```
      subroutine adummy
c..... cliche storage set up here

      cliche param
      parameter (izx=60, jrx=40, izx2=izx-2  )
      parameter ( itbw=2*izx-1  )
      parameter (kxp=(izx-2)*(jrx-2), ibw=izx-1 )
      parameter (nplt=3000, nps=100 )
      parameter (ixpf=4*(izx-2),nfourxf=150)
      parameter (ixpg=2*izx2)
       parameter (neng=500)
      parameter (ksim=51)
      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)
     c ,swg1,swg2,swg3,swg4

      common/equil/b(izx,jrx),rho(izx,jrx),qub(izx,jrx),r(izx,jrx)
     c ,phi(izx,jrx),yyy(izx,jrx),xxx(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 ww(1), ww1(1)
      endcliche

      cliche const
      use param
      common/title/aname(5)
      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 kplot,npm,fpsi,fz,fu,fv,azm,apsim,u0,v0,amass,nengx
     c ,fourpi,omegst,omegr,omegexb,flr,sf6,sf8,kplotm,kzs,zedge
     c ,cpuo,cio,syso,valfk,xu,xv,n,pi,vw,psiw,dvin,dvout,itt,nen,lee
      common/contm/
     c psi0rel,psi1rel,psi2rel,z1rel,z2rel,z3rel,z0rel,nslosh,bmg
     c ,ncenter,pslosh,pcenter,rp,ztrans,itrans,bm,itran,ztran,rp1
     c ,bcen,pring,epsp,phicen,phiplg,kin,xpot,ypot,wpot,pfudge,rpx
     c ,phice,phipl,betslsh,betcent,z0,z1,z2,z3,z4,psi1,psi2,psi0
     c ,betcene,betslse,psloshe,pcentee,bmax,alsi,bm1,psis1,cold
     c ,p2wide,psi3rel,psi3,p1max,bv0,bv3,bv4,bceng,psloshin,psloshen
     c ,nsloshin,pxp1,pxp2,p3a,p3b,p3c,p3d,psim,pe10,ae1,be1,ce1,de1
     c ,psi0erel,psi0e,psime,p2ewide,wp2e,p2floor,p1floor,p2flag
     c ,fring,long,no3d,no1d,dphi,dip,psistr,psislp,psihrel,psih
     c ,dpsihrel,dpsih,er,rcwall,exrho
       common/pcons/at0,bt0,ct0,cp0
     c ,ap1,ap2,ap3,at1,at2,at3,bp1,bp2,bp3,bt1,bt2,bt3,cp1,cp2,cp3
     c ,ct1,ct2,ct3,bmx1,bmx2,bmx3,bmn1,bmn2,ppas1,ppas2,ppas3,p1trap
     c ,z1c,z2c,z3c,z1min,z2min,as(3),als(3),zs(3),bs(3),dpas1,d1trap
     c ,betrap,betp s1,bvx2,bvx3,ncoil,z2ct
      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),xflute(izx,nps),enpot(neng),tenergy(neng)
      x ,enkin(neng),timengy(neng),tenrel(neng),enbend(neng)
      c ,encurve(neng),enflr(neng)
       common/curvco/cr,lb,rw,beta0,delrho,stable,en0,cee,r0,
      c echarg,omeg1,omeg2,en1,besarg,zol,dtrel,p0,omeg0
      c ,omana1,omana2,groana,theta0
       common/tmcon/h12(izx),hzt0(izx),h34(izx),abp(izx),bbp(izx)
      c ,cbp(izx),abf(izx),bbf(izx),cbf(izx),hp3(jrx),hp12(jrx)
      c ,htrans(izx),abq(izx),bbq(izx),cbq(izx),ebp(izx)
      c ,fbp(izx),gbp(izx),betring,hp0(jrx),hpm(jrx),hpme(jrx),hflr(jrx)
      c ,hzp0(izx),hzp1(izx),hzp2(izx),hzp3(izx),hzt1(izx),hzt2(izx)
      c ,hzt3(izx)
        common/tmfield/bvac(izx),dbvdz(izx),d2bvdz2(izx),dp1dpsi(jrx)
      c ,p1(jrx),p1k(ksim,jrx),hpk12(ksim),hpk0(ksim),deli1(ksim)
      c ,deli2(ksim),deli3(ksim),deli4(ksim),rzz(izx,jrx),dbdpsi(izx,jrx)
      c   ,phi1(izx),phi2(izx),pperp(izx,jrx),ppar(izx,jrx),dflute3(izx)
      c ,qubv(izx,jrx),p2(jrx),dp2dpsi(jrx),dflute1(izx),dflute2(izx)
      c ,flute1(jrx),flute2(jrx),flute3(jrx),p2k(ksim,jrx),deli5(ksim)
      c ,pperps(izx,jrx),errprp(izx,jrx),errprl(izx2,jrx-1)
      c ,pperpe(izx,jrx),epsi(izx,jrx),omeg1wkb(jrx),omeg2wkb(jrx)
      c ,gamwkb(jrx),dflute4(izx),rhoave(jrx),xxxave(jrx),yyyave(jrx)
      c ,p2t(jrx),dp2dpst(jrx),p3(jrx),dp3dpsi(jrx),hpkm(ksim)
      c ,hpkme(ksim),droave(jrx),droterm(izx),ering(izx,jrx)
       common/forzed/nfour,nfourx,nfourmax,nfourp,jfour,ixp,iocv

       real lb,ltrans,ltran,nslosh,ncenter,nsloshin
       endcliche

       return
       end

c..... the main routine

c ************************************************************************
c *                                                                      *
c   FLORA is an initial value stability code developed by R. Freis and
c   B. Cohen, based on Newcomb's long thin axisymmetric formalism, including
c   finite larmor radius effects. FLORA calculates the linear response to
c   low frequency perturbations of the equilibrium magnetic field .

c*************************************************************************
c..... notice of 4/8/82. this version runs correctly for isw=1, and
c..... runs correctly for isw=-1 .

c.....5/12/82. flora runs testcase 1 , 0 beta, 0 pressure, homogeneous
c..... plasma, correctly.

c..... flora1 transforms variables z,psi to u,v which are always equally
c..... spaced. transformation: z=au*u**xu, and psi=apsi*v**xv, where
c..... zmax=umax, psimax=vmax, and fz*zmax=fu*umax, fpsi*psimax=fv*vmax .
c..... fz, fu, fpsi, fv, input, xu=ln fz / ln fu, xv=ln fpsi / ln fv .
c..... au=umax**(-xx+1) , apsi=vmax**(-yy+1) .

c..... flora2 solves test case 2 , rotating rigid rotor stability. ref:
c..... freidberg and pearlstein, phys fluids 21(7) july 1978 1207


c.....flora4 includes background constant density, enbar ( as does flora3 ),
c..... and kzs switch which when set to zero, generates initial perturbations
c..... independent of z in random spatial generator (ex0=1.) .


c..... flora5
c.....  is vectorized version of flora4.(calls rightvec instead of
c..... right ), also has timing routine from b. langdon (requires
c..... bzohar loaded as a binary ).
c...... insert cliche storage here


c..... flora7 is mod. flora5, with psi stretching function
c..... exactly centered in amat. (flora5 used linear interpolation
c..... to get vpsi(j+1/2)). Also revised diagnostic plots included.

c.....flora12 is flora11 (rigid rotor with corrected equil. and
c..... corrected curvature terms (flora10!) with fourier mode analyses
c..... added ( using cpft and rpft) and data for zed post processing.
c.....additional input data: jfour (v index at which xr is analyzed in
c.....z ), nfourp ( analyzo xr every nfour'th time step ),nfourmax
c..... (number of times the buffer is read to the history file), note
c..... xr is extended a factor of 4 to look like a periodic full wave
c..... for cpft. If jfour is input 0, code sets it to jx/4 .
c
c.....flora13 is flora12 with curvature driven flute mode equilibrium
c.....(equilrot replaced by equilcur, rigidcon replaced by curvecon )

c.....floratm, tandem mirror equilibrium
c
c.....flortm1, tandem mirror equilibrium, with 3-d plot of equilib.
c     quantities added. ( uses tv80 and graflib )

c......flortex, tandem mirror equilb. with corrections to flortm1. In-
c...... put switches swg1, swg2, swg3, swg4 added.

