	MAXO			
10	MX	41U	28U	27I
	MXV	11U	4P	1D
11	MY	21U	16N	115
6	NY			
423	NYM1			
427	X			
135	XFALM			
267	XFACP			
2	XIN			
1	XMAX			
0	XMIN			
434	Y			
437	YFACM			
440	YFACP			
3	YIN			
2	YMAX			
2	YMIN			
	Z			
	ZM			
	ZM1			
	ZM2			
	ZM3			
	ZM4			
	ZM5			
	ZM6			
	ZM7			
	ZM8			
	ZM9			
	ZM10			
	ZM11			
	ZM12			
	ZM13			
	ZM14			
	ZM15			
	ZM16			
	ZM17			
	ZM18			
	ZM19			
	ZM20			
	ZM21			
	ZM22			
	ZM23			
	ZM24			
	ZM25			
	ZM26			
	ZM27			
	ZM28			
	ZM29			
	ZM30			
	ZM31			
	ZM32			
	ZM33			
	ZM34			
	ZM35			
	ZM36			
	ZM37			
	ZM38			
	ZM39			
	ZM40			
	ZM41			
	ZM42			
	ZM43			
	ZM44			
	ZM45			
	ZM46			
	ZM47			
	ZM48			
	ZM49			
	ZM50			
	ZM51			
	ZM52			
	ZM53			
	ZM54			
	ZM55			
	ZM56			
	ZM57			
	ZM58			
	ZM59			
	ZM60			
	ZM61			
	ZM62			
	ZM63			
	ZM64			
	ZM65			
	ZM66			
	ZM67			
	ZM68			
	ZM69			
	ZM70			
	ZM71			
	ZM72			
	ZM73			
	ZM74			
	ZM75			
	ZM76			
	ZM77			
	ZM78			
	ZM79			
	ZM80			
	ZM81			
	ZM82			
	ZM83			
	ZM84			
	ZM85			
	ZM86			
	ZM87			
	ZM88			
	ZM89			
	ZM90			
	ZM91			
	ZM92			
	ZM93			
	ZM94			
	ZM95			
	ZM96			
	ZM97			
	ZM98			
	ZM99			
	ZM100			

TABLE OF EXTERNAL NAMES

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

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

MXV = 90

TABLE OF LOOPS ENCOUNTERED

LINE INDEX	FROM	TO	ADDRESS	LENGTH
30	13	25	470B	36
10	16	20	5020	11
70	27	43	531A	110
40	30	34	541B	10
60	39	42	573C	43

BLOCK NAMES AND LENGTHS IN OCTAL

45 #IR 0 #CI 4 ROUND

644-CONINT

STATIC SPACE (IN OCTAL)	
B SAVE:	40
T SAVE:	5
CONSTANTS:	1
VARIABLES:	440
TEMPORARIES:	0
CODE:	203
TOTAL:	711

```

1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998

1. C      subroutine wsize (xmin,xmax,ymin,ymax,xlo,xhi,ylo,yhi)
2. C      sets plot window size.
3. C      data fmin /0.1/, fmax /0.9/, fav /0.5/
4. C
5. C      delx=xmax-xmin
6. C      dely=ymax-ymin
7. C      if (dely dt delx) go to 10
8. C      xlo=fmin
9. C      xhi=fmax
10. C      fy=(xhi-xlo)*dely/delx
11. C      yhi=fav*0.5*fy
12. C      ylo=yhi-fy
13. C      return
14. C
15. C      ylo=fmin
16. C      yhi=fmax
17. C      fx=(yhi-ylo)*delx/dely
18. C      xlo=fav*0.5*fx
19. C      xhi=xlo+fx
20. C      return
21. C
22. C      end

```


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

NUMBER USE SOURCE PROGRAM REFERENCES

10 36d 121 5j
 (SN=STATEMENT NUMBER, GSN=GENERATED STATEMENT NUMBER)
 (FN=FORMAT NUMBER, UNDEF=UNDEFINED STATEMENT NUMBER)
 TABLE OF NAMES ENCOUNTERED (ADDRESSES FOR DUMMY ARGUMENT IS THE ARGUMENT NUMBER)

