```
      subroutine adummy

c..... cliche storage set up here

      cliche param
      parameter (izx=30, jrx=30, izx2=izx-2  )
      parameter ( itbw=2*izx-1  )
      parameter (kxp=(izx-2)*(jrx-2), ibw=izx-1 )
      parameter (nplt=3000, nps=100 )
      endcliche
      cliche fstor
      use param
      common/fun/f1(izx,jrx),f2(izx,jrx),f3(izx,jrx),f4(izx,jrx)
     c ,f5(izx,jrx),f7(izx,jrx),
     c g1(izx,jrx),g2(izx,jrx),g3(izx,jrx),g4(izx,jrx)

      common/equil/b(izx,jrx),rho(izx,jrx),qub(izx,jrx),r(izx,jrx)
     c ,phi(izx,jrx),yep(izx,jrx),chi(izx,jrx),qv(izx,jrx)
      common/pertur/xioo(kxp),xio(kxp),xiol(kxp)
     c ,xroo(kxp),xro(kxp),xrol(kxp)
      endcliche

      cliche matrix
      use param
      common/coeff/a1(kxp,9),a2(kxp,9),a3(kxp,9),b1(kxp,3)
     c ,rhs1(kxp),rhs2(kxp)
c.....unnamed common for dynamic memeory expansion
      common ww1(1), ww1(1)
      endcliche

      cliche const
      use param
      common/con/gam1,gam2,ix,jx,mm,izxp,kxx,nmax,lmax,isw,ihbw
     c ,fac1,fac2,bias,du,dv,dt,ndiag,ex0,b0,rho0,ex1,fi1,fizx,fj1
     c ,fjrx,psi0,z0,kplot,npm,fpsi,fz,fu,fv,azm,apsim,u0,v0,amass
     c ,fourpi,omegst,omegr,omegexb,fir,sf6,sf8,kplotm,kzs,zedge
     c ,cpuo,cio,syso,valfk,xu,xv
      common/mesh/psi(jrx),z(izx),u(izx),v(jrx),dpsi(jrx),dz(izx)
     c ,vpsi(jrx),uuz(izx),vpsih(jrx),uuzh(izx)
      common/graf/  xrtime(nplt),xrspz(izx,nps),xrsppsi(jrx,2*nps)
     c ,time(nplt)
      common/rotcon/cr,r0sq,rp,rw,beta0,ratrod,ti,en0,cee,r0,
     c echarg,enbar
      endcliche

      return
      end
```

```
641   40    continue
642         j=2
643         do 45 i=2,ix-1
644         k1=i-1+(j-2)*(ix-2)
645         k2=j-1+(jx-2)*(i-2)
646         k=.5*(1+isw)*k1+.5*(1-isw)*k2
647         do 44 m=4,6
648         a1(k,m-3)=0.
649         a2(k,m-3)=0.
650         a3(k,m-3)=0.
651    44   continue
652         b1(k,1)=0.
653    45   continue
654         j=jx-1
655         do 48 i=2,ix-1
656         k1=i-1+(j-2)*(ix-2)
657         k2=j-1+(jx-2)*(i-2)
658         k=.5*(1+isw)*k1+.5*(1-isw)*k2
659         do 47 m=4,6
660         a1(k,m+3)=0.
661         a3(k,m+3)=0.
662         a2(k,m+3)=0.
663    47   continue
664         b1(k,3)=0.
665    48   continue
666         return
667         end
```

```
48
49 c..... the main routine
50
51 c..... notice of 4/8/82, this version runs correctly for isw=1, and
52 c..... runs correctly for isw=-1 .
53
54 c.....5/12/82, flora runs testcase 1 , 0 beta, 0 pressure, homogeneous
55 c..... plasma, correctly.
56
57 c..... flora1 transforms variables z,psi to u,v which are always equally
58 c..... spaced, transformation: z=au*u**xu, and psi=apsi*v**xv, where
59 c..... zmax=umax, psimax=vmax, and fz*zmax=fu*umax, fpsi*psimax=fv*vmax .
60 c..... fz, fu, fpsi, fv, input, xu=ln fz / ln fu, xv=ln fpsi / ln fv .
61 c..... au=umax**(-xx+1) , apsi=vmax**(-yy+1) .
62
63 c..... flora2 solves test case 2 , rotating rigid rotor stability, ref:
64 c..... freidberg and pearlstein, phys fluids 21(7) july 1978 1207
65
66
67 c.....flora4 includes background constant density, enbar ( as does flora3 ),
68 c..... and kzs switch which when set to zero, generates initial perturbations
69 c..... independent of z in random spatial generator (ex0=1.) .
