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

      cliche param
      parameter (izx=66, jrx=30, 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 (kslm=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 ,ph1(izx,jrx),yyy(izx,jrx),xxx(izx,jrx),qv(izx,jrx)
      common/pertur/x1oo(kxp),x1o(kxp),x1ol(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,fl1,flzx,fj1
     c ,fjrx,kplot,npm,fpsi,fz,fu,fv,azm,apsim,u0,v0,amass
     c ,fourpi,omegst,omegr,omegexb,flr,sf6,sf8,kplotm,kzs,zedge
     c ,cpuo,clo,syso,valfk,xu,xv,n,pi,vw,psiw,dvin,dvout
      common/contm/
     c psi0rel,psi1rel,psi2rel,z1rel,z2rel,z3rel,z0rel,nslosh,bmg
     c ,ncenter,pslosh,pcenter,rp,ztrans,ltrans,bm,ltran,ztran,rp1
     c ,bcen,pring,epsp,phicen,phipig,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,psls1,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
     c ,psi0erel,psi0e,psime,p2ewide,wp2e
      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)
      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),h1(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),h56(izx),abr(izx)
```

```
      c  ,bbr(lzx),cbr(lzx),betring,hp0(jrx),hpm(jrx),hpme(jrx)
         common/tmfield/bvac(lzx),dbvdz(lzx),d2bvdz2(lzx),dp1dps1(jrx)
      c  ,p1(jrx),p1k(ksim,jrx),hpk12(ksim),hpk0(ksim),del11(ksim)
      c  ,del12(ksim),del13(ksim),del14(ksim),rzz(lzx,jrx),dbdps1(lzx,jrx)
      c   ,phi1(lzx),phi2(lzx),pperp(lzx,jrx),ppar(lzx,jrx),dflute3(lzx)
      c  ,qubv(lzx,jrx),p2(jrx),dp2dps1(jrx),dflute1(lzx),dflute2(lzx)
      c  ,flute1(jrx),flute2(jrx),flute3(jrx),p2k(ksim,jrx),del15(ksim)
      c  ,pperps(lzx,jrx),errprp(lzx,jrx),errpr1(lzx2,jrx-1)
      c  ,pperpe(lzx,jrx),eps1(lzx,jrx),omeg1wkb(jrx),omeg2wkb(jrx)
      c  ,gamwkb(jrx),dflute4(lzx),rhoave(jrx),xxxave(jrx),yyyave(jrx)
      c  ,p2t(jrx),dp2dps1(jrx),p3(jrx),dp3dps1(jrx),hpkm(ksim)
      c  ,hpkme(ksim)
       common/forzed/nfour,nfourx,nfourmax,nfourp,jfour,lxp,locv

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

       return
       end

c..... the main routine

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
r.....z ), nfourp ( analyze 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 = 1.-(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))
      use param
      use fstor
      use matrix
      use const

      data t'  ' e6/
      integer tallyb(2000b)
      common / q8iocs/iocf(0:15)
      common/pic100/npete
      data itally/1/

c......call link call here
      call link('unit59=terminal,unit2=(infltm4,open),unit3=(output,
     c create) //')

      if(itally.gt.0) then
      do 200 ii=1,15
 200  if(iocf(ii).eq.0)go to 210
      ii=0
 210  ioctally=ii
      ioctally=14
      if(ioctally.eq.0)go to 299
      call timer(ioctally,'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)
      call pstart(dev,4rplot,1,'box u21$',1)
c     npete=1
      call p100
      call input
      call inputtm
c     call rigidcon
c     call curvecon
      call grid
      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(xrol(1),xro(1),kxx)
      call mymove(xiol(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))
      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
       if(mod(n,nfourp).eq.0)call fourier
 100  continue
```