ADDRESS NAME	TYPE	MAIN USAGE	BLOCK	SOURCE	PROGRAM REFERENCES
6 DFLX	R	VARIABLE		140	80 35
7 DFLY	R	VARIABLE		140	80 45
5 IAV	R	VARIABLE		150	25
4 IMAX	R	VARIABLE		130	70 25
3 IMIN	R	VARIABLE		120	60 25
11 IY	R	VARIABLE		160	150 145
10 IY	R	VARIABLE		100	90 85
13 MAIN		ENTRY		10/2	
6 X10	R	VARIABLE	DUM. ARG.	160	155 80 75 10
5 XLO	R	VARIABLE	DUM. ARG.	165	80 65 10
2 XMAX	R	VARIABLE	DUM. ARG.	30	10
1 XMIN	R	VARIABLE	DUM. ARG.	30	10
10 Y10	R	VARIABLE	DUM. ARG.	140	135 100 95 10
7 YLO	R	VARIABLE	DUM. ARG.	140	125 105 10
4 YMAX	R	VARIABLE	DUM. ARG.	40	10
3 YMIN	R	VARIABLE	DUM. ARG.	40	10

TABLE OF EXTERNAL NAMES

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

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

BLOCK NAMES AND LENGTHS IN OCTAL

52-WSIZE 14 91P 0 #C1

STATIC SPACE (IN OCTAL)