70
71
72 c..... flora5
73 c.....  is vectorized version of flora4,(calls rightvec instead of
74 c..... right ), also has timing routine from b. langdon (requires
75 c..... bzohar loaded as a binary ),
76 c..... insert cliche storage here
77
78
79 c..... flora7 is mod. flora5, with psi stretching function
80 c..... exactly centered in amat, (flora5 used linear interpolation
81 c..... to get vpsi(j+1/2)), Also revised diagnostic plots included,
82       use param
83       use fstor
84       use matrix
85       use const
86
87       data tim/1.e6/
88       integer tallyb(2000b)
89       common / q8locs/locf(0:15)
90       data itally/1/
91
92 c.....call link call here
93       call link('unit59=terminal,unit2=(inflora,open),unit3=(output,
94      c create) //')
95
96       if(itally.gt.0) then
97       do 200 ii=1,15
98  200  if(locf(ii).eq.0)go to 210
99       ii=0
100 210  ioctally=ii
101      if(ioctally.eq.0)go to 299
102      call timer(ioctally,'ztally00',tallyb,2000b,floratim,1)
103 299  itally=-1
104      endif
105      isw=1
106      if(izx.gt.jrx)isw=-1
107      jtbw=.5*(1+isw)*itbw+.5*(1-isw)*(2*jrx-1)
```

```
      subroutine comat(abar,nd)

c.....transforms the elements of the a1(k,m) array into into the
c..... elements of the compressed column matrix abar which will be
c..... operated upon by banfac and bansol.

c..... insert storage cliches here
      use param
      use fstor
      use matrix
      use const

      dimension abar(kxp,1)


      kxx=kxp
      len=ltbw*kxp
      call bcast(abar(1,1),0.,len)
      do 10 k=1,kxx
      do 10 m=1,9
      lp1=m+((m-1)/3)*(ihbw-4)
      lp2=1+mod(m-1,3)*(ihbw-1)+(m-1)/3
      lp=.5*(1+isw)*lp1+.5*(1-isw)*lp2
      abar(k,lp)=a1(k,m)
   10 continue
      return
      end
```

```
      ihbw=.5*(1+isw)*ibw+.5*(1-isw)*(jrx-1)
      nn=jtbw*kxp
      nn1=jtbw+kxp
      call memory(ww,nn-1)
      call memory(ww1(nn),nn1)
      call input
      call rigidcon
      call grid
      call constant
      call equilrot
      call fltoll
      call amat
      call comat(ww,jtbw)
      call initial
      call mymove(xrol(1),xro(1),kxx)
      call mymove(xiol(1),xio(1),kxx)
      call mymove(xroo(1),xro(1),kxx)
      call mymove(xioo(1),xio(1),kxx)
      call banfac(kxp,ihbw,ww,1,-(kxp-1))
      t=0.
      do 100 n=1,nmax
      t=t+dt
      time(n)=t
      fac1=-1./dt
      fac2=1./dt
      do 90 l=0,lmax
      call rightvec
      call zmovewrd(ww1(nn),rhs1,kxx)
      call bansol(kxp,ihbw,ww,1,-(kxp-1),ww1(nn))
      do 10 j=2,jx-1
      kp=1+izxp*(j-2)
      call zmovewrd(xrol,ww1(nn),kxx)
   10 continue
      call zmovewrd(ww1(nn),rhs2,kxx)
      call bansol(kxp,ihbw,ww,1,-(kxp-1),ww1(nn))
      do 20 j=2,jx-1
      kp=1+kxp*(j-2)
      call zmovewrd(xiol,ww1(nn),kxx)
   20 continue
      fac1=-.5/dt
   90 fac2=.5/dt
      call zmovewrd(xioo,xio,kxx)
      call zmovewrd(xio,xiol,kxx)
      call zmovewrd(xroo,xro,kxx)
      call zmovewrd(xro,xrol,kxx)
c..... time array
      xrtime(n)=xro(kplot)
      if(mod(n,ndiag).eq.0)call diagno
  100 continue
      call timeused(icp,io,isy)
      cpuo=icp*tim
      cio=io*tim
      syso=isy*tim
      call picture
      call timend
      call exit(1)
      end
```

```
      subroutine right

c..... calculates right hand side vector for both equations,
c..... rhs1(k)=2*a2*xr(n)-a2*xr(n-1)+b1*(xl(1)-xl(n-1)) , and
c..... rhs2(k)=2*a3*xl(n)-a2*xl(n-1)+b1*(xr(1)-xr(n-1)) ,

c..... insert cliches for storage here
      use param
      use fstor
      use matrix
      use const

      do 10 j=2,jx-1
      do 10 i=2,ix-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      t1=0.
      t2=0.
      t3=0.
      tt1=0.
      tt2=0.
      tt3=0.