```
      call clsdsk(iocv,0)
      call timeused(icp,io,isy)
      cpuo=icp*tim
      cio=io*tim
      syso=isy*tim
      call picsher
      call close(100)
      call keep80(1,3)
      call fr80id
      call threed
      call plote
      call timend
      call exit(1)
      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,lb,rw,xm
     c  ,zol,dtrel,theta0
      call ddi(curve,2,3,1)
      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 inputtm

c..... calculates grid quantities needed by sub. grid
c..... in floratm version

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

c.....glossary of input parameters
c       psi0rel..value of psi (in relative units) where plasma pperp
c      profile is half the maximum.
c      p2wide..parameter inversely proportional to "ramp width" of p2
c       psi0erel..realitive value of psi where ering pperp (pe2) is half
c       the maximum
c       p2ewide..parameter inversely proportional to "ramp width" of pe2
c       psi1rel..lower limit of psi where pring =0.
c       psi2..upper limit of psi where pring=0.
c       z0rel..axial center of plug
c       z1rel..axial position of plug inward mirror
c       z2rel..axial position of plug outboard mirror
c       z3rel..inner axial position where pring=0.
c       z4rel..outter axial position where pring=0.
c       nslosh..peak sloshing ion number density
c       ncenter..peak center cell ion density
c       betslsh...peak sloshing ion beta-perp w.r.t. bvac(z0)
c       betcent...peak center cell ion beta-perp w.r.t. bvac(z=0)
c       betslse...peak sloshing electron beta-perp w.r.t. bvac(z0)
c       betcene...peak center cell electron beta-perp w.r.t. bvac(z=0)
c       betring...peak electron ring beta perp w.r.t. bvac(z0)
```

```
c       rp...plug vacuum mirror ratio
c       ztrans...begining of center cell transition region
c       ltrans...transition length
c       bmg...vacuum mirror peak (approx.)  in gauss
c       bceng...center cell bmax in gauss
c       epsp...minimum pressure, below which b=bvac
c       phicen...center cell potential relative to plug ion t perp
c       phiplg...plug potential relative to plug ion t perp
c       kin...number of points i n sismsons rule quadratures
c       xpot...exponent coefficient in center cell potential
c       wpot...exponent coefficient in plug potential
c       ypot...power of polynomial in potential psi dependence
c       pxp1...power of polynomial in e ring pperp(psi) (p1(psi))
c       pxp2...power of polynomial in e ring pperp(psi) (p1(psi))
c       zmax...axial end of system in cm.
c...e ring pperp at psi=0.

c..... input for tandem mirror equlib.
       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 ,xpot/1./,ypot/2./,wpot/2./,kin/5/psi1rel/2./,psi2rel/1.5/
      c ,psi0rel/.50/,zmax/100./,z1rel/.5/,z2rel/1./,z0rel/.75/
      c ,z3rel/.625/,z4rel/.875/,ztrans/.4/,ltrans/.05/, p2wide/.1/
      c ,bmg/1.e4/, bceng/1.e3/, nsloshin/2.e13/, ncenter/1.e13/,
      c betslsh/.25/, betcent/.10/, rp/4./, epsp/1.e-6/,rp1/2./
      c ,phicen/.1/, phiplg/.1/, betcene/.1/, betslse/.1/
      c ,dt/1.e-5/, pfudge/0./, cold/1.e-3/, betring/0./,psi3rel/1.05/
      c ,pxp1/2./, pxp2/2./, safe/.9/, pe10/0./,psi0erel/.5/,p2ewide/.2/
       namelist/curve/echarg,lb,rw,xm,rw1
      c ,zmax,dt,theta0,pfudge
      c ,psi0rel,z1rel,z2rel,z0rel,nsloshin,cold,z3rel,z4rel
      c ,ncenter,betslsh,betcent,rp,ztrans,ltrans,bmg,rp1
      c ,bceng,epsp,phicen,phiplg,kin,xpot,ypot,wpot,betcene,betslse
      c ,betring,psi1rel,psi2rel,p2wide,psi3rel,pxp1,pxp2,safe,pe10
      c ,p2ewide,psi0erel
       call ddi(curve,2,3,1)
       call pframe
       call p100
       call ddo(curve,100,0,1)
c.....transform input z into physical units
       z0=z0rel*zmax
       z1=z1rel*zmax
       z2=z2rel*zmax
       z3=z3rel*zmax
       z4=z4rel*zmax
       ztran=ztrans*zmax
       ltran=ltrans*zmax
       bm=bmg/sqrt(4.*pi)
       bcen=bceng/sqrt(4.*pi)
c..... generate zmax, psimax, u(ix),v(jx), du, dv
       asq=.25/((2.*rp)**(2./3.)-1.)*(z2-z1)**2
        argt1=+(z1-ztran)/ltran
       bv0=bm*(1./(1.+(z0-z1)**2/asq)**1.5+1./(1.+(z0-z2)**2/asq)
      c **1.5)
       bmax=bm*(1.+1./(1.+(z1-z2)**2/asq)**1.5) +bcen*.5*
      c (1.-tanh(argt1))
       bv4=bm*(1./(1.+(z1)**2/asq)**1.5+1./(1.+(z2)**2/asq)**1.5)
       psimax=rw**2*bmax*.5
       psiw=rw1**2*bmax*.5
       psi0=psimax*psi0rel
```