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

```
1999 1 C subroutine cosp1 (rrr,eee,b, gamma,ppp)
2000 C
2001 C equation of state routine for air.
2002 C
2003 C input variable definitions.
2004 C rrr = mass density
2005 C eee = internal energy per unit volume
2006 C (converted for internal use to energy per unit mass)
2007 C n = number of entries in arrays rrr & eee
2008 C
2009 C this routine calculates gamma and ppp as follows:
2010 C gamma = 1 + p/(rho*e)
2011 C ppp = pressure(units of eee)
2012 C
2013 C
2014 C parameter (m=92)
2015 C
2016 C dimension rrr(n), eee(n), gamma(n), ppp(n)
2017 C dimension t1(m), t2(m), t22(m), t222(m), rho(m), e(m), int(m)
2018 C dimension p(m), emp(m), q(m), j(m), j1(m), j2(m)
2019 C dimension g1(8,105), jcy(m), js(m)
2020 C dimension g1(120), g2(120), g3(120), g4(120), g5(120), g6(120)
2021 C
2022 C equivalence (g1(1),g(1,1)), (g2(1),g(1,16)), (g3(1),g(1,31))
2023 C equivalence (g4(1),g(1,46)), (g5(1),g(1,61)), (g6(1),g(1,76))
2024 C equivalence (g7(1),g(1,91)), (int(1),tem(1)), (g(1),g(1,1))
2025 C
2026 C data zero1 /774 413/
2027 C data x116e /2.7725887222397744835689081810414791107177734375/
2028 C data r116e /0.36067376022224095773465197533690643310546875/
2029 C
2030 C q = gamma 1.0 is stored for 32 bit word machines in powers of
2031 C 16 across for mass density variation and intermediate values
2032 C l 16 for powers of 16 vertically which represent the internal
2033 C energy variation.
2034 C
2035 C 16**(2) qp rho qp 16**(6)
2036 C 16**(8) l e e 16 16**(15)
2037 C
2038 C data g1 /8.4222,8*.4152,8*.4110,8*.4081,8*.4058,8*.4040,8*.4024,8
2039 C 1*.4011,8*.3998,8*.3988,8*.3978,8*.3969,8*.3961,8*.3953,8*.3935/
2040 C data g2 / 3918.,3918.,3918.,3918.,3918.,3918.,3918.,3918.,3918.,3723.,37
2041 C 15.,3707.,3699.,3690.,3680.,3663.,3637.,3555.,3538.,3522.,3502.,3
2042 C 416.,3430.,3344.,3238.,3170.,3170.,3370.,3364.,3347.,3277.,3099.,
2043 C 3 2885.,3257.,3227.,3201.,3134.,3062.,3014.,2884.,2591.,3166.,3110,
2044 C 4 3063.,2946.,2831.,2783.,2677.,2358.,3111.,3006.,2940.,2787.,2635
2045 C 5 *.2588.,2502.,2236.,3075.,2906.,2810.,2665.,2466.,2418.,2350.,213
2046 C 6 1.,3043.,2819.,2695.,2554.,2317.,2269.,2216.,2038.,2929.,2740.,25
2047 C 7 93.,2455.,2206.,2136.,2097.,1915.,2840.,2672.,2500.,2366.,2166.,2
2048 C 8 019.,1988.,1879.,2764.,2611.,2429.,2285.,2129.,1890.,1811.,
2049 C 9 2114.,2055.,2384.,2210.,2079.,1818.,1799.,1747.,2669.,2504.,2343,
2050 C $ 2141.,2037.,1822.,1709.,1689.,2624.,2473.,2704.,2096.,1998.,1828
2051 C $.1684.,1639/
2052 C data g3 / 2599.,2446.,2268.,2087.,1961.,1874.,1673.,1601.,2401.,21
2053 C 1 91.,1972.,1775.,1592.,1444.,1358.,1203.,2002.,1969.,1749.,1536.,1
2054 C 2 176.,1292.,1107.,1044.,1911.,1829.,1633.,1429.,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, 1895, 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, 0988, 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/
18. data 94 / 1808, 1781, 1667, 1451, 1248, 1074, 0930, 0843, 2134, 20
1 40, 1978, 1782, 1565, 1368, 1206, 1074, 2210, 2072, 1957, 1739, 1
2 516, 1312, 1137, 1000, 2245, 2109, 1983, 1772, 1563, 1390, 1247,
3 1133, 2299, 2132, 2017, 1795, 1579, 1384, 1221, 1090, 2350, 2157,
4 2023, 1798, 1575, 1370, 1197, 1057, 2397, 2194, 2034, 1796, 1572
5 1372, 1205, 1070, 2452, 2227, 2050, 1805, 1576, 1379, 1236, 111
6 8, 2510, 2256, 2069, 1814, 1581, 1383, 1231, 1103, 2560, 2282, 20
7 91, 1822, 1585, 1385, 1226, 1083, 2605, 2312, 2111, 1829, 1588, 1
8 386, 1222, 1070, 2677, 2358, 2129, 1836, 1592, 1386, 1218, 1071,
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/
19. data 95 / 2963, 2531, 2199, 1918, 1625, 1407, 1230, 1096, 4323, 35
1 82, 3109, 2889, 2803, 2706, 2410, 2224, 4610, 4026, 3624, 3212, 2
2 926, 2531, 2375, 2015, 4199, 3837, 3401, 2979, 2623, 2318, 2 08,
3 1854, 3924, 3642, 3194, 2760, 2427, 2157, 1902, 1721, 3794, 3479,
4 3025, 2673, 2311, 2019, 1842, 1613, 3674, 3448, 2961, 2593, 2255
5 1994, 1785, 1594, 3573, 3443, 2910, 2517, 2293, 2096, 1843, 167
6 9, 3661, 3438, 2935, 2597, 2336, 2225, 2143, 2116, 3674, 3435, 30
7 80, 2728, 2606, 2977, 2573, 2573, 3685, 3453, 3210, 3014, 2942, 2
8 933, 2912, 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/
20. data 96 / 4290, 4205, 4118, 4092, 4077, 4065, 4059, 4047, 5411, 53
1 85, 5359, 5353, 5351, 5350, 5350, 5823, 5812, 5801, 5797, 5
2 796, 5797, 5797, 5797, 6076, 6070, 6085, 6082, 6082, 6083, 6083,
3 6083, 6308, 6305, 6303, 6303, 6305, 6305, 6305, 6305, 6481, 6483,
4 6485, 6483, 6484, 6486, 6487, 6487, 6627, 6632, 6637, 6636, 6637
5 6640, 6640, 6640, 6754, 6761, 6769, 6768, 6770, 6773, 6773, 677
6 3, 6866, 6875, 6885, 6884, 6886, 6890, 6890, 6890, 6966, 6977, 69
7 89, 6989, 6991, 6995, 6995, 6995, 7056, 7070, 7083, 7083, 7085, 7
8 090, 7090, 7090, 7139, 7154, 7169, 7172, 7176, 7177, 7177,
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/
21. data 97 / 7411, 7422, 7453, 7454, 7457, 7463, 7463, 7463, 8069, 81
1 03, 8138, 8139, 8145, 8152, 8153, 8153, 8454, 8496, 8538, 8540, 8
2 547, 8556, 8557, 8557, 8727, 8774, 8822, 8825, 8832, 8842, 8843,
3 8843, 8938, 8990, 9042, 9046, 9054, 9064, 9065, 9065, 9111, 9166,
4 9222, 9226, 9235, 9246, 9247, 9247, 9258, 9316, 9374, 9374, 9387
5 9389, 9400, 9400, 9384, 9445, 9506, 9511, 9520, 9532, 9533, 953
6 3, 9496, 9499, 9622, 9627, 9637, 9649, 9650, 9650, 9656, 9661, 97
7 27, 9731, 9741, 9754, 9755, 9755, 9686, 9757, 9821, 9826, 9836, 9
8 849, 9850, 9850, 9769, 9837, 9906, 9912, 9922, 9936, 9937, 9937,
9 9845, 9915, 9986, 9991, 4, 9999, 9915, 9987, 6, 9999, 9981, 7, 9999/

real air eos. table lookup on gilmore data. (no temp. model)
 to avoid costly logarithmic functions the table "g" is stored in a
 form so that the hexadecimal word structure of a 32 bit machine
 may be exploited
 this logic may be transferred to other machines by recalculating
 the table "q" appropriate to the word architecture of that machine.
 machine dependent functions and key numbers must also be changed.