c.....flortm2, like flortex with revised electron ring, a la D'ippolito
c.....(e-ring pperp in b field only, and additional term in curvature
c..... drive ),

c.....flortm3, like flortm2 with corrections to pressusre normalization,
c..... and additional diagnostics. ( 3-d plots of curvature drive-e ring
c..... term, and perp. pressure balance check ) , also 3-d plots of
c..... pparallel pressure check, and e-psi (=-dphi/dpsi ) . Phi2 modified
c..... to [illegible],-(psi/psi3)**ypot ,

c.....flortm4, modified plasma pperp with addition of p3(j) to give
c..... a positive slope near the center,

c.....flortm5, modified p1 in flortm4 to be two functions, pe1 and
c..... pe2, joined at psime with equal slope and value,pe1=ae1+be1*(
c..... psi/psime)+ce1*(psi/psime)**2, and pe2=.5*(1-tanh((psi-psi0e)/p2ewide))
c
c..... flortm6, modified flortm5 as follows: for p2(psi)le. to p2flag ( an
c..... input value), p2 set to p2floor (an input value) and p1 set to
c..... p1floor (an input value). Long-thin ering option added. This modifies
c..... b dependence of ering pperp to look longer (by changing abf, bbf, cbf)
c..... if long ( an input value ) = 1, otherwise leaves pperp of ring un-
c..... changed. Plot output options, no1d=1. prevents graflib plots, no3d=1
c..... prevents tv80lib 3d plots.
c
```

```
c,,,,,, flortm8 (l8 for larger psi grid) modified flortm6 as follows;
c,,,,,, the analytic calculation of gamwkb corrected to include drho/dpsi
c,,,,,, term (important in the limit of large exb rotation). also phi1
c,,,,,, changed to be constants in core and plug ( phicen in core, and
c,,,,,, phicen+dphi in plug, dphi a new input variable ). Also b calculated
c,,,,,, with expansion in low beta regions.

c,,,,,, flortn8, like flortl8 with first order energy check added .


c,,,,,,flrm1 like flortn8 with multi region equilibrium.
c,,,,,, Bvac is generated from 3 solonoids ( 1 choke coil and 2
c,,,,,, mirror coils). Pressures are the sum of passing and trapped
c,,,,,, components. See the glossary of input parameters in subroutine
c,,,,,, inputtm for the revised list .

c,,,,,,flrm2, either 2 or 3 regions, depending on the number of solonoids
c,,,,,, specified (ncoil=2, or 3 ) in the input. For ncoil=2, no passing
c,,,,,, pressures are allowed, and the situation is similar to earlier
c,,,,,, versions, except that the vaccum fields are generated by solonoids
c,,,,,, instead of circular filaments.

c,,,,,, flrm3, like flrm2 with corrections to energy subroutine. Also
c,,,,,, option to remove hollowness from pperp psi profile (dip=0.), and
c,,,,,, ering psi profile changed from quadratic to cubic in inner region.

c,,,,,,flrm4, cold plasma halo modeled by changing zmax boundary conditions
c,,,,,, for psi > psih (psihrel an input parameter )

c,,,,,, flrm6, like flrm4 and flrm5, (mixed boundary condition at zmax,
c,,,,,, higher order b. c.) with special yyy to force "rigid mode" in psi.

c,,,,,, flrrot, modified flrm6 to study rotational stability with pos-
c,,,,,, itive density gradients. Set phi2(psi)=er (a constant), add
c,,,,,, rhoc = a + b*psi to rho .
      use param
      use fstor
      use matrix
      use const

      data tim/1.e6/
      integer tallyb(2000b)
      common / q8locs/locf(0:15)
      common/pic100/npete
      data itally/1/

c,,,,,,,call link call here
      call link('unit59=terminal,unit2=(inflm4,open),unit3=(output,
     c create) //')

      if(itally.gt.0) then
      do 200 ii=1,15
 200  if(locf(ii).eq.0)go to 210
      ii=0
 210  loctally=ii
      loctally=14
      if(loctally.eq.0)go to 299
      call timer(loctally,'ztally00',tallyb,2000b,floratim,1)
 299  itally=-1
      endif
      isw=1
      if(izx.gt.jrx)isw=-1
      jtbw=.5*(1+isw)*itbw+.5*(1-isw)*(2*jrx-1)
      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)
      namelist/noplot/no1d,no3d
      call ddi(noplot,2,0,1)
      if(no1d.ne.1)call pstart(dev,4rplot,1,'box u21$',1)
c     npete=1
      if(no1d.ne.1)call p100
      call input
      call inputtm
c......temporary input to test three region model
c     call inptemp
c     call rigidcon
c     call curvecon
      call constant
      call tmcon2
      call equiltm
c     call equilrot
c     call equilcur
      call f1to11
      call amat
      call comat(ww,jtbw)
      call initial
c.....special version for testing fourier analysis and zed file
c..... maker
      call fourplay
      call fourier
      call mymove(xroi(1),xro(1),kxx)
      call mymove(xioi(1),xio(1),kxx)
c     call mymove(xroo(1),xro(1),kxx)
c     call mymove(xioo(1),xio(1),kxx)
      call banfac(kxp,ihbw,ww,1,-(kxp-1))
       call energy
      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(xroi,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(xioi,ww1(nn),kxx)
 20   continue
      fac1=-.5/dt
 90   fac2=.5/dt
```

```
      call zmovewrd(xloo,xlo,kxx)
      call zmovewrd(xlo,xlol,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
       if(mod(n,nfourp).eq.0)call fourier
      if(((mod(n,nen)).eq.0).or.((tt.ne.0).and.(lee.le.neng))call energy
 100  continue
      call clsdsk(iocv,0)
      call timeused(icp,io,isy)
      cpuo=icp*tim
      cio=io*tim
      syso=isy*tim
      if(nold.ne.1)
     c call picsher
      call close(100)
      if(no3d.eq.1)go to 300
      call keep80(1,3)
      call fr80id
      call threed
      call plote
 300  continue
      call timend
      call exit(1)
      end
      subroutine amat

c..... calculates the matrix coefficients for a1, a2, a3, b1, b2
c..... in the equation a1*x(n+1)=a2*x(n)+a3*x(n-1)+b1*y(n)+b2*y(n-1) .
c..... uses f1 to f11 from subroutine f1to11 and equilibrium quantities.

c..... cliche storage here

      use param
      use fstor
      use matrix
      use const

      dimension bc(jrx),delco1(jrx),delco2(jrx)
      data unit/1./

      gam3=-gam2
      du2=du**2
      dt2=dt**2
      dv2=dv**2
      dvt=2.*dv
      m2=mm**2
      jx=jrx
      ix=izx
      do 10 i=2,ix-1
      do 10 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
      r2=r(i,j)**2
      vp=vpsi(j)
      uz=uuz(i)
      bijmh=(b(i,j)+b(i,j-1))*.5
      bijph=(b(i,j)+b(i,j+1))*.5
      bip1jph=(b(i+1,j+1)+b(i+1,j))*.5
      bip1jmh=(b(i+1,j-1)+b(i+1,j))*.5
      bim1jph=(b(i-1,j+1)+b(i-1,j))*.5
      bim1jmh=(b(i-1,j-1)+b(i-1,j))*.5
      g4iphjph=(g4(i+1,j+1)+g4(i,j)+g4(i+1,j)+g4(i,j+1))*.25*uuzh(i)
      g4iphjmh=(g4(i+1,j-1)+g4(i,j)+g4(i+1,j)+g4(i,j-1))*.25*uuzh(i)
      g4imhjph=(g4(i-1,j+1)+g4(i,j)+g4(i-1,j)+g4(i,j+1))*.25*uuzh(i)
      g4imhjmh=(g4(i-1,j-1)+g4(i,j)+g4(i-1,j)+g4(i,j-1))*.25*uuzh(i)
      g2iphj=(g2(i+1,j)+g2(i,j))*.5*uuzh(i)
      g2imhj=(g2(i-1,j)+g2(i,j))*.5*uuzh(i-1)
      g3iphj=(g3(i+1,j)+g3(i,j))*.5*uuzh(i)
      g3imhj=(g3(i-1,j)+g3(i,j))*.5*uuzh(i-1)

      f1ijph=(f1(i,j)+f1(i,j+1))*.5*vpsih(j)
      f1ijmh=(f1(i,j)+f1(i,j-1))*.5*vpsih(j-1)
      f5ijph=(f5(i,j)+f5(i,j+1))*.5*vpsih(j)
      f5ijmh=(f5(i,j)+f5(i,j-1))*.5*vpsih(j-1)

      if(j.gt.2)go to 60
      f1ijmh=0.
      f6ijmh=0.