```
      psi0e=psimax*psi0erel
      psi3=psimax*psi3rel
      psi1=psimax*psi1rel
      psi2=psimax*psi2rel
      v(jx)=psimax
      u(ix)=zmax
      du=u(ix)/(ix-1.5)
      dv=v(jx)/(jx-1.5)
c.....transform input constants from relative to physical units
      pmag0=(bv0)**2*.5
      pmag1=(bv4+bcen)**2*.5
      psloshin=(betslsh)*pmag0
      pcenter=(betcent)*pmag1
      psloshen=betslse*pmag0
      pcentee=betcene*pmag1
      pring=betring*pmag0
      phice=phicen*psloshin/(nsloshin*echarg)
      phipl=phiplg*psloshin/(nsloshin*echarg)
c....calculate maximum p1(psi)
      psidif=psi2-psi1
      p1max=(pxp1*psidif/((pxp1+pxp2)*psi1))**pxp1*(pxp2*psidif/
     c ((pxp1+pxp2)*psi2))**pxp2
c.....check that kin is not too big
      ksimp=ksim
      if(kin.gt.ksimp)then
      kin=ksimp
      write(59,200)kin
 200  format('kin initially too large, reduced to',i4)
      end if
c..... check that kin is odd for simpsons quadratures
      kch=mod(kin,2)
      if (kch.eq.0)then
      kin=kin-1
      write(59,201)kin
 201  format('kin initially even, changed to',i4)
      end if
c..... adjust epsp to accomodate zero pressure case
      epsp=epsp*(1.e-4+psloshin+pcenter)/(1.e-8*epsp+psloshin+pcenter)
c.....check that bcen and betcen are physically consistent
      asq=.25/((2.*rp)**(2./3.)-1.)*(z2-z1)**2
      bv0=bm*(1./(1.+(z0-z1)**2/asq)**1.5+1./(1.+(z0-z2)**2/asq)
     c**1.5)
      argt3=(z3-ztran)/ltran
      bv3=bm*(1./(1.+(z3-z1)**2/asq)**1.5+1./(1.+(z3-z2)**2/asq)
     c**1.5)+bcen*.5*(1.-tanh(argt3))
      rpx=bmax/(bv0)
      rp3=bv3/bv0
      afac=(rp*(1.-1./rp**2))**2/bm**2
      afac1=(rp*(1.-1./rp3**2))**2/bm**2
      tcon6=pring/((1.-1./rp3**2)**2)
      tmin=afac*(bcen+bv4)**2
      if(tmin.le.betcent+betcene)then
      bcen1=sqrt((betcent+betcene)/afac)-bv4
      tmine=tmin-betcene
      write(59,210)bcen1,tmine
 210  format('initial central cell beta and b are inconsistent for',
     c ' equlibrium',/'either increase bcen to more than',e14.6/
     c 'or decrease betcent to less than',e14.6/'use namelist',
     c ' data format')
       namelist/better/betcent,bcen
```