      do 5 m=1,9
      ip=i-2+m-((m-1)/3)*3
      jp=j-1+(m-1)/3
      if(jp.eq.1.or.jp.eq.jrx.or.ip.eq.1.or.ip.eq.izx)go to 5
      kp1=ip-1+(ix-2)*(jp-2)
      kp2=jp-1+(jx-2)*(ip-2)
      kp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
      t1=t1+a2(k,m)*xroo(kp)
      t2=t2+a3(k,m)*xro(kp)
      tt1=tt1+a2(k,m)*xloo(kp)
      tt2=tt2+a3(k,m)*xlo(kp)
    5 continue
      do 6 mn=1,3
      jq=j-2+mn
      if(jq.eq.1.or.jq.eq.jrx)go to 6
      kq1=i-1+(ix-2)*(jq-2)
      kq2=jq-1+(jx-2)*(i-2)
      kq=.5*(1+isw)*kq1+.5*(1-isw)*kq2
      t3=t3+b1(k,mn)*(xlo(kq)-xloo(kq))
      tt3=tt3+b1(k,mn)*(xro(kq)-xroo(kq))
    6 continue
      rhs1(k)=(2.*t2-t1+fac1*t3)
      rhs2(k)=2.*tt2-tt1+fac2*tt3
   10 continue
      return
      end
```

```
      subroutine constant

c..... insert storage cliche here
      use param
      use fstor
      use matrix
      use const

      gam1=.25*(3*bias+1)
      gam2=.25*(1-bias)
      ip=.5*(ix-2)
      jp=.5*(jx-2)
      kp1=ip-1+(jp-2)*(ix-2)
      kp2=jp-1+(jx-2)*(ip-2)
      kplot=.5*(1+isw)*kp1+.5*(1-isw)*kp2
      if(kplotm.ne.0)kplot=kplotm
       return
      end
```

```
      subroutine rightvec

c..... calculates right hand side vector for both equations,
c..... rhs1(k)=2*a2*xr(n)-a2*xr(n-1)+b1*(xi(1)-xi(n-1)) , and
c..... rhs2(k)=2*a3*xi(n)-a2*xi(n-1)+b1*(xr(1)-xr(n-1)) ,

c..... insert cliches for storage here
      use param
      use fstor
      use matrix
      use const

      call sscal(kxx,0.,rhs1,1)
      call sscal(kxx,0.,rhs2,1)
      do 100 m=1,9
      mdel=(m-1)/3
      m1=m-1
      koff1=-mdel*5+m+(mdel-1)*ix
      koff2=(m-((mdel+1)*3-1))*jx+1-2*m1+7*mdel
      koff=.5*(1+isw)*koff1+.5*(1-isw)*koff2
      do 110 k=1,kxx
      rhs1(k)=rhs1(k)+2.*a3(k,m)*xro(k+koff)-a2(k,m)*xroo(k+koff)
110   rhs2(k)=rhs2(k)+2.*a3(k,m)*xio(k+koff)-a2(k,m)*xioo(k+koff)
      if(m.eq.2.or.m.eq.5.or.m.eq.8)go to 119
      go to 100
119   continue
      do 120 k=1,kxx
      mbar=m-1-(m/4)*2
      rhs1(k)=rhs1(k)+fac1*b1(k,mbar)*(xio(k+koff)-xioo(k+koff))
120   rhs2(k)=rhs2(k)+fac2*b1(k,mbar)*(xro(k+koff)-xroo(k+koff))
100   continue
      return
      end
```

```
183
184
185       subroutine equil
186
187 c.....special case equilibrium, 0 beta, 0 pressure, rho=const.
188 c.....  test case 1
189 c.....set up 1/4/82 by r. freis
190
191 c.....insert cliche storage here
192       use param
193       use matrix
194       use const
195       use fstor
196
197       data rho0/1.e12/,b0/1.e4/,azm/1./,apsim/1./
198
199       do 10  j=1,jx
200       do 10 i=1,ix
201       uz=uuz(i)
202       rho(i,j)=rho0
203       b(i,j)=b0
204       r(i,j)=sqrt(2.*abs(psi(j))/b0)
205       chi(i,j)=0.
206       yep(i,j)=0.
207       qub(i,j)=b0
208  10   continue
209       return
210       end
```

```
      subroutine rigidcon

c..... special constants needed for rigid rotor equilibrium.

c..... storeage cliche here
      use param
      use const

c..... input for rigid rotor
      data echarg/4.8e-10/, en0/1.00e+12/, b0/1.e4/, amass/3.34e-24/
     c , cee/3.e10/, valfk/.4/, fourpi/12.56637/, pi/3.1415926/
     c , enbar/0.e11/
      namelist/rotor/b0,beta0,ratrod,valfk,en0,echarg,r0,rwb,enbar
      call ddi(rotor,2,3,1)

      carg=sqrt(1.-beta0)
      r0sq=r0**2
      aasq=r0sq*(1.-carg)/beta0
      cr=.5*(alog(1.+carg)-alog(1.-carg))
      aa=sqrt(aasq)
      rw=rwb*aa
      valf=b0/(sqrt(fourpi*en0*amass))
      omegci=echarg*b0/(amass*cee)
      omegp2=fourpi*en0*echarg**2/amass
      omegst=2.*beta0*omegci*cee**2/(omegp2*r0sq)
      vomeg=echarg*sqrt(en0*fourpi/amass)*r0sq*.5/(beta0*cee)
      u(ix)=(pi*valf/(2*omegst*valfk))/(1-.5/(ix-1.5))
      du=(u(ix)-u0)/(ix-1.5)
      targ=cosh(rw**2/r0sq+cr)*sqrt(beta0)
      v(jx)=b0*r0sq*.5*alog(targ)/sqrt(fourpi)
      dv=(v(jx)-v0)/(jx-1.5)
      return
      end
```

```
      subroutine equilrot

c..... sets up equilibrium for rigid rotor, test case 2 .
c.......flora3 adds cold plasma halo to equilbruim density

c..... insert cliche storage here
      use param
      use const
      use fstor

      ps10=b0*r0sq*.5/sqrt(fourpi)
      omegr=ratrod*omegst*(1.-enbar/en0)
      foursq=sqrt(fourpi)
      do 5 i=1,ix
      uz=uuz(i)
      do 5 j=1,jx
      fac=exp(psi(j)/ps10)/sqrt(beta0)
      b(i,j)=b0*sqrt(fac**2-1.)/(fac*foursq)
      rho(i,j)=en0*amass/(beta0*fac**2)+enbar*amass
      beta=1/fac**2
      arg1=fac+sqrt(fac**2-1.)