60    continue
      uzbar=-uuz(i)*r(i,j)/(du2*dv2)
      a1(k,1)=-gam1*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)
      a2(k,1)=-gam2*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)
      a3(k,1)=-gam3*bim1jmh*g4imhjmh*bijmh*uzbar*vpsih(j-1)
     c *r(i-1,j-1)

      a1(k,2)=-f1ijmh/((dt*dv)**2)+gam1*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
      a2(k,2)=-f1ijmh/((dt*dv)**2)+gam2*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))
      a3(k,2)=-f1ijmh/((dt*dv)**2)+gam3*(f5ijmh/dv2+bijmh**2*uzbar
     c *vpsih(j-1)*r(i,j-1)*(g4imhjmh+g4iphjmh))

      a1(k,3)=-gam1*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a2(k,3)=-gam2*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a3(k,3)=-gam3*bip1jmh*g4iphjmh*bijmh*uzbar*vpsih(j-1)*r(i+1,j-1)
      a1(k,4)=gam1*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))
      a2(k,4)=gam2*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))
      a3(k,4)=gam3*((bim1jmh*g4imhjmh*vpsih(j-1)*bijmh+bim1jph*g4imhjph
     c *vpsih(j)*bijph)*uzbar*r(i-1,j)+mm**2*b(i,j)*uzbar*
     c g2imhj*g3(i-1,j)*dv2/vpsi(j))


      a1(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam1*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
      a2(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam2*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))
```

```
       a3(k,5)=((f1ijph+f1ijmh)/dv2-f2(i,j))/dt2+gam3*(-(f5ijph+f5ijmh
     c )/dv2+f7(i,j)+g1(i,j)+(-bijmh**2*(g4imhjmh+g4iphjmh)*vpsih(j-1)
     c -bijph**2*(g4imhjph+g4iphjph)*vpsih(j))*r(i,j)*uzbar-
     c mm**2*b(i,j)*uzbar*(g2imhj+g2iphj)*g3(i,j)*dv2/vpsi(j))

       a1(k,6)=gam1*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))
       a2(k,6)=gam2*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))
       a3(k,6)=gam3*((bip1jmh*g4iphjmh*bijmh*vpsih(j-1)+bip1jph*g4iphjph
     c *bijph*vpsih(j))*uzbar*r(i+1,j)+mm**2*b(i,j)*uzbar*
     c g2iphj*g3(i+1,j)*dv2/vpsi(j))

       a1(k,7)=gam1*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
       a2(k,7)=gam2*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
       a3(k,7)=gam3*(-bim1jph*g4imhjph*bijph*vpsih(j)*uzbar*r(i-1,j+1))
       a1(k,8)=-f1ijph/(dt2*dv2)+gam1*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
       a2(k,8)=-f1ijph/(dt2*dv2)+gam2*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
       a3(k,8)=-f1ijph/(dt2*dv2)+gam3*(f5ijph/dv2+(bijph**2*(g4imhjph
     c +g4iphjph)*vpsih(j)*r(i,j+1)*uzbar))
       a1(k,9)=gam1*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
       a2(k,9)=gam2*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
       a3(k,9)=gam3*(-bip1jph*g4iphjph*bijph*vpsih(j)*uzbar*r(i+1,j+1))
c.....,b1 array for rhs
        f3ijmh=(f3(i,j)+f3(i,j-1))*.5*vpsih(j-1)
       f3ijph=(f3(i,j)+f3(i,j+1))*.5*vpsih(j)
       denom=1./(dv2)
       b1(k,1)=f3ijmh*denom
       b1(k,3)=f3ijph*denom
       b1(k,2)=-(f3ijmh+f3ijph-f4(i,j)*dv2/vp)*denom
 10    continue
c.....,correct coefficients on boundaries
       sfi1=sign(unit,fi1)
       sfj1=sign(unit,fj1)
       sfjrx=sign(unit,fjrx)
       sfizx=sign(unit,fizx)
c..... set corners to 0
       k1i=ix-2
       k1j=1+(ix-3)*(jx-2)
       k1=.5*(1+isw)*k1i+.5*(1-isw)*k1j
       fac1=-1.
       if(sfj1.eq.1.and.sfizx.eq.1)fac1=r(ix-1,2)*b(ix-1,2)/
     c (r(ix,2)*b(ix,2))
       a1(k1,5)=a1(k1,5)+fac1*a1(k1,3)
       a2(k1,5)=a2(k1,5)+fac1*a2(k1,3)
       a3(k1,5)=a3(k1,5)+fac1*a3(k1,3)
       a1(k1,3)=0.
       a2(k1,3)=0.
       a3(k1,3)=0.
       k2i=1+(ix-2)*(jx-3)
       k2j=jx-2
       k2=.5*(1+isw)*k2i+.5*(1-isw)*k2j
       fac3=-dvout/dvin
       if(sfjrx.eq.1.and.sfi1.eq.1)fac3=r(2,jx-1)*b(2,jx-1)/
     c (r(1,jx-1)*b(1,jx-1))
         a1(k2,5)=a1(k2,5)+fac3*a1(k2,7)
         a2(k2,5)=a2(k2,5)+fac3*a2(k2,7)
         a3(k2,5)=a3(k2,5)+fac3*a3(k2,7)
        a1(k2,7)=0.
        a2(k2,7)=0.
        a3(k2,7)=0.
        fac2=-1.
        if(sfj1.eq.1.and.sfi1.eq.1)fac2=r(2,2)*b(2,2)/(r(1,2)*b(1,2))
        a1(1,5)=a1(1,5)+fac2*a1(1,1)
        a2(1,5)=a2(1,5)+fac2*a2(1,1)
        a3(1,5)=a3(1,5)+fac2*a3(1,1)
        a1(1,1)=0.
        a2(1,1)=0.
        a3(1,1)=0.
        fac4=-dvout/dvin
c       if(sfjrx.eq.1.and.sfizx.eq.1)fac4=r(ix-1,jx-1)*b(ix-1,jx-1)/
c      c (r(ix,jx-1)*b(ix,jx-1))
        a1(kxp,5)=a1(kxp,5)+fac4*a1(kxp,9)
        a2(kxp,5)=a2(kxp,5)+fac4*a2(kxp,9)
        a3(kxp,5)=a3(kxp,5)+fac4*a3(kxp,9)
        a1(kxp,9)=0.
        a2(kxp,9)=0.
        a3(kxp,9)=0.
        i=2
        do 11 j=2,jx-1
        if(sfi1.eq.1.)sfi1=r(2,j)*b(2,j)/(r(1,j)*b(1,j))
        k1=i-1+(j-2)*(ix-2)
        k2=j-1+(jx-2)*(i-2)
        k=.5*(1+isw)*k1+.5*(1-isw)*k2
        do 11 m=2,8,3
        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
c       bc(j)=0.
c       if(psi(j).lt.(psih-dpsih))bc(j)=1.
c       if(psi(j).ge.(psih-dpsih).and.psi(j).le.(psih+dpsih))
        bc(j)=.5*(tanh((-psi(j)+psih)/dpsih)+1)
        rbt=r(ix-1,j)*b(ix-1,j)
        rbt1=r(ix,j)*b(ix,j)
        down2=8.*bc(j)*uuzh(ix-1)/du+3.*(1.-bc(j))
        up2=1.-bc(j)
        gm2=up2/down2
        up1=-up2*rbt1*.75+bc(j)*rbt*uuzh(ix-1)/du
        down1=(bc(j)*uuzh(ix-1)/du+up2*3./8.)*rbt1
        gm1=up1/down1
        delco1(j)=gm1
        delco2(j)=gm2
        up=(bc(j)*r(ix-1,j)*b(ix-1,j)*uuzh(ix-1)/du+rbt*(bc(j)-1.))
        down=(bc(j)*r(ix,j)*b(ix,j)*uuzh(ix-1)/du+rbt*(-bc(j)+1.))
        sfizx=up/down
        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)+gm1*a1(k,m+1)
        a1(k,m-1)=a1(k,m-1)+gm2*a1(k,m+1)
        a2(k,m)=a2(k,m)+gm1*a2(k,m+1)
```

```
      a2(k,m-1)=a2(k,m-1)+gm2*a2(k,m+1)
      a3(k,m)=a3(k,m)+gm1*a3(k,m+1)
      a3(k,m-1)=a3(k,m-1)+gm2*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)+sfj1*b1(k,1)
31    continue
32    continue
      j=jx-1
      fac5=sfjrx
      if(sfjrx.eq.-1)fac5=fac5*dvout/dvin
      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)+fac5*a1(k,m+3)
      a2(k,m)=a2(k,m)+fac5*a2(k,m+3)
      a3(k,m)=a3(k,m)+fac5*a3(k,m+3)
34    continue
      b1(k,2)=b1(k,2)+fac5*b1(k,3)
35    continue
40    continue
      j=2
      do 45 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 44 m=4,6
      a1(k,m-3)=0.
      a1(k,m-3)=0.
```

```
      a2(k,m-3)=0.
      a3(k,m-3)=0.