```
      read(59,better)
      pcenter=betcent*pmag0
      end if
c..... check plug ion beta and e ring beta
      tmin1=afac*bv0**2
      if (tmin1.le.(betslsh+betslse))then
      tbetsl=tmin1-betslse
      tbetr=(-betslsh+tmin1)
      delbet=betslsh+betslse-tmin1
      write(59,220)delbet,tbetsl,tbetr
      namelist/fix/betslsh,betslse
 220  format('the sum of (betslsh+betslse) is too large  by',
     c e14.6/'either decrease betslsh to',e14.6/'or decrease
     c  betslse to',e14.6/'or something (use namelist data format)')
      read(59,fix)
      psloshin=betslsh*pmag0
      psloshen=betslse*pmag0
      end if

c..... check betring in plug
      if(tcon6.ge.(safe*.5*bv0**2))then
      ratio=safe*.5*bv0**2/tcon6
      pring=pring*ratio
      ratinv=1./ratio
      write(59,231)ratinv,pring
 231   format('pring was too large by a factor (',e14.6,'), and was',
     c' reduced to ',e14.6 )
      end if

c..... check that rp1 is less than rpx
      if(rp1.ge.rpx)then
      rp1t=rp1
      rp1=.5*rpx
      write(59,230)rp1t,rpx,rp1
 230  format('rp1 (',e12.6,') was larger than rpx (',e12.6,'), and was',
     c' reduced to ',e12.6 )
      end if
      end
      subroutine tmcon2

c..... calculates equilibruim quantities for tandem mirror

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

c.....calculate heaveside step functions h12, h1, h34
      call bcast(h12(1),0.,ix)
      call bcast(h1(1),0.,ix)
      call bcast(h34(1),0.,ix)
      call bcast(htrans(1),0.,ix)
       do 2 i=1,ix
       if(z(i).ge.z1.and.z(i).le.z2)h12(i)=1.
       if(z(i).le.z1)h1(i)=1.
      if(z(i).ge.ztran)htrans(i)=1.
  2    if(z(i).ge.z3.and.z(i).le.z4)h34(i)=1.

      psim=psi0*(1.-p2wide)
      psime=psi0e*(1.-p2ewide)
```

```
c.....calculate heaveside step functions hp0(j), hp3(j) and hp12(j)
      call bcast(hp12(1),0.,jx)
      call bcast(hp3(1),0.,jx)
      call bcast(hp0(1),0.,jx)
      call bcast(hpm(1),0.,jx)
      call bcast(hpme(1),0.,jx)
      do 4 j=1,jx
      if(psi(j).le.psi0)hp0(j)=1.
      if(psi(j).le.psi3)hp3(j)=1.
      if(psi(j).le.psim)hpm(j)=1.
      if(psi(j).le.psime)hpme(j)=1.
 4    if(psi(j).ge.psi1.and.psi(j).le.psi2)hp12(j)=1.

c..... calculate vacuum b field (bvac) and derivatives dbv/dz, d2bv/dz2
      asq=.25/((2.*rp)**(2./3.)-1.)*(z2-z1)**2
      bv0=bm*(1./(1.+(z0-z1)**2/asq)**1.5+1./(1.+(z0-z2)**2/asq)
     c**1.5)
      do 5 i=1,ix
      eb1=(z(i)-z1)/asq
       eb2=(z(i)-z2)/asq
       fb1=sqrt(1.+eb1*(z(i)-z1))
       fb2=sqrt(1.+eb2*(z(i)-z2))
       argt=+(z(i)-ztran)/ltran
       dbvdz(i)=-3*bm*(eb1/fb1**5+eb2/fb2**5)-bcen*.5*(1.-
     c tanh(argt)**2)/ltran
       d2bvdz2(i)=-3*bm*(1./(asq*fb1**5)+1./(asq*fb2**5)-eb1**2*5/
     cfb1**7-eb2**2*5/fb2**7)+bcen*tanh(argt)*(1.-tanh(argt)**2)/
     c ltran**2
 5     bvac(i)=bm*(1./fb1**3+1./fb2**3)+bcen*.5*(1.-tanh(argt))
       argt3=(z3-ztran)/ltran
       bv3=bm*(1./(1.+(z3-z1)**2/asq)**1.5+1./(1.+(z3-z2)**2/asq)
      c**1.5)+bcen*.5*(1.-tanh(argt3))
       rp3=bv3/(+bv0)
       bm1=rp1*( bv0)
       alsit=((1.-1./rp1**2)/(1.-1 /rpx**2)*(rp1/rpx))**2
       alsi=alsit+.5*(1.-alsit)*prudge