```

2119      1st=0
2120      nst=n
2121      10      nst=min(nst,m)
2122
2123      do 20 ire=1,nst
2124      rho(ire)=zeror(irr(istire)
2125      e(ire)=amax(3.68,eee(istire)/irr(istire))
2126
2127      c      calculate mass density variation index "i".
2128      tem(ire)=alog(rho(ire))*r116e+500.0
2129      i(ire)=tem(ire)
2130      omp(ire)=tem(ire)-i(ire)
2131      i(ire)=502-i(ire)
2132      p(ire)=1.0-omp(ire)
2133      i(ire)=max0(i(ire),1)
2134
2135      c      calculate internal energy variation index "j".
2136      tem(ire)=alog(e(ire))*r116e
2137      jcy(ire)=ifix(tem(ire))
2138      tem(ire)=tem(ire)-jcy(ire)
2139      tem(ire)=exp(xl16e*tem(ire))
2140      jcy(ire)=jcy(ire)-7
2141      js(ire)=tem(ire)
2142      q(ire)=tem(ire)-js(ire)
2143      j(ire)=js(ire)+15*jcy(ire)
2144      j(ire)=min0(j(ire),104)
2145      j(ire)=i(ire)+8*j(ire)
2146      20      i(ire)=j(ire)-8
2147
2148      do 30 ire=1,nst
2149      t11(ire)=q(i(ire))
2150      t21(ire)=q(i(ire)+1)
2151      t2(ire)=q(j(ire))
2152      t22(ire)=q(j(ire)+1)
2153
2154      c      calculate gamma by linear interpolation.
2155      do 40 ire=1,nst
2156      t12(ire)=t12(ire)-t11(ire)
2157      t22(ire)=t22(ire)-t21(ire)
2158      gamma(is(ire))=1.0-omp(ire)*(t11(ire)+t12(ire))+p(ire)*
2159      (t21(ire)+q(ire)+t22(ire))
2160
2161      or=or+nst
2162      1st=1st+nst
2163      1f=(or-qt(0))/(or-1)
2164
2165      c      calculate pressure in units of eos
2166      do 50 ire=1,n
2167      ppp(ire)=amax(10.0,eee(ire)*(gamma(ire)-1.0))
    