   44 continue
      b1(k,1)=0.
   45 continue
      j=jx-1
      do 48 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 47 m=4,6
      a1(k,m+3)=0.
      a3(k,m+3)=0.
      a2(k,m+3)=0.
   47 continue
      b1(k,3)=0.
   48 continue
      return
      end

c***********************************************************************
      subroutine bvcal(bs,zs,as,als,z,n,zcc,bcc,znorm,bv,bvp,bvpp,
     1   b4,b5,z4,z5,nc)
      dimension bs(1),zs(1),as(1),als(1),z(1),bv(1),bvp(1),bvpp(1)
      call bcccal(bs,zs,as,als,z,n,bv,bvp,bvpp,b4,b5,z4,z5,nc)
      do 15 i=1,n
      bcorr=bcc
      bcorrp=0.
      bcorrpp=0.
      tanhyp=tanh((z(i)-zcc)/znorm)
       bcorrp=-bcc*.5*(1.- tanhyp**2)/znorm
       bcorrpp=+bcc*tanhyp*(1.-tanhyp**2)/znorm**2
      bcorr=bcc*.5*(1.-tanhyp)
 18   bv(i)=bv(i)+bcorr
      bvp(i)=bvp(i)+bcorrp
      bvpp(i)=bvpp(i)+bcorrpp
      if(z(i).eq.z4)b4=b4+bcorr
      if(z(i).eq.z5)b5=b5+bcorr
 15   continue
      return
      end
c***********************************************************************
      subroutine bcccal(bs,zs,as,als,z,n,bv,bvp,bvpp,b4,b5,z4,z5,nc)
      dimension z(n),bv(n),bvp(n),bvpp(n)
      dimension bs(3),zs(3),as(3),als(3),alpha(3,3),ak(3),is(3)

15
c....compute matrix elements

      do 10 j=1,nc
      do 10 i=1,nc
      alpha(i,j)=bfun(zs(i),zs(j),als(j),as(j),0)
 10   continue

c....determinant
      if(nc.eq.2)alpha(3,3)=1.

      det=alpha(1,1)*(alpha(2,2)*alpha(3,3)-alpha(3,2)*alpha(2,3))
     1   -alpha(1,2)*(alpha(2,1)*alpha(3,3)-alpha(3,1)*alpha(2,3))
     2   +alpha(1,3)*(alpha(2,1)*alpha(3,2)-alpha(3,1)*alpha(2,2))
```

```
c....solution

      ak(1)=(bs(1)*(alpha(2,2)*alpha(3,3)-alpha(3,2)*alpha(2,3))
     1      -bs(2)*(alpha(1,2)*alpha(3,3)-alpha(3,2)*alpha(1,3))
     2      +bs(3)*(alpha(1,2)*alpha(2,3)-alpha(2,2)*alpha(1,3)))
     3      /det
      ak(2)=(-bs(1)*(alpha(2,1)*alpha(3,3)-alpha(3,1)*alpha(2,3))
     1       +bs(2)*(alpha(1,1)*alpha(3,3)-alpha(3,1)*alpha(1,3))
     2       -bs(3)*(alpha(1,1)*alpha(2,3)-alpha(2,1)*alpha(1,3)))
     3      /det
      ak(3)=(bs(1)*(alpha(1,2)*alpha(2,3)-alpha(1,3)*alpha(2,2))
     1      -bs(2)*(alpha(1,1)*alpha(2,3)-alpha(1,3)*alpha(2,1))
     2      +bs(3)*(alpha(1,1)*alpha(2,2)-alpha(1,2)*alpha(2,1)))
     3      /det

c....fields and derivatives

      do 50 i=1,n
      bv(i)=0.
      bvp(i)=0.
      bvpp(i)=0.
      do 50 j=1,nc
      bv(i)=bv(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),0)
      bvp(i)=bvp(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),1)
      bvpp(i)=bvpp(i)+ak(j)*bfun(z(i),zs(j),als(j),as(j),2)
 50   continue

c....minima and their positions

      do 70 j=1,nc
      do 60 i=2,n
      if(zs(j).lt.z(i)) go to 63
 60   continue
 63   is(j)=i-1
 70   continue
      b4=aminaf(bv,is(1),is(2),1,i4,amin)
      z4=z(i4)
      if(ncoil.eq.2)return
      b5=aminaf(bv,is(2),is(3),1,i5,amin)
      z5=z(i5)

      return
      end
c***********************************************************************
      function bfun(z,zx,al,a,ind)
      ind1=ind+1
      up=zx+.5*al
      um=zx-.5*al
      go to (10,20,30),ind1
 10   t1=gfun(z,up,a)
      t2=gfun(z,um,a)
      go to 40
 20   t1=gfun1(z,up,a)
      t2=gfun1(z,um,a)
      go to 40
 30   t1=gfun2(z,up,a)
      t2=gfun2(z,um,a)
 40   bfun=(t1-t2)/a**2
      return
      end
c*************************************************************************************
      function gfun(z,u,a)
      x1=u-z
      x2=sqrt(x1**2+a**2)
      gfun=x1/x2
      return
      end
c*********************************************************************************
      function gfun1(z,u,a)
      x1=u-z
      x2=(x1**2+a**2)**1.5
      gfun1=-a**2/x2
      return
      end
c***********************************************************************************
      function gfun2(z,u,a)
      x1=u-z
      x2=(x1**2+a**2)**2.5
      gfun2=-3.*x1*a**2/x2
      return
      end

      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=nd*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

      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 curvecon

c.... calculates constants necessary for curvature driven
c....  flute mode case.

c..... insert storage cliches here
      use param
      use const
      real klbsq

c..... input for curvature driven flute case
      data echarg/4.8e-10/, en0/1.00e+12/, b0/1.e4/, amass/3.34e-24/
     c , cee/3.e10/, stable/.4/, fourpi/12.56637/, pi/3.1415926/
     c , delrho/.05/,dtrel/.02/,xm/3.8317/, theta0/1.570796/
      namelist/curve/b0,beta0,delrho,stable,en0,echarg,ib,rw,xm
     c ,zol,dtrel,theta0
      call ddi(curve,2,3,1)
      if(nold.ne.1)     call ddo(curve,100,0,1)
      zmax=zol*lb
      p0=beta0*b0**2*.5
      psimax=rw**2*b0*.5
      omeg0sq=b0**2/(en0*amass*lb**2)
      omeg0=sqrt(omeg0sq)
      ag1=1.-2.*delrho/xm**2
      klbsq=0.
      if(kzs.ne.0)klbsq=(pi/(2.*zol))**2
      delomeg=0.
      if(sf8.ne.0)
     1 delomeg=stable*((beta0/xm**2-klbsq/mm**2)/(ag1*sf8)+1.-sf6*ag1/
     2 sf8)*omeg0sq
      omeg1=omeg0+sqrt(delomeg)
      omeg2=omeg1-2*sqrt(delomeg)
      dt=dtrel/omeg1
      en1=2.*en0*delrho/rw**2
      u(ix)=zmax
      v(jx)=psimax
      du=u(ix)/(ix-1.5)
      dv=v(jx)/(jx-1.5)
      besarg=(xm/rw)
c..... calculate analytic growth rate
      tsf6=tan(theta0*.5)*sf6
      radical=((ag1*mm*tsf6)**2*(omeg1+omeg2)**2-4.*((omeg1*omeg2*ag1
     c *sf8+omeg0sq*beta0/xm**2)*mm**2-klbsq*omeg0sq))
      if(radical.lt.0)go to 5
      root=sqrt(radical)
      omana1=ag1*tsf6*mm*(omeg1+omeg2)*.5+root*.5
      omana2=omana1-root
      groana=0.
      return
```

```
  5   continue
      omana1=ag1*tsf6*mm*(omeg1+omeg2)*.5
      groana=sqrt(-radical)*.5
      omana2=0.
      return
      end
       subroutine energy

c...   energy calculates the total first order energy (tenergy) as
c...   an integral (using trapazoidal quadrature) over the volume,
c...   every nen'th time step, using two consecutive time steps to
c...   determine the value at n+.5 .