c.....calculate heaveside function h56(i) for second component of
c.....  sloshing pperp
      do 10 i=1,ix
       eb2=(z(i)-z2)/asq
      if(z(i).gt.z1.and.z(i).lt.z2.and.bm1.ge.bvac(i))
     c h56(i)=1.
 10    continue

c..... calculate pressure coeff, including electron ring (pr)
      con1=1./((1.-1./rpx**2)**2*bmax**4)
      con2=con1*(-2*bmax**2)
      con3=-.5*con2*bmax**2
      con4=pring/((1.-1./rp3**2)**2*bv3**4)
      con5=con4*(-2*bv3**2)
      con6=-.5*con5*bv3**2
      don1=1./((1.-1./rp1**2)**2*bm1**4)
      don2=don1*(-2*bm1**2)
      don3=-.5*don2*bm1**2
c..... calculate constants for p3 pressure componetnt
      data p30/.01/
      wp2=p2wide*.5*psi0
      wp2e=p2ewide*.5*psi0e
      wp3=wp2/psim
```

```
        psi0r=psi0/psim
        p3b=p30
        p3c=(.5*(1.-(tanh(2.))**2)/wp3-p3b*p2wide*(2.-p2wide))
     c  *.5/p2wide
        p3d=-(p3b+2.*p3c*psi0r)/(3.*psi0r**2)
        p3a=-(p3b*psi0r+p3c*psi0r**2+p3d*psi0r**3)

c.....calculate constants for pe1
        g2p=-.5*(1.-(tanh(-2.))**2)/wp2e
        g2f=.5*(1.-tanh(-2.))
        ae1=pe10
        ce1=psime*g2p-g2f+ae1
        be1=2.*(g2f-ae1)-psime*g2p
c.....readjust peak plug pressure for sloshing
        rtemp=(bm1/bmax)**2
        rpxi2=1./rpx**2
        ter1=(rtemp-rpxi2)**2
        ter2=(1.-rpxi2)**2
        bstr2=bmax**2*(-ter1+alsi*rtemp*ter2)/(-ter1+alsi*ter2)
        pfloat=((con1-don1*alsi)*bstr2**2+(con2-don2*alsi)*bstr2
     c  +con3-don3*alsi)*(1.+p3a)
        psiosh=psioshin/pfloat
        psioshe=psioshen/pfloat
        nsiosh=nsioshin/pfloat
        psis1=psiosh+psioshe
        pcen1=pcenter+pcentee
        do 6 i=1,ix
        eon1=con1-don1*alsi*h56(i)
        eon2=con2-don2*alsi*h56(i)
        eon3=con3-don3*alsi*h56(i)
        abf(i)=con4*h34(i)
        bbf(i)=con5*h34(i)
        cbf(i)=con6*h34(i)

        abp(i)=eon1*(psis1*h12(i)+pcen1*h1(i))
        bbp(i)=eon2*(psis1*h12(i)+pcen1*h1(i))
        cbp(i)=eon3*(psis1*h12(i)+pcen1*h1(i))

        abr(i)=eon1*(nsiosh*h12(i)+ncenter*h1(i))
        bbr(i)=eon2*(nsiosh*h12(i)+ncenter*h1(i))
        cbr(i)=eon3*(nsiosh*h12(i)+ncenter*h1(i))

        abq(i)=eon1*(psiosh*h12(i)+pcenter*h1(i))
        bbq(i)=eon2*(psiosh*h12(i)+pcenter*h1(i))
 6      cbq(i)=eon3*(psiosh*h12(i)+pcenter*h1(i))
        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
      p1max=ae1+be1*psibar+ce1*psibar**2