      acosh=alog(arg1)
       r(i,j)=r0*sqrt(-cr+acosh)
      qub(i,j)=b(i,j)
      omegstr=omegst*(1.-enbar*amass/rho(i,j))
      entest=enbar*amass
      if(entest.ge.rho(i,j))omegstr=0.
      omegexb=(1.+ratrod)*omegstr
      omeggb=+beta*omegstr*.5/(1.-beta)
       chi(i,j)=rho(i,j)*(2.*omegexb+omeggb-omegst)
      yep(i,j)=-rho(i,j)*(omegexb+omeggb)*(omegexb-omegst)
      chi(i,j)=chi(i,j)*flr
      yep(i,j)=yep(i,j)*flr
 5    continue
      do 30 i=1,ix
      do 30 j=2,jx-1
      qv(i,j)=.5*(qub(i,j+1)*b(i,j+1)-qub(i,j-1)*b(i,j-1))
 30   continue
      return
      end
```

```
812
813         subroutine mymove(a,b,len)
814         dimension a(1),b(1)
815         do 10 i=1,len
816         a(i)=b(i)
817   10    continue
818         return
819         end
```

```
      subroutine initial

c.....set up initial displacement vectors, xro and xio
c..... test case 1, cos(kz) in z, flat in psi
c..... set up 1/4/82 by r. freis

c.....insert cliche storage here
      use param
      use fstor
      use matrix
      use const

      data pi/3.1415926/

      do 10 j=2,jx-1
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      do 10 i=2,ix-1
      if(kzs.eq.0)go to 5
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
    5 continue
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      xro(k)=ex0*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xio(k)=ex0*(r3+r4-1.)+ex1*cos(.5*pi*z(i)/zedge)
   10 continue
      return
      end
```

```
      subroutine picture
c...uses graflc, graflib and grafcore to make plots of xr vs. time and
c....space.


c.....insert cliche for common  here
      use parem
      use fstor
      use matrix
      use const

      dimension iy(2), it(5), lab(2),dum(izx2),ymin(5),ymax(5)
     c ,dum1(npit),rplot(jrx-1)
      data epp/1.e20/
      call pstart(dev,4rplot,1,'box u21$',1)
      call p100
      call orgfile(nume)
      write(100,102)nume
 102  format(10x,'this problem run by ',a8 )
      write (100,101) dt, ix,jx,nmax,lmax,bias,ratrod,omegst,
     c omegr,omegexb,flr,sf6,sf8,enbar,kplot,kzs,valfk,cpuo,cio,syso
     c ,xu,xv,en0,b0
 101  format(////'dt=',e16.6/'ix=',i8/'jx=',i8/'total time steps =',i8/
     c 'no. of iterations =',i8/'bias=',f10.5/'ratrod=',e16.6/
     c 'drift freq. (omegst) =',e16.8/'rotation freq. (omegr) =',e16.8/
     c 'e x b freq. (omegexb) =',e16.8/'flr=',e16.8/'sf6=',e16.8/
     c 'sf8=',e16.8/'enbar=',e16.8/'kplot=',i8/'kzs=',i8/'valfk =',e16.8
     c /5x,'cpu time =',e16.8/
     c 5x,'i-o time =',e16.8/5x,'sys time =',e16.8/3x,'u exponent (xu) =
     c ',e16.8/3x,'v exponent (xv)=',e16.8/5x,'en0=',e16.8/5x,
     c 'b0=',e16.8)
c.....plot coordinate stretching

      kx='u$'
      iy(1)='z$'
      it(1)='z vs u, '
      it(2)='(z=const'
      it(3)='*u**xu)$'
      call pframe
      call pscale(0,u(2),u(ix),z(2),z(ix),1)
      call pcurve(0,u(2),z(2),ix-1,1,it,kx,iy,1h$,1h$)
      kx='v$'
      iy(1)='psi$'
      it(1)='psi vs v'
      it(2)=', (psi=c'
      it(3)='onst*v**'
      it(4)='xv)$'
      call pscale(0,v(2),v(ix),psi(2),psi(jx),2)
      call pcurve(0,v(2),psi(2),jx-1,2,it,kx,iy,1h$,1h$)
      iy(1)='r(   ,v)$'
      it(1)='r(   ,v) '
      it(2)='vs v$'
      ipld=ix/4
      call zcitoa(xout,0,ipld,2,0)
      call cmove(it(1),2,xout,0,2)
      call cmove(iy(1),2,xout,0,2)
      do 5 jj=1,jx-1
    5 rplot(jj)=r(ipld,jj+1)
      call pscale(0,v(2),v(ix),rplot(1),rplot(jx-1),3)
```

```
285
286       subroutine input
287 c.....insert storage cliches here
288       use param
289       use const
290
291 c..... boundary conditions are set as follows:
292 c.....       at z=z0 (i=1), fi1=-1. implies x=0.
293 c.....                      fi1=1. implies slope=0.