      use param
      use fstor
      use matrix
      use const

       dimension rxi(jrx),rxr(jrx),drxr(izx,jrx),drxi(izx,jrx),quad(12)
     c ,quad1(4),psuma(jrx,12),psumb(jrx,4),dum1(izx),dum2(izx)
     c ,xrg(izx),xig(izx),xrgo(izx),xigo(izx),drxro(izx,jrx)
     c ,drxio(izx,jrx),dxrz2(izx),dxiz2(izx)

       itt=itt+1
       if((itt.eq.2).or.(nen.eq.1))then
      imove=jx*ix
        call zmovewrd(drxro,drxr,imove)
        call zmovewrd(drxio,drxi,imove)
      tquadoo=tquado
        tquado=tquad
      tquad2oo=tquad2o
      tquad2o=tquad2
      tquad3oo=tquad3o
      tquad3o=tquad3
      tquad4oo=tquad4o
      tquad4o=tquad4
        end if
       do 9 i=2,ix-1
        do 5 j=2,jx-1
         k1=i-1+(j-2)*(ix-2)
         k2=j-1+(jx-2)*(i-2)
         k=.5*((1+isw)*k1+(1-isw)*k2)
         rxr(j)=r(i,j)*xro(k)
         rxi(j)=r(i,j)*xio(k)
 5       continue
        call ddpsi(rxr,drxr,i)
        call ddpsi(rxi,drxi,i)
 9     continue

c     loop 100 evaluates z quadrature

        do 100 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+(1-isw)*k2)
     if((itt.eq.2).or.(nen.eq.1))then
     xrgo(i)=xroo(k)
     xigo(i)=xioo(k)
     end if
```

```
        xrg(i)=xro(k)
10      xig(i)=xio(k)
     do 211 i=2,ix-1
211  dum2(i)=r(i,j)*b(i,j)*xrg(i)
       call ddz(dum2,dxrz2)
     do 212 i=2,ix-1
212  dum2(i)=r(i,j)*b(i,j)*xig(i)
       call ddz(dum2,dxiz2)
      do 11 i=2,ix-1
11    dum1(i)=(1./uuz(i))*qub(i,j)*dxrz2(i)/((r(i,j)*b(i,j))**2)
      psuma(j,1)=psum(dum1,ix)/vpsi(j)
      do 12 i=2,ix-1
12    dum1(i)=(1./uuz(i))*qub(i,j)*dxiz2(i)/((r(i,j)*b(i,j))**2)
      psuma(j,2)=psum(dum1,ix)/vpsi(j)
      do 13 i=2,ix-1
13    dum1(i)=b(i,j)*drxi(i,j)
      call ddz(dum1,dum2)
      do 14 i=2,ix-1
14    dum1(i)=(1./uuz(i))*qub(i,j)*r(i,j)**2*dum2(i)
      psuma(j,3)=psum(dum1,ix)/vpsi(j)/mm**2
      do 15 i=2,ix-1
15    dum1(i)=b(i,j)*drxr(i,j)
      call ddz(dum1,dum2)
      do 16 i=2,ix-1
16    dum1(i)=(1./uuz(i))*qub(i,j)*r(i,j)**2*dum2(i)
      psuma(j,4)=psum(dum1,ix)/vpsi(j)/mm**2
      do 17 i=2,ix-1
17    dum1(i)=(1./uuz(i))*xrg(i)**2*r(i,j)*(qubv(i,j)*rzz(i,j)
    c +r(i,j)*ering(i,j)/b(i,j))
      psuma(j,5)=psum(dum1,ix)/vpsi(j)
      do 18 i=2,ix-1
18    dum1(i)=(1./uuz(i))*xig(i)**2*r(i j)*(qubv(i,j)*rzz(i,j)
    c +r(i,j)*ering(i,j)/b(i,j))
      psuma(j,6)=psum(dum1,ix)/vpsi(j)
      do 19 i=2,ix-1
19    dum1(i)=(1./uuz(i))*xrg(i)**2*yyy(i,j)/b(i,j)
      psuma(j,7)=mm**2*psum(dum1,ix)/vpsi(j)
      do 20 i=2,ix-1
20    dum1(i)=(1./uuz(i))*xig(i)**2*yyy(i,j)/b(i,j)
      psuma(j,8)=mm**2*psum(dum1,ix)/vpsi(j)
      do 21 i=2,ix-1
21    dum1(i)=(1./uuz(i))*yyy(i,j)*xrg(i)*r(i,j)*drxr(i,j)
      psuma(j,9)=-2.*psum(dum1,ix)/vpsi(j)
      do 22 i=2,ix-1
22    dum1(i)=(1./uuz(i))*yyy(i,j)*r(i,j)**2*b(i,j)*drxr(i,j)**2
      psuma(j,10)=psum(dum1,ix)/vpsi(j)
      do 23 i=2,ix-1
23    dum1(i)=-(1./uuz(i))*yyy(i,j)*xig(i)*r(i,j)*drxi(i,j)
      psuma(j,11)=2*psum(dum1,ix)/vpsi(j)
      do 24 i=2,ix-1
24    dum1(i)=(1./uuz(i))*yyy(i,j)*r(i,j)**2*b(i,j)*drxi(i,j)**2
      psuma(j,12)=psum(dum1,ix)/vpsi(j)
      if((itt.eq.2).or.(nen.eq.1))then
      do 30 i=2,ix-1
30    dum1(i)=(1./uuz(i))*rho(i,j)*((xrg(i)-xrgo(i))/dt)**2/b(i,j)
      psumb(j,1)=psum(dum1,ix)/vpsi(j)
      do 31 i=2,ix-1
31    dum1(i)=(1./uuz(i))*rho(i,j)*((xig(i)-xigo(i))/dt)**2/b(i,j)
      psumb(j,2)=psum(dum1,ix)/vpsi(j)
      do 32 i=2,ix-1
```

```
32    dum1(i)=(1./uuz(i))*(r(i,j)*(drxr(i,j)-drxro(i,j))/dt)**2*b(i,j)
     c *rho(i,j)
      psumb(j,3)=psum(dum1,ix)/vpsi(j)/mm**2
      do 33 i=2,ix-1
33    dum1(i)=(1./uuz(i))*(r(i,j)*(drxi(i,j)-drxio(i,j))/dt)**2*b(i,j)
     c *rho(i,j)
      psumb(j,4)=psum(dum1,ix)/vpsi(j)/mm**2
      end if
100   continue
       do 120 l=1,12
120    quad(l)=psum(psuma(1,l),jx)
       tquad=ssum(12,quad,1)*dv*du
     tquad2=0.
     do 121 l=1,4
121  tquad2=tquad2+quad(l)
     tquad3=quad(5)+quad(6)
     tquad4=0.
     do 122 l=7,12
122  tquad4=quad(l)+tquad4
       if((ltt.eq.2).or.(nen.eq.1))then
       tquad1=0.
       lee=lee+1
       do 130 l=1,4
       quad1(l)=psum(psumb(1,l),jx)
130    tquad1=tquad1+quad1(l)
       enpot(lee)=(3.*tquad+6.*tquado-tquadoo)/8.
       enkin(lee)=tquad1*dv*du
       tenergy(lee)=abs(enpot(lee)+enkin(lee))
     timengy(lee)=time(n)
     enbend(lee)=(3.*tquad2+6.*tquad2o-tquad2oo)/8.*du*dv
     encurve(lee)=abs(3.*tquad3+6.*tquad3o-tquad3oo)/8.*du*dv
     tenrel(lee)=abs(tenergy(lee))/(enkin(lee)+abs(enpot(lee)))*2
     enflr(lee)=abs(3.*tquad4+6.*tquad4o-tquad4oo)/8.*du*dv
     ltt=0
      end if
       return
       end

       function psum(f,k)

       dimension f(1)

       nq=k-4
       psum=ssum(nq,f(3),1)+.5*(f(2)+f(k-1))
       return
       end

       subroutine ddz(f1,f2)

     use const
       dimension f1(1),f2(1)

       do 10 i=4,ix-1
10     f2(i-1)=((f1(i)-f1(i-2))/(2.*du)*uuz(i-1))**2
       f2(2)=(((1.-fi1)*f1(2)*uuzh(1)+(f1(3)-f1(2))*uuzh(2))/(du*2))**2
       f2(ix-1)=((-(1.-fizx)*f1(ix-1)*uuzh(ix-1)+(f1(ix-1)-f1(ix-2))
     c *uuzh(ix-2))/(du*2))**2
       return
       end
```

```
          subroutine ddpsi(f1,f2,i)

        use const
          dimension f1(1),f2(izx,1)

          do 10 j=4,jx-1
 10       f2(i,j-1)=(f1(j)-f1(j-2))/(2.*dv)*vpsi(j-1)
          f2(i,2)=(2.*f1(2)*vpsih(1)+(f1(3)-f1(2))*vpsih(2))/(dv*2)
          f2(i,jx-1)=(-(1.-fjrx)*f1(jx-1)*vpsih(jx-1)+(f1(jx-1)-
     c f1(jx-2))*vpsih(jx-2))/(dv*2)
          return
          end
        subroutine equil

c......special case equilibrium, 0 beta, 0 pressure, rho=const.
c..... test case 1
c.....set up 1/4/82 by r. freis

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

        data rho0/1.e12/,b0/1.e4/,azm/1./,apsim/1./

        do 10  j=1,jx
        do 10 i=1,ix
        uz=uuz(i)
        rho(i,j)=rho0
        b(i,j)=b0
        r(i,j)=sqrt(2.*abs(psi(j))/b0)
        xxx(i,j)=0.
        yyy(i,j)=0.
        qub(i,j)=b0
 10     continue
        return
        end
        subroutine equilcur

c....equilibrium for curvature driven flute mode case.