      do 10 j=1,jx
      psir=psi(j)/psim
      psire=psi(j)/psime
      pe1=(ae1+be1*psire+ce1*psire**2)*hpme(j)
      pe2=.5*(1.-tanh((psi(j)-psi0e)/wp2e))*(1.-hpme(j))
      p1(j)=(pe1+pe2)/p1max
      dp1dpsi(j)=(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)
      dp2dpsi(j)=dp2dpst(j)+dp3dpsi(j)
      if((p2(j)).lt.epsp)go to 50
      do 11 i=1,ix
      if(abp(i).eq.0)then
      b(i,j)=bvac(i)
      go to 11
      end if
      u1=p2(j)*abp(i)+p1(j)*abf(i)
      u2=p2(j)*bbp(i)+.5+p1(j)*bbf(i)
      t1=-u2*.5/u1
      u3=p2(j)*cbp(i)-bvac(i)**2*.5+p1(j)*cbf(i)
      radic=sqrt(t1**2-u3/u1)
      bsq1=t1+radic
      bsq2=t1-radic
      bsq=bsq1
      if(t1.gt.radic)bsq=bsq2
      b(i,j)=sqrt(bsq)
   11 continue
      go to 10
c.... b=bvac if p2.lt.epsp
   50 do 51 i=1,ix
   51 b(i,j)=bvac(i)
   10 continue

c.....loop 15, calculate phi, the electric potential
      do 15 i=1,ix
      arg1=((z(i)-z1)/(z1-z0))**2*(-xpot)*(1.-h1(i))
      arg2=wpot*((z(i)-z0)/(z0-z2))**2
   15 phi1(i)=phice*exp(arg1)+phipl/cosh(arg2)
      do 16 j=2,jx
      phi2(j)=(1.-(psi(j)*hp3(j)/psi3)**ypot)*hp3(j)
   16 continue
c.....loop 20,calculate pperp,  ppar and db/dpsi
      ppt=1./(1.-1./rp1**2)**2
      dter1=(-8./3-4.*(bmax/bm1)**3/3.+4.*(bmax/bm1))/bm1
      dter2=-alsi*ppt*dter1*psis1
      do 20 i=1,ix
      do 20 j=1,jx
      b2=b(i,j)**2
      b4=b2**2
      dter3=dter2*h56(i)
      dter=(4.*abp(i)*bmax**2/3.+2.*bbp(i))*bmax+dter3
      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)
      rho(i,j)=amass*(p2(j)**.5*(abr(i)*b4+bbr(i)*b2+cbr(i))+
     c p2(j)**.5*ncenter*cold)
      fac=b(i,j)*(1.+2.*p1(j)*(2.*abf(i)*b2+bbf(i)))
      fac1=dp1dps(j)*(abf(i)*b4+bbf(i)*b2+cbf(i))
      fac2=dp2dps(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))
      dbdps(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)=(-dp2dps(j)*(ppert+ppart)-p2(j)*dp3db*dbdps(i,j))
      pharg=psi(j)*hp3(j)
      dphidpsi=phi1(i)*ypot*(pharg/psi3)**(ypot-1.)/
     c (-psi3)*hp3(j)
      epsi(i,j)=-dphidpsi
      dpdpsi=pperte*dp2dps(j)+p2(j)*dp2dbe*dbdps(i,j)
      omegci=echarg*b(i,j)/(amass*cee)
      omegstr=-b(i,j)*dpdpsi/omegci
      omeggb=pperte*p2(j)*dbdps(i,j)/omegci
      omegexb=-rho(i,j)*cee*dphidpsi*.5
      xxx(i,j)=(omeggb-omegstr+2.*omegexb)*sf6
      yyy(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 co..tinue

c....calculate diagnostic on perpendicular pressure balance
      do 80 i=1,ix
      do 80 j=i,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)
      dter3=dter2*h56(i)
      dter4=(4.*abp(i)*bmax**2/3.+2.*bbp(i))*bmax+dter3
      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 /dolb
       end if
  81   errpr(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)*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
  30    continue
  31    continue
        do 32 k=1,kin
       if((p2k(k,j)).le.epsp.or.abp(i).eq.0) then
       bk=bvac(i)
       else
       u1=p2k(k,j)*abp(i)+p1k(k,j)*abf(i)
       u2=p2k(k,j)*bbp(i)+p1k(k,j)*bbf(i)+.5
       t1=-u2*.5/u1
       u3=p2k(k,j)*cbp(i)+p1k(k,j)*cbf(i)-bvac(i)**2*.5
       radic=sqrt(t1**2-u3/u1)
       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
```