294 c.....      at z=zmax (i=ix), fizx=-1. implies x=0
295 c.....                       fizx=1. implies slope=0.
296 c.....     at psi=psi0 (j=1), fji=-1. implies x=0.
297 c.....                       fj1=1. implies slope=0.
298 c.....     at psi=max (j=jx), fjrx=-1. implies x=0.
299 c.....                       fjrx=1. implies slope=0.
300       data mm/4/, bias/.5/ lmax/2/, nmax/5/,dv/1./, du/1./,dt/1./
301      c ,ndiag/100/,fi1/1./,fizx/1./,fj1/1./,fjrx/1./,flr/1./
302      c ,sf6/1./, sf8/1./, kplotm/0/, kzs/1/
303
304       namelist/now1/mm, bias, lmax, nmax,dv, dt, du,ndiag
305      c ,fi1,fizx,fj1,fjrx,rho0,b0,ex0,ex1,u0,v0,fpsi,fu,fv,fz
306      c ,azm,apsim,flr,sf6,sf8,kplotm,kzs
307
308       call ddi(now1,2,3,1)
309       jx=jrx
310       kxx=kxp
311       ix=izx
312       return
313       end
```

```
      call pcurve(0,v(2),rplot(1),jx-1,3,it,kx,iy,1h$,1h$)
      it(4)='each     '
      it(5)='th time$'
      call zcitoa(xout,0,ndiag,3,0)
      call cmove(it(4),5,xout,0,3)
      kx='z$'
      iy(1)='xr(z,psi'
      iy(2)='p)$'
      it(1)='xr(z,psi'
      it(2)='(    )) v'
      it(3)='s z      '
      jp1=jx/2
      call zcitoa(xout,0,jp1,3,0)
      call cmove(it(2),1,xout,0,3)
      do 20 np=1,npm
      lab(1)=5hn=  $
      call zcitoa(xout,2,np,2,0)
      call cmove(lab(1),2,xout,2,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=izx-2
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 40
      do 50 nnp=np,np+4
      nnpp=nnp-np+1
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 55 ip=2,ix-1
      ymax(nnpp)=amax1(ymax(nnpp),xrspz(ip,nnp))
55    ymin(nnpp)=amin1(ymin(nnpp),xrspz(ip,nnp))
50    continue
      ymin1=epp
      ymax1=-epp
      do 57 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
57    ymin1=amin1(ymin1,ymin(nn))
      call pscale(0,z(2),z(ix-1),ymin1,ymax1,n4)
40    continue
      call pcurve(0,z(2),xrspz(2,np),n3,n4,it,kx,iy,1h$,lab)
20    continue
      kx='psi$'
      do 122 ll=1,2
      ip1=ix/2**ll
      iy(1)='xr(zp,ps'
      iy(2)='i)$'
      it(1)='xr(z(    '
      it(2)='),psi) v'
      it(3)='s psi    '
      call zcitoa(xout,0,ip1,3,0)
      call cmove(it(1),5,xout,0,3)
      do 120 np=1+(ll-1)*nps,npm+(ll-1)*nps
       npld=np-(ll-1)*nps
      lab(1)=5hn=  $
      call zcitoa(xout,0,npld,2,0)
      call cmove(lab(1),2,xout,0,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=jx-2
```

```
314
315       subroutine grid
316
317 c..... relates physical grid z,psi to computational grid u,v (equally
318 c..... spaced ). uses input fpsi, fv, fz, fu and azm, apsim .
319
320 c.....insert cliche storage here
321       use param
322       use const
323
324       xv=alog(fpsi)/alog(fv)
325       xu=alog(fz)/alog(fu)
326       zzp=0.
327       psip=0.
328       do 5 i=1,ix
329    5  u(i)=u0+du*(i-1.5)
330       u(1)=-u(1)
331       azm=u(ix)**(1.-xu)
332       do 10 i=1,ix
333       uuz(i)=u(i)**(1.-xu)/(xu*azm)
334       z(i)=azm*u(i)**xu
335
336       uuzh(i)=(u(i)+.5*du)**(1.-xu)/(xu*azm)
337       dz(i)=z(i)-zzp
338       zzp=z(i)
339    10 continue
340       zedge=azm*.5*(u(ix)**xu+u(ix-1)**xu)
341       do 15 j=1,jx
342       v(j)=v0+(j-1.5)*dv
343    15 continue
344       v(1)=-v(1)
345       apsim=v(jx)**(1.-xv)
346       do 20 j=1,jx
347       vpsi(j)=v(j)**(1.-xv)/(apsim*xv)
348       vpsih(j)=(v(j)+.5*dv)**(1.-xv)/(apsim*xv)
349       psi(j)=apsim*v(j)**xv
350       dpsi(j)=psi(j)-psip
351       psip=psi(j)
352    20 continue
353       return
354        end
```

```
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 140
      do 150 nnp=np,np+4
      nnpp=nnp-np+1