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

        do 10 i=1,ix
        do 10 j=1,jx
c..... special b(i,j) to test b.c. on flute test case
        b(i,j)=b0
        r(i,j)=sqrt(2*psi(j)/b(i,j))
        rho(i,j)=(en0-en1*r(i,j)**2*.5)*amass
        xxx(i,j)=rho(i,j)*(omeg1+omeg2)*sf6
        yyy(i,j)=rho(i,j)*(-omeg1*omeg2)*sf8
        qub(i,j)=b(i,j)
        qv(i,j)=p0/psi(jx)
  10    continue
         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

      psi0=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)/psi0)/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)
       xxx(i,j)=rho(i,j)*(2.*omegexb+omeggb-omegst)
      yyy(i,j)=-rho(i,j)*(omegexb+omeggb)*(omegexb-omegst)
      xxx(i,j)=xxx(i,j)*flr
      yyy(i,j)=yyy(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
      subroutine equiltm

c.....equilibrium for tandem mirror electron ring plug

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

      data epsb/1.e-2/, p30/.01/
c..... loop 10, calculate p2 p1 and b and dp2/dpsi and dp1/dpsi
      wp2=p2wide*.5*psi0
      psibar=-be1*.5/ce1
      psistr=-ce1/(3.*de1)
      psibar=psistr+sqrt(psistr**2-be1/(3.*de1))
      p1max=ae1+be1*psibar+ce1*psibar**2+de1*psibar**3

      do 10 j=1,jx
      psir=psi(j)/psim
      psire=psi(j)/psime
      pe1=(ae1+be1*psire+ce1*psire**2+de1*psire**3)*hpme(j)
      pe2=.5*(1.-tanh((psi(j)-psi0e)/wp2e))*(1.-hpme(j))
      p1(j)=(pe1+pe2)/p1max
      dp1dpsi(j)=(de1*3.*psire**2+
     c be1+2.*ce1*psire)/(psime*p1max)*hpme(j)-.5*(1.-(tanh
     c ((psi(j)-psi0e)/wp2e))**2)/(wp2e*p1max)*(1.-hpme(j))
      p2t(j)=(1.-tanh((psi(j)-psi0)/wp2))*.5
      dp2dpst(j)=-.5*(1.-(tanh((psi(j)-psi0)/wp2))**2)/wp2
      p3(j)=(p3a+p3b*psir+p3c*psir**2+p3d*psir**3)*hp0(j)
      dp3dpsi(j)=(p3b+2.*p3c*psir+3.*p3d*psir**2)*hp0(j)/psim
      p2(j)=p2t(j)+p3(j)*dip
      dp2dpsi(j)=dp2dpst(j)+dp3dpsi(j)*dip
      hflr(j)=1.

c..... set p2 and p3 constant at large psi
      if (p2(j).le.p2flag)then
      p2(j)=p2floor
      p1(j)=p1floor
      dp2dpsi(j)=0.
      dp1dpsi(j)=0.
      hflr(j)=0.
      end if
      do 11 i=1,ix
c     if(abp(i).eq.0)then
c     b(i,j)=bvac(i)
c     go to 11
c     end if
      u1=p2(j)*abp(i)+p1(j)*abf(i)
      u2=p2(j)*bbp(i)+.5+p1(j)*bbf(i)
      u3=p2(j)*cbp(i)-bvac(i)**2*.5+p1(j)*cbf(i)
c..... at,bt,ct coefficients for low density expansion
       at=u1
       bt=u2-.5
       ct=u3+bvac(i)**2*.5
       bv=bvac(i)
       ordera2=4.*ct*bt+4.*(bt*bv)**2+8.*at*ct*bv**2+12.*at*bt*bv**4
     c +8*at**2*bv**6
       bsq3=bv**2-2.*(ct+bt*bv**2+at*bv**4)+ordera2
       if((ordera2/bsq3).le.epsp) then
       b(i,j)=sqrt(bsq3)
      else

      t1=-u2*.5/u1
      radic2=(t1**2-u3/u1)
       round=abs(radic2)/(t1**2+abs(u3/u1))
      if (round.le.1.e-6)radic2=0.
      radic=sqrt(radic2)
      bsq1=t1+radic
      bsq2=t1-radic
      bsq=bsq1
      if(t1.gt.radic)bsq=bsq2
      b(i,j)=sqrt(bsq)
      end if
   11 continue
  10  continue

c......loop 15, calculate phi, the electric potential
      do 15 i=1,ix
      arg1=((z(i)-z1)/(z1-z0))**2*(-xpot)*(1.-hzt0(i))
      arg2=wpot*((z(i)-z0)/(z0-z2))**2
```

```
 15   phi1(i)=phice*exp(arg1)+phipl/cosh(arg2)
c.....special phi1 for high mm rotation mode test
      do 17 i=1,ix
      phi1(i)=phice*(1.-dphi*(1.-hzt0(i)))
 17   continue
      do 16 j=2,jx
c     phi2(j)=(1.-(psi(j)*hp3(j)/psi3)**ypot)*hp3(j)
c..... special phi2 for constant omegexb
      phi2(j)=er*psi(j)
 16   continue
c.....loop 20,calculate pperp,  ppar and db/dpsi
c.....first calculate d coef. for ppar
      dp1=(ap1*bmax**2*4./3.+2.*bp1)*bmax
      dp2=(ap2-ap1)*bmn1**3/3.+(bp2-bp1)*bmn1+dp1+(cp1-cp2)
     c /bmn1
      dp3=(ap3-ap2)*bvx2**3/3.+(bp3-bp2)*bvx2+dp2+(cp2-cp3)
     c /bvx2
      dt0=-8.*ct0/(bmax*3.)
      dt1=-8.*ct1/(bvx2*3.)
      dt2=-8.*ct2/(3.*bvx3)
      dt3=-8.*ct3/(3.*bm1)
      ppt=1./(1.-1./rp1**2)**2
      dter1=(-8./3-4.*(bvx3/bmn2)**3/3.+4.*(bvx3/bmn2))/bmn2
      dter2=-alsi*ppt*dter1*psls1
      do 20 i=1,ix
      do 20 j=1,jx
      if(dter2.eq.0.)dter2=0.
      b2=b(i,j)**2
      b4=b2**2
      dter=dp1*hzp1(i)+dp2*hzp2(i)+dp3*hzp3(i)+dt1*hzt1(i)
     c +dt2*hzt2(i)-dt3*hzt3(i)+dt0*hzt0(i)
      ppart=(-abp(i)/3.*b4-bbp(i)*b2+cbp(i)+dter*b(i,j))
      ppert=(abp(i)*b4+bbp(i)*b2+cbp(i))
      pperte=(abq(i)*b4+bbq(i)*b2+cbq(i))
      pperp(i,j)=p2(j)*ppert
      ppar(i,j)=p2(j)*ppart
      qub(i,j)=  (b2+ppar(i,j)-pperp(i,j))/b(i,j)
c     rho(i,j)=amass*(p2(j)*(ebp(i)*b4+fbp(i)*b2+gbp(i))+
c    c ncenter*cold*(1.+(rcwall-1.)*psi(j)/psi(jx)))
c...... special rho for exb rotation case
      rho(i,j)=amass*ncenter*exp(exrho*psi(j)/psi(jx))
      fac=b(i,j)*(1.+2.*p1(j)*(2.*abf(i)*b2+bbf(i)))
      fac1=dp1dpsi(j)*(abf(i)*b4+bbf(i)*b2+cbf(i))
      fac2=dp2dpsi(j)*(abp(i)*b4+bbp(i)*b2+cbp(i))
      fac3=p2(j)*(2.*abp(i)*b2+bbp(i))+p1(j)*(2.*abf(i)*b2+bbf(i))
      dbdpsi(i,j)=-(fac1+fac2)/(b(i,j)*(1.+fac3*2.))
      dp2db=b(i,j)*(abp(i)*b2*2.+bbp(i))*2.
      dp2dbe=b(i,j)*(abq(i)*b2*2.+bbq(i))*2.