```

FOSP1
2167
2168
2169
FOSP1
FOSP1
FOSP1

C

59.
60.

return
and

VECTOR LOOP BEGINS AT SEQ. NO.
VECTOR LOOP BEGINS AT SEQ. NO.
VECTOR LOOP BEGINS AT SEQ. NO.

25, P= 4143C
50, P= 4267C
57, P= 4322d

MAXO	MINO	3 N	4130 NR	4131 NST	3054 OMP	2720 P	5 PPP	3210 Q	2430 R10	4126 R11G	1 RRR	4124 R21R01	1650 T11	2004 T12	2140 T21	2274 T22	1514 TEM	4125 X11G	
I	I	I	I	I	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
INI	INI	VARIABLE	VARIABLE	VARIABLE	IDIM ARRAY	IDIM ARRAY	IDIM ARRAY	IDIM ARRAY	IDIM ARRAY	VARIABLE	IDIM ARRAY	VARIABLE	IDIM ARRAY	IDIM ARRAY	IDIM ARRAY	IDIM ARRAY	ID EQ. ARRAY	VARIABLE	
330	420	57N	560	550	530	530	58S	530/2	28P	34U	27U	26U	53U	53U	53U	53U	40U	34U	
24U	23U	54S	54U	54U	32U	32S	3P	40S	26S	28U	26U	12S	51U	51U	52U	52U	39U	28S	
		31/4	24P	45N	50	10	10		14S	30	10		46S	48S	47S	49S	37U	110	
			23S	25N									4D	4D	4D	4D	36U	34S	
			24S															36S	30U

TABLE OF EXTERNAL NAMES

ABBRVIATIONS USED ABOVE (THESE ARE KEYS TO THE SOURCE LISTING LINE NUMBER)	34U	28U
A USED IN FORTRAN ASSIGN STATEMENT	34U	28U
D DEFINED IN DECLARATIVE STATEMENT	37U	
F STATEMENT NUMBER ENDING A DO LOOP		
I INDEX OF A DO OR IMPLIED DO LOOP		
J STATEMENT NUMBER USED IN TRANSFER		
L SOURCE LINE OF A STATEMENT NUMBER		
N NAME USED AS A DO LOOP PARAMETER		
P USED IN CALL/FUNC CALL OR ARRAY DEF		
R FORMAT USED IN A READ STATEMENT		
S STORED SO CONTENTS MAY BE CHANGED		
U NAME USED IN EXECUTABLE STATEMENT		
W FORMAT USED IN A WRITE STATEMENT		
X DEFINED OR DECLARED BUT NOT USED		
Y LEN OR MORE REFERENCES TO SYMBOL		

M = 92

TABLE OF LOOPS ENCOUNTERED

LABEL INDEX	FROM	TO	ADDRESS	LENGTH
20 IRE	25	44	41620	70
30 IRE	45	49	42570	10
40 IRE	50	53	42770	23
50 IRE	57	58	43400	11

BLOCK NAMES AND LENGTHS IN DECIMAL

240 #IB 3 #CI

4354 EOSP1
STATIC SPACE (IN OCTAL)

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

TOTAL: 4617

2170
 2171
 2172
 2173
 2174
 2175
 2176
 2177
 2178
 2179
 2180
 2181
 EDSP2

```

1. C .....
   C suboutline eosp2 (ccc,ccc,n,gamma,ppp)
   C .....
   C equation of state for dust.
   C .....
2. real rcc(n), ccc(n), gamma(n), ppp(n)
3. data gamma,gamam / 1.4,0.4/
4. do 10 i=1,n
5. gamma(i)=gamma
6. ppp(i)=ccc(i)*gamam
7. return
8. end

```

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 ** 61 41
 10A27d 41
 10B37a 41

(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	IDIM ARRAY	DUM.ARG.	6U	2P	1D
7	FOSP2		ENTRY		1D/2		
3	GAMA	R	VARIABLE		5U	3S	
4	GAMAM1	R	VARIABLE		6U	3S	
4	GAMMA	*R	IDIM ARRAY	DUM.ARG.	5S	2P	1D
5	I	I	VARIABLE		6U/2	5U	41
	MAIN.		ENTRY				
3	N	I	VARIABLE	DUM.ARG.	4N	2P/4	1D
5	PIP	*R	IDIM 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-EDSP2 27 #IB 0 #CL

STATIC SPACE (IN OCTAL)	
B SAVE:	25
F SAVE:	2
CONSTANTS:	1
VARIABLES:	5
TEMPORARIES:	0
CODE:	34
TOTAL:	71

TABLE OF EXTERNAL NAMES

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

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

- A USED IN FORTRAN ASSIGN STATEMENT
- D DEFINED IN DECLARATIVE STATEMENT
- E STATEMENT NUMBER ENDING A DO LOOP
- I INDEX OF A DO OR IMPLIED DO LOOP
- J STATEMENT NUMBER USED IN TRANSFER
- L SOURCE LINE OF A STATEMENT NUMBER
- N NAME USED AS A DO LOOP PARAMETER
- P USED IN CALL/FUNC CALL OR ARRAY DEF
- R FORMAT USED IN A READ STATEMENT
- S STORED SO CONTENTS MAY BE CHANGED
- U NAME USED IN EXECUTABLE STATEMENT
- W FORMAT USED IN A WRITE STATEMENT
- X DEFINED OR DECLARED BUT NOT USED
- ? 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 FDS 60 #1B 22 #CL

STATIC SPAC (IN OCTAL)

B SAVE	57
F 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
DISIPL	40
EOS	109
FOSP1	99
FOSP2	106
FULLO	61
FLTK3D	1
MYCON	85
PLIARR	65
PROF3D	46
RESTRI	72
WSIZE	96

*** COMPILATION SUMMARY ***

EXECUTE LINE - 1-mp1ot3d on:adix
CPU SECONDS - 3.1893
I/O SECONDS - 1.0630
SYS SECONDS - 0.0472
SIZE(100%AL) - 60100X0

END

DATE

FILMED

DTIC

JULY 88