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 155 ip=2,jx-1
      ymax(nnpp)=amax1(ymax(nnpp),xrsppsi(ip,nnp))
  155  ymin(nnpp)=amin1(ymin(nnpp),xrsppsi(ip,nnp))
  150  continue
      ymin1=epp
      ymax1=-epp
      do 157 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
  157  ymin1=amin1(ymin1,ymin(nn))
      call pscale(0,psi(2),psi(jx-1),ymin1,ymax1,n4)
  140  continue
      call pcurve(0,psi(2),xrsppsi(2,np),n3,n4,it,kx,iy,1h$,lab)
  120 continue
  122 continue
      kx='r$'
      iy(1)='xr(zp,r)'
      iy(2)='$'
      it(1)='xr(z(    '
      it(2)='),r) vs  '
      it(3)='r,       '
      do 220 ip=1,2
      ipl=ix/2**ip
      call zcitoa(xout,0,ipl,3,0)
      call cmove(it(1),5,xout,0,3)
      do 300 jj=1,jx-2
      rplot(jj)=r(ipl,jj+1)
  300 continue
      do 220 np=1+(ip-1)*nps,npm+(ip-1)*nps
      lab(1)=5hn=  $
      npld=np-(ip-1)*nps
      call zcitoa(xout,0,npld,2,0)
      call cmove(lab(1),2,xout,0,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=jx-2
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 240
      do 250 nnp=np,np+4
      nnpp=nnp-np+1
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 255 ip=2,jx-1
      ymax(nnpp)=amax1(ymax(nnpp),xrsppsi(ip,nnp))
  255  ymin(nnpp)=amin1(ymin(nnpp),xrsppsi(ip,nnp))
  250  continue
      ymin1=epp
      ymax1=-epp
      do 257 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
  257  ymin1=amin1(ymin1,ymin(nn))
```

```
      subroutine f1to11
c.....calculates the f1 to f11 functions needed to generate the a and
c..... b matrices . uses the equilibrium quantities r, rho, b, etc.
c..... insert cliche storage here

      use param
      use fstor
      use matrix
      use const

      m2=mm**2
      du2=du**2
      do 10 i=1,ix
      do 10 j=2,jx
      r2=r(i,j)**2
      uz=uuz(i)
      bb=b(i,j)
      vp=vps(j)
      r4=r2**2
      f1(i,j)=rho(i,j)*bb*r4
      f2t=(1.-m2)*rho(i,j)/bb+r2*vp*(rho(i,j+1)-rho(i,j-1))/(2.*dv)
      f2(i,j)=f2t/vp
      f3(i,j)=mm*ch(i,j)*r4*bb
      f4(i,j)=(1.-m2)*mm*ch(i,j)/bb
      f5(i,j)=-m2*yep(i,j)*r4*bb
      f7(i,j)=(1.-m2)*(-m2)*yep(i,j)/(bb*vp)
      g4(i,j)=qub(i,j)*r(i,j)**2
      g3(i,j)=r(i,j)*b(i,j)
      g2(i,j)=qub(i,j)/(r(i,j)*b(i,j))**2
      g1(i,j)=+(mm*uuz(i))**2*r(i,j)*(r(i+1,j)+r(i-1,j)-2*r(i,j))
     c*qv(i,j) /du**2


 10   continue
c..... fill in edge values
      do 20 i=1,ix
      f1(i,1)=-f1(i,2)
      f2(i,1)=f2(i,2)
      f3(i,1)=-f3(i,2)
      f4(i,1)=f4(i,2)
      f5(i,1)=-f5(i,2)
      f7(i,1)=f7(i,2)
      g4(i,1)=-g4(i,2)
 20   continue
      return
      end
```

```
      call pscale(0,rplot(1),rplot(jx-2),ymin1,ymax1,n4)
  240  continue
      call pcurve(0 rplot(1),xrsppsi(2,npl),n3,n4,it,kx,iy,1h$,lab)
  220 continue
      call pframe
c..... time plots of xrtime
      kx='t$'
      iy(1)='xr(kplot'
      iy(2)=')$'
      it(1)='xr(kplot'
      it(2)=') vs t$'
      call pcurve(0,time,xrtime,nmax,12,it,kx iy,1h$,1h$ )
      do 70 ll=1,nmax
  70  dum1(ll)=abs(xrtime(ll))
      call pcurve(2,time,dum1,nmax,34,1h$,1h$,1h$,1h$,1h$)
      call pclose
      return
      end
```

```
401
402
403         subroutine amat
404
405 c..... calculates the matrix coefficients for a1, a2, a3, b1, b2
406 c..... in the equation a1*x(n+1)=a2*x(n)+a3*x(n-1)+b1*y(n)+b2*y(n-1) .
407 c..... uses f1 to f11 from subroutine f1to11 and equilibrium quantities.
408
409 c..... cliche storage here
410
411         use param
412         use fstor
413         use matrix
414         use const
415
416         data unit/1./
417
418         gam3=-gam2
419         du2=du**2
420         dt2=dt**2
421         dv2=dv**2
422         dvt=2.*dv
423         m2=mm**2.