      dp3db=dp2db-4.*abp(i)*b(i,j)**3/3.-2.*bbp(i)*b(i,j)
     c +dter
      qubv(i,j)=(-dp2dpsi(j)*(ppert+ppart)-p2(j)*dp3db*dbdpsi(i,J))
      pharg=psi(j)*hp3(j)
c     dphidpsi=phi1(i)*ypot*(pharg/psi3)**(ypot-1.)/
c    c (-psi3)*hp3(j)
c..... special dphidpsi for constant omegexb
      dphidpsi=phi1(i)*er/cee
      epsi(i,j)=-dphidpsi
      dpdpsi=pperte*dp2dpsi(j)+p2(j)*dp2dbe*dbdpsi(i,J)
      omegci=echarg*b(i,j)/(amass*cee)
      unit=1./(sqrt(4.*pi))
      omegstr=-unit*b(i,j)*dpdpsi/(omegci*rho(i,j))
      omeggb=unit*pperte*p2(j)*dbdpsi(i,j)/(omegci*rho(i,j))
      omegexb=-cee*dphidpsi
      xxx(i,j)=hflr(j)*rho(i,j)*(omeggb-omegstr+2.*omegexb)*sf6
      yyy(i,j)=-hflr(j)*rho(i,j)*(omegexb+omeggb)*(omegexb-omegstr)*sf8

      ppertr=(abf(i)*b4+bbf(i)*b2+cbf(i))
      pperps(i,j)=pperp(i,j)+p1(j)*ppertr
      pperpe(i,j)=p1(j)*ppertr
 20   continue

c......check for negative pressure, and terminate prob. if necessary
      do 25 j=1,jx
      do 25 i=1,ix
      if(pperp(i,j).lt.0.or.pperpe(i,j).lt.0.)then
      write(59,101)
 101  format('problem terminated due to negative pressure' )
      call exit(1)
      end if
  25  continue

c....calculate diagnostic on perpendicular pressure balance
      do 80 i=1,ix
      do 80 j=1,jx
      errprp(i,j)=(b(i,j)**2-bvac(i)**2+2.*pperps(i,j))/bvac(i)**2
 80   continue
c....calculate diagnostic on parallel pressure balance
      do 81 j=2,jx
      do 81 i=2,ix-1
      delb=b(i+1,j)-b(i-1,j)
      if(abs(delb).lt.epsb)then
      abar=-abp(i)*5./3.
      bbar=-3.*bbp(i)
      dter4=dp1*hzp1(i)+dp2*hzp2(i)+dp3*hzp3(i)+dt1*hzt1(i)
     c +dt2*hzt2(i)-dt3*hzt3(i)
      dbar=dter4*2.
      factor=-(abar*b(i,j)**4+bbar*b(i,J)**2+cbp(i)+dbar*b(i,j))
      else
      factor=(ppar(i+1,j)*b(i+1,j)-ppar(i-1,j)*b(i-1,j))
     c /delb
       end if
  81  errprl(i-1,j-1)=(pperp(i,j)-2.*ppar(i,j)+factor)/bvac(i)**2*2

c.....cal. r and rzz*r
      do 40 i=1,ix
      sum1=0.
      sum2=0
      sum3=0.
      sum4=0.
      sum5=0.
       do 40 j=2,jx
       dps=dpsi(j)
      psib=psi(j-1)
       if(j.eq.2)then
      dps=psi(2)
      psib=0.
      end if
       if(i.gt.2)go to 31
      call bcast(hpk0(1),0.,kin)
      call bcast(hpk12(1),0.,kin)
      call bcast(hpkm(1),0. kin)
      call bcast(hpkme(1),0.,kin)
      do 29 k=1,kin
      psik=psib+dps*(k-1)/(kin-1)
      if(psik.lt.psi0)hpk0(k)=1.
      if(psik.lt.psim)hpkm(k)=1.
      if(psik.lt.psime)hpkme(k)=1.
      if(psik.lt.psi2.and.psik.gt.psi1)hpk12(k)=1.
 29   continue
c..... cal. p1 (p1k) at intermediate points in dpsi interval
        do 30 k=1,kin
        psik=psib+dps*(k-1)/(kin-1)
      psikr=psik/psim
      psikre=psik/psime
        p1ek=(ae1+be1*psikre+ce1*psikre**2+de1*psikre**3)*hpkme(k)
      p2ek=.5*(1.-tanh((psik-psi0e)/wp2e))*(1.-hpkme(k))
        p1k(k,j)=(p1ek+p2ek)/p1max
      p3k=(p3a+p3b*psikr+p3c*psikr**2+p3d*psikr**3)*hpk0(k)
      p2k(k,j)=(1.-tanh((psik-psi0)/wp2))*.5 +p3k*dip
 30    continue
 31    continue
       do 32 k=1,kin
      if(p2k(k,j).le.p2flag)then
      p2k(k,j)=p2foor
      p1k(k,j)=p1floor
      end if
      u1=p2k(k,j)*abp(i)+p1k(k,j)*abf(i)
      u2=p2k(k,j)*bbp(i)+p1k(k,j)*bbf(i)+.5
      u3=p2k(k,j)*cbp(i)+p1k(k,j)*cbf(i)-bvac(i)**2*.5
c..... at,bt,ct coefficients for low density expansion
       at=u1
       bt=u2-.5
       ct=u3+bvac(i)**2*.5
       bv=bvac(i)
       ordera2=4.*ct*bt+4.*(bt*bv)**2+8.*at*ct*bv**2+12.*at*bt*bv**4
     c +8*at**2*bv**6
       bsq3=bv**2-2.*(ct+bt*bv**2+at*bv**4)+ordera2
       if((ordera2/bsq3).le.epsp) then
       bk=sqrt(bsq3)
      else
      t1=-u2*.5/u1
      radic2=(t1**2-u3/u1)
       round=abs(radic2)/(t1**2+abs(u3/u1))
      if(round.le.1.e-6)radic2=0.
      radic=sqrt(radic2)
      bsq1=t1+radic
      bsq2=t1-radic
      bsq=bsq1
      if(t1.gt.radic)bsq=bsq2
      bk=sqrt(bsq)
      end if
       capf=1.+2.*p1k(k,j)*(2.*abf(i)*bk**2+bbf(i))+2.*p2k(k,j)*
     c (2.*abp(i)*bk**2+bbp(i))
       deli1(k)=1./bk
       deli2(k)=1./(bk**3*capf)
       deli3(k)=deli2(k)/(capf*bk**2)
       deli4(k)=p1k(k,j)/(capf*bk)**3
       deli5(k)=p2k(k,j)/(capf*bk)**3
 32    continue
      call simps(deli1,dcapi1,kin,dps)
      call simps(deli2,dcapi2,kin,dps)
      call simps(deli3,dcapi3,kin,dps)
c     call simps(deli4,dcapi4,kin,dps)
      call simps(deli5,dcapi5,kin,dps)
      capi1=dcapi1+sum1
      capi2=dcapi2+sum2
      capi3=dcapi3+sum3
      capi4=dcapi4+sum4
      capi5=dcapi5+sum5
       (i,j)=sqrt(2.*capi1)
      ,v1=(bvac(i)*dbvdz(i))**2
      vv2=bvac(i)*d2bvdz2(i)+dbvdz(i)**2
      rzz(i,j)=-vv1*capi2**2/r(i,j)**3+3.*vv1*capi3/r(i,j)
     c -vv2*capi2/r(i,j)+8.*bvac(i)**2*(abf(i)*capi4+abp(i)*
     c capi5)/r(i,j)
      sum1=+capi1
      sum2=capi2
      sum3=capi3
      sum4=capi4
      sum5=capi5
 40   continue
c.....set r(i,1)=r(i,2)
      do 41 i=1,ix
 41   r(i,1)=r(i,2)
c      calculate diagnositc quantities flute1, flute2 and flute3
      do 60 j=2,jx-1
      is=2
      if(mod(ix-2,2).eq.0)is=3
      do 61 i=is,ix-1
      ia=i-is+1
      eterm=dp2dpsi(j)*(abp(i)*b(i,j)**4+bbp(i)*b(i,j)**2+cbp(i))
     c +p2(j)*(4*abp(i)*b(i,j)**3+2*bbp(i)*b(i j))*dbdpsi(i,j)
      rterm=0.
      if(dp2dpsi(j).ne.0.)