```
            dell2(k)=1./(bk**3*capf)
            dell3(k)=dell2(k)/(capf*bk**2)
            dell4(k)=p1k(k,j)/(capf*bk)**3
            dell5(k)=p2k(k,j)/(capf*bk)**3
 32         continue
           call simps(dell1,dcapi1,kin,dps)
           call simps(dell2,dcapi2,kin,dps)
           call simps(dell3,dcapi3,kin,dps)
c          call simps(dell4,dcapi4,kin,dps)
           call simps(dell5,dcapi5,kin,dps)
           capi1=dcapi1+sum1
           capi2=dcapi2+sum2
           capi3=dcapi3+sum3
           capi4=dcapi4+sum4
           capi5=dcapi5+sum5
           r(i,j)=sqrt(2.*capi1)
           vv1=(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)
           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)
           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)
 60        continue

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)
      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 simps(fin,fout,knn,df)
c.....simpsons rule quadratures, knn must be odd
      dimension fin(1)
      nse=(knn-1)/2
      se=ssum(nse,fin(2),2)
      nso=nse-1
      so=ssum(nso,fin(3),2)
      fout=df/(3.*(knn-1))*(fin(1)+fin(knn)+4.*se+2.*so)
      return
      end
      subroutine equilcur

c....equilibrium for curvature drivan 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 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 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 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(kzs.eq.0)then
      rbf=1./(r(i,j)*b(i,j))
      else
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      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
      xro(k)=ex0*rbf*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xio(k)=ex0*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(kzs.eq.0)then
      rbf=1./(r(i,j)*b(i,j))
      else
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      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
      xroo(k)=ex0*rbf*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xioo(k)=ex0*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), fji=-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./

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
     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)
      call ddo(now1,100,0,1)
      jx=jrx
      kxx=kxp
      ix=izx
      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 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)
      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)/b**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 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

      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
      if(sfjrx.eq.1.and.sfizx.eq.1)fac4=r(ix-1,jx-1)*b(ix-1,jx-1)/
```

```
      c (r(ix,jx-1)*b(ix,jx-1))
       a1(    ,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
       if(sfizx.eq.1.)sfizx=r(ix-1,j)*b(ix-1,j)/(r(ix,j)*b(ix,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 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)+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.
        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

        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 right

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

      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)*xioo(kp)
      tt2=tt2+a3(k,m)*xio(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)*(xiol(kq)-xioo(kq))
      tt3=tt3+b1(k,mn)*(xrol(kq)-xroo(kq))
    6 continue
      rhs1(k)=(2.*t2-t1+fac1*t3)
      rhs2(k)=2.*tt2-tt1+fac2*tt3
```

```
 10   continue
      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
      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))
      omegc=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*va../(2*omegst*valfk))/(1-.5/(ix-1.5))
      du=(u(ix)-u0)/(jx-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 mymove(a,b,len)
      dimension a(1),b(1)
      do 10 i=1,len
      a(i)=b(i)
 10   continue
      return
      end

      subroutine picsher
c...uses grafic, graflib and grafcore to make plots of xr vs. time and
c....space.


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

      dimens on iy(2), it(5), lab(2),dum(izx2),ymin(5),ymax(5)
     c ,dum1(nplt),rplot(jrx-1)
      data epp/1.e20/
      call orgfile(nume)
      call pframe
      call p100
      write(100,102)nume
      write(100,400)aname
 400  format(/10x,'problem identification : ' ,5a8 )
 102  format(10x,'this problem run by ',a8 )
      write (100,101) dt, ix,jx,nmax,lmax,bias,
     c flr,sf6,sf8,kplot,kzs,cpuo,cio,syso
     c ,xu,xv,bv0,bm1,bv3
 101  format(////'dt=',e16.6,4x,'ix=',i8,4x,'jx=',i8/'total time steps =
     c 'i8,4x,
     c 'no. of iterations =',i8,4x,'bias=',f10.5/
     c 'flr=',e16.8,4x,'sf6=',e16.8,4x,
     c 'sf8=',e16.8/'kplot=',i8,4x,'kzs=',i8/
     c 'cpu time =',e16.8/
     c 'i-o time =',e16.8/'sys time =',e16.8/3x,'u exponent (xu) =
     c ',e16.8/3x,'v exponent (xv)=',e16.8/'bv0=',e16.8,4x,'bm1=',
     c e16.8,4x,'bv3=',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)
```