424         jx=jrx
425         lx=lzx
426         do 10 i=2,lx-1
427         do 10 j=2,jx-1
428         k1=i-1+(j-2)*(lx-2)
429         k2=j-1+(jx-2)*(i-2)
430         k=.5*(1+isw)*k1+.5*(1-isw)*k2
431         r2=r(i,j)**2
432         vp=vpsi(j)
433         uz=uuz(i)
434         bijmh=(b(i,j)+b(i,j-1))*.5
435         bijph=(b(i,j)+b(i,j+1))*.5
436         bip1jph=(b(i+1,j+1)+b(i+1,j))*.5
437         bip1jmh=(b(i+1,j-1)+b(i+1,j))*.5
438         bim1jph=(b(i-1,j+1)+b(i-1,j))*.5
439         bim1jmh=(b(i-1,j-1)+b(i-1,j))*.5
440         g4iphjph=(g4(i+1,j+1)+g4(i,j))*.5*uuzh(i)
441         g4iphjmh=(g4(i+1,j-1)+g4(i,j))*.5*uuzh(i)
442         g4imhjph=(g4(i-1,j+1)+g4(i-1,j))*.5*uuzh(i-1)
443         g4imhjmh=(g4(i-1,j-1)+g4(i-1,j))*.5*uuzh(i-1)
444         g2iphj=(g2(i+1,j)+g2(i,j))*.5*uuzh(i)
445         g2imhj=(g2(i-1,j)+g2(i,j))*.5*uuzh(i-1)
446         g3iphj=(g3(i+1,j)+g3(i,j))*.5*uuzh(i)
447         g3imhj=(g3(i-1,j)+g3(i,j))*.5*uuzh(i-1)
448
449         f1ijph=(f1(i,j)+f1(i,j+1))*.5*vpsih(j)
450         f1ijmh=(f1(i,j)+f1(i,j-1))*.5*vpsih(j-1)
451         f5ijph=(f5(i,j)+f5(i,j+1))*.5*vpsih(j)
452         f5ijmh=(f5(i,j)+f5(i,j-1))*.5*vpsih(j-1)
453
454         if(j.gt.2)go to 60
455         f1ijmh=0.
456         f6ijmh=0.
457   60    continue
458         uzbar=-uuz(i)*r(i,j)/(du2*dv2)
459         a1(k,1)=-gam1*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
460        c *r(i-1,j-1)
```

```
1018
1019        subroutine diagno
1020 c.....sets up arrays for spatial plots
1021
1022 c..... insert cliches for common here
1023        use param
1024        use fstor
1025        use matrix
1026        use const
1027
1028        np=np+1
1029        do 10 i=2,ix-1
1030        kp1=i-1+(jx/2-1)*(ix-2)
1031        kp2=jx/2-1+(jx-2)*(i-2)
1032        kpp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
1033        xrspz(i,np)=xro(kpp)
1034   10   continue
1035        do 20 ll=1,2
1036        ixpl=ix/2**ll
1037        do 20 j=2,jx-1
1038        kp1=ixpl-1+(j-2)*(ix-2)
1039        kp2=j-1+(jx-2)*(ixpl-2)
1040        kpp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
1041        xrsppsi(j,np+(ll-1)*nps)=xro(kpp)
1042        xrsppsi(j,np+(ll-1)*nps)=xro(kpp)
1043   20   continue
1044        npm=np
1045        return
1046        end
```

```
460         c *r(i-1,j-1)
461          a2(k,1)=-gam2*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
462         c *r(i-1,j-1)
463          a3(k,1)=-gam3*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
464         c *r(i-1,j-1)
465
466          a1(k,2)=-f1ijmh/((dt*dv)**2)+gam1*(f5ijmh/dv2+bijmh**2*uzbar
467         c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
468          a2(k,2)=-f1ijmh/((dt*dv)**2)+gam2*(f5ijmh/dv2+bijmh**2*uzbar
469         c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
470          a3(k,2)=-f1ijmh/((dt*dv)**2)+gam3*(f5ijmh/dv2+bijmh**2*uzbar
471         c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
472
473          a1(k,3)=-gam1*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
474          a2(k,3)=-gam2*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
475          a3(k,3)=-gam3*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
476          a1(k,4)=gam1*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
477         c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
478         c g2imhj*g3(i-1,j)*dv2/vpsi(j))
479          a2(k,4)=gam2*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
480         c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
481         c g2imhj*g3(i-1,j)*dv2/vpsi(j))
482          a3(k,4)=gam3*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
483         c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
484         c g2imhj*g3(i-1,j)*dv2/vpsi(j))
485
486
487          a1(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam1*(-(f5ijph+f5ijmh
488         c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
489         c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
490         c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
491          a2(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam2*(-(f5ijph+f5ijmh
492         c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
493         c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
494         c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
495          a3(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam3*(-(f5ijph+f5ijmh
496         c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
497         c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
498         c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
499
500          a1(k,6)=gam1*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
501         c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
502         c g2iphj*g3(i+1,j)*dv2/vpsi(j))
503          a2(k,6)=gam2*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
504         c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
505         c g2iphj*g3(i+1,j)*dv2/vpsi(j))
506          a3(k,6)=gam3*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
507         c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
508         c g2iphj*g3(i+1,j)*dv2/vpsi(j))
509
510          a1(k,7)=gam1*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
511          a2(k,7)=gam2*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
512          a3(k,7)=gam3*(-bimijph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
513          a1(k,8)=-f1ijph/(dt2*dv2)+gam1*(f5ijph/dv2+(bijph**2*(g4imhjph
514         c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
515          a2(k,8)=-f1ijph/(dt2*dv2)+gam2*(f5ijph/dv2+(bijph**2*(g4imhjph
516         c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
517          a3(k,8)=-f1ijph/(dt2*dv2)+gam3*(f5ijph/dv2+(bijph**2*(g4imhjph
518         c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
519          a1(k,9)=gam1*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
520          a2(k,9)=gam2*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
```

```
521         a3(k,9)=gam3*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
522  c.....b1 array for rhs
523           f3ijmh=(f3(i,j)+f3(i,j-1))*.5*vpsih(j-1)
524         f3ijph=(f3(i,j)+f3(i,j+1))*.5*vpsih(j)
525         denom=1./(dv2)
526         b1(k,1)=f3ijmh*denom
527         b1(k,3)=f3ijph*denom
528         b1(k,2)=-(f3ijmh+f3ijph-f4(i,j)*dv2/vpl*denom
529   10    continue
530  c.....correct coefficients on boundaries
531         sfi1=sign(unit,fi1)
532         sfj1=sign(unit,fj1)
533         sfjrx=sign(unit,fjrx)
534         sfizx=sign(unit,fizx)
535  c.....  set corners to 0
536         k1i=ix-2
537         k1j=1+(ix-2)*(jx-3)
538         k1=.5*(1+isw)*k1i+.5*(1-isw)*k1j
539         fac1=-1.