c    c rterm=dp2dpsi(j)*(ebp(i)*b(i,j)**4+fbp(i)*b(i,j)**2+gbp(i))
c    c /(2.*p2(j)**.5)
c    c +p2(j)**.5*(4*ebp(i)*b(i,j)**3+2*fbp(i)*b(i,j))*dbdpsi(i,j)
     c rterm=exrho/psi(jx)*rho(i,j)
       droterm(ia)=rterm/b(i,j)**2/uuz(i)
      ringj=dp1dpsi(j)*(abf(i)*b(i,j)**4+bbf(i)*b(i,j)**2+cbf(i))
     c +p1(j)*(4*abf(i)*b(i,j)**3+2*bbf(i)*b(i,j))*dbdpsi(i,j)
      dflute1(ia)=+rzz(i,j)*qubv(i,j)/(r(i,j)*b(i,j)**2)/uuz(i)
     c +eterm*ringj/b(i,j)**3/uuz(i)
      dflute2(ia)=yyy(i,j)/((r(i,j)*b(i,j)
     c )**2*b(i,j))/uuz(i)
      dflute3(ia)=rho(i,j)/((r(i,j) *b(i,j))**2*b(i,j))/uuz(i)
      dflute4(ia)=xxx(i,j)/((r(i,j) *b(i,j))**2*b(i,j))/uuz(i)
 61   continue
      call simps(dflute1,ans1,ia,du)
      call simps(dflute2,ans2,ia,du)
      call simps(dflute3,ans3,ia,du)
      call simps(dflute4,ans4,ia,du)
      call simps(droterm,ans5,ia,du)
      flute1(j)=ans1*(ia-1)
      flute2(j)=(ans2*(mm**2-1)+ans1)*(ia-1)
      flute3(j)=-ans1/ans3
      rhoave(j)=ans3*(ia-1)
      xxxave(j)=ans4*(ia-1)
      yyyave(j)=ans2*(ia-1)
      droave(j)=ans5*(ia-1)
```

```
  60   continue

       grow=0.
       growmax=0.
       do 62 j=2,jx-1
       if(flute3(j).gt.0)grow=grow+flute3(j)
       growmax=amax1(growmax,flute3(j))
   62  continue
       write(59,102)grow,growmax
  102  format('grow=',e14.6,3x,'growmax=',e14.6)
c..... local growth rate for high mm, (wkb approx. )
       do 70 j=2,jx-1
       omegwkb=mm*.5*xxxave(j)/rhoave(j)
       trad=.25*mm**2*(xxxave(j)**2+4.*rhoave(j)*yyyave(j))
     c -rhoave(j)**2*flute3(j)-droave(j)*yyyave(j)
       if(trad.le.0.)then
       gamwkb(j)=sqrt(-trad)/rhoave(j)
       omeg1wkb(j)=omegwkb
       omeg2wkb(j)=0.
       else
       omeg1wkb(j)=omegwkb +sqrt(trad)/rhoave(j)
       omeg2wkb(j)=omegwkb-sqrt(trad)/rhoave(j)
       end if
  70   continue
       return
       end
       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=vpsi(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*xxx(i,j)*r4*bb
       f4(i,j)=(1.-m2)*mm*xxx(i,j)/bb
       f5(i,j)=-m2*yyy(i,j)*r4*bb
       f7(i,j)=(1.-m2)*(-m2)*yyy(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
       dppdpsi=dp2dpsi(j)*(abp(i)*b(i,j)**4+bbp(i)*b(i,j)**2+cbp(i))
     c +p2(j)*(4*abp(i)*b(i,j)**3+2*bbp(i)*b(i,j))*dbdpsi(i,j)
       dpedpsi=dp1dpsi(j)*(abf(i)*b(i,j)**4+bbf(i)*b(i,j)**2+cbf(i))
     c +p1(j)*(4*abf(i)*b(i,j)**3+2*bbf(i)*b(i,j))*dbdpsi(i,j)
       ering(i,j)=dppdpsi*dpedpsi
```

```
      g1(i,j)=+(mm*uuz(i))**2*r(i,j)*(rzz(i,j)
     c*qubv(i,j)+dppdpsi*dpedpsi*r(i,j)/b(i,j))
      g1(i,j)=g1(i,j)*swg1
      g2(i,j)=g2(i,j)*swg2
      g3(i,j)=g3(i,j)*swg3
      g4(i,j)=g4(i,j)*swg4
c..... special g1 to test b.c. on flute test case
c     g1(i,j)=-(mm*uuz(i))**2*r(i,j)*r(i,j)/lb**2
c    c*qv(i,j)


 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


      subroutine grid

c..... relates physical grid z,psi to computational grid u,v (equally
c..... spaced ). uses input fpsi, fv, fz, fu and azm, apsim .

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

      xv=alog(fpsi)/alog(fv)
      xu=alog(fz)/alog(fu)
      zzp=0.
      psip=0.
      do 5 i=1,ix
  5   u(i)=u0+du*(i-1.5)
      u(1)=-u(1)
      azm=u(ix)**(1.-xu)
      do 10 i=1,ix
      uuz(i)=u(i)**(1.-xu)/(xu*azm)
      z(i)=azm*u(i)**xu

      uuzh(i)=(u(i)+.5*du)**(1.-xu)/(xu*azm)
      dz(i)=z(i)-zzp
      zzp=z(i)
   10 continue
      zedge=azm*.5*(u(ix)**xu+u(ix-1)**xu)
      do 15 j=1,jx
      v(j)=v0+(j-1.5)*dv
  15  continue
      v(1)=-v(1)
      apsim=v(jx)**(1.-xv)
      do 20 j=1,jx
      vpsi(j)=v(j)**(1.-xv)/(apsim*xv)
      vpsih(j)=(v(j)+.5*dv)**(1.-xv)/(apsim*xv)
      psi(j)=apsim*v(j)**xv
      dpsi(j)=psi(j)-psip
      psip=psi(j)
   20 continue
c..... calculates v at plasma edge
       vw=(psiw/apsim)**(1./xv)
      dvin=(vw-v(jx-1))/dv
      dvout=(v(jx)-vw)/dv
      return
       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/

      rbf=1.
      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(psih.gt.psi(j))then
      rbf=1./(r(i,j)*b(i,j))
      kzsp=0
      else
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      kzsp=1
      endif
    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
      cosx=cos(.5*pi*(z(i)*kzsp)/zedge)
      xro(k)=ex0*cosx*rbf*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xio(k)=ex0*cosx*rbf*(r3+r4-1.)+ex1*cos(.5*pi*z(i)/zedge)
c     xio(k)=cos(theta0)*xro(k)
   10 continue
      do 20 j=2,jx-1
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      do 20 i=2,ix-1
      if(psih.gt.psi(j))then
      rbf=1./(r(i,j)*b(i,j))
      kzsp=0
      else
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      kzsp=1
      endif
   15  continue
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      cosx=cos(.5*pi*(z(i)*kzsp)/zedge)
      xroo(k)=ex0*cosx*rbf*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xioo(k)=ex0*cosx*rbf*(r3+r4-1.)+ex1*cos(.5*pi*z(i)/zedge)
c     xioo(k)=cos(theta0)*xroo(k)
  20  continue
      return
      end

      subroutine input
c.....insert storage cliches here
      use param
      use const
      use fstor

c..... boundary conditions are set as follows:
c.....         at z=z0 (i=1), fi1=-1. implies x=0.
c.....                        fi1=1. implies slope=0.
c.....       at z=zmax (i=ix), fizx=-1. implies x=0
c.....                          fizx=1. implies slope=0.
c.....      at psi=psi0 (j=1), fj1=-1. implies x=0.
c.....                         fj1=1. implies slope=0.
c.....      at psi=max (j=jx), fjrx=-1. implies x=0.
c.....                         fjrx=1. implies slope=0.
      data mm/4/, bias/.5/ lmax/2/, nmax/5/,dv/1./, du/1./
     c ,ndiag/100/,fi1/1./,fizx/1./,fj1/1./,fjrx/1./,flr/1./
     c ,sf6/1./, sf8/1./, kplotm/0/, kzs/1/, swg1/1./, swg2/1./
     c , swg3/1./, swg4/1./, nen/1/

c.....forced data loaded for testing fourier analyses and zed file
c.....maker
      data jfour/1/,nfourp/1/,nfourmax/5/
      namelist/now1/aname,mm, bias, lmax, nmax,ndiag,nen
     c ,fi1,fizx,fj1,fjrx,ex0,ex1,fpsi,fu,fv,fz
     c ,kplotm,kzs,jfour,nfourp,nfourmax,sf6,sf8,swg1,swg2,swg3,swg4

      call ddi(now1,2,3,1)
      if(nold.ne.1)call ddo(now1,100,0,1)
      jx=jrx
      kxx=kxp
      ix=izx
      return
      end
      subroutine inptemp

c.....temporary input to test three region equilib.
      use param
      use const
      real klbsq
      namelist/btemp/bmx1,bmx2,bmx3,bmn1,bmn2,betpas1,ppas2
     c ,ppas3,betrap,z1c,z2c,z3c,z1min,z2min,as,als
```