540         if(sfj1.eq.1.and.sfizx.eq.1)fac1=1.
541         a1(k1,5)=a1(k1,5)+fac1*a1(k1,3)
542         a2(k1,5)=a2(k1,5)+fac1*a2(k1,3)
543         a3(k1,5)=a3(k1,5)+fac1*a3(k1,3)
544         a1(k1,3)=0.
545         a2(k1,3)=0.
546         a3(k1,3)=0.
547         k2i=1+(ix-2)*(jx-3)
548         k2j=jx-2
549         k2=.5*(1+isw)*k2i+.5*(1-isw)*k2j
550         fac3=-1.
551         if(sfjrx.eq.1.and.sfi1.eq.1)fac3=1.
552          a1(k2,5)=a1(k2,5)+fac3*a1(k2,7)
553          a2(k2,5)=a2(k2,5)+fac3*a2(k2,7)
554          a3(k2,5)=a3(k2,5)+fac3*a3(k2,7)
555         a1(k2,7)=0.
556         a2(k2,7)=0.
557         a3(k2,7)=0.
558         fac2=-1.
559         if(sfj1.eq.1.and.sfi1.eq.1)fac2=1.
560         a1(1,5)=a1(1,5)+fac2*a1(1,1)
561         a2(1,5)=a2(1,5)+fac2*a2(1,1)
562         a3(1,5)=a3(1,5)+fac2*a3(1,1)
563         a1(1,1)=0.
564         a2(1,1)=0.
565         a3(1,1)=0.
566         fac4=-1.
567         if(sfjrx.eq.1.and.sfizx.eq.1)fac4=1.
568         a1(kxp,5)=a1(kxp,5)+fac4*a1(kxp,9)
569         a2(kxp,5)=a2(kxp,5)+fac4*a2(kxp,9)
570         a3(kxp,5)=a3(kxp,5)+fac4*a3(kxp,9)
571         a1(kxp,9)=0.
572         a2(kxp,9)=0.
573         a3(kxp,9)=0.
574         i=2
575         do 11 j=2,jx-1
576         k1=i-1+(j-2)*(ix-2)
577         k2=j-1+(jx-2)*(i-2)
578         k=.5*(1+isw)*k1+.5*(1-isw)*k2
579         do 11 m=2,8,3
580         a1(k,m)=a1(k,m)+sfi1*a1(k,m-1)
```

```
      a2(k,m)=a2(k,m)+sfi1*a2(k,m-1)
      a3(k,m)=a3(k,m)+sfi1*a3(k,m-1)
11    continue
13    continue
      i=ix-1
      do 12 j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 12 m=2,8,3
      a1(k,m)=a1(k,m)+sfizx*a1(k,m+1)
      a2(k,m)=a2(k,m)+sfizx*a2(k,m+1)
      a3(k,m)=a3(k,m)+sfizx*a3(k,m+1)
12    continue
20    continue
      i=2
      do 21  j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 21  m=1,7,3
      a1(k,m)=0.
      a2(k,m)=0.
      a3(k,m)=0.
21    continue
      i=ix-1
      do 22 j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 22 m=3,9,3
      a1(k,m)=0.
      a2(k,m)=0.
      a3(k,m)=0.
22    continue
      j=2
      do 31 i=2,ix-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 30 m=4,6
      a1(k,m)=a1(k,m)+sfj1*a1(k,m-3)
      a2(k,m)=a2(k,m)+sfj1*a2(k,m-3)
      a3(k,m)=a3(k,m)+sfj1*a3(k,m-3)
30    continue
      b1(k,2)=b1(k,2)+b1(k,1)
31    continue
32    continue
      j=jx-1
      do 35 i=2,ix-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 34 m=4,6
      a1(k,m)=a1(k,m)+sfjrx*a1(k,m+3)
      a2(k,m)=a2(k,m)+sfjrx*a2(k,m+3)
      a3(k,m)=a3(k,m)+sfjrx*a3(k,m+3)
34    continue
      b1(k,2)=b1(k,2)+b1(k,3)
35    continue
```