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

      cliche param
      parameter (izx=34, 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)
      endcliche
      cliche fstor
      use param
      common/fun/f1(izx,jrx),f2(izx,jrx),f3(izx,jrx),f4(izx,jrx)
     c ,f5(izx,jrx),f7(izx,jrx),
     c g1(izx,jrx),g2(izx,jrx),g3(izx,jrx),g4(izx,jrx)

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

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

      cliche const
      use param
      common/con/gam1,gam2,ix,jx,mm,izxp,kxx,nmax,lmax,isw,ihbw
     c ,fac1,fac2,bias,du,dv,dt,ndiag,ex0,b0,rho0,ex1,fi1,fizx,fj1
     c ,fjrx,psi0,z0,kplot,npm,fpsi,fz,fu,fv,azm,apsim,u0,v0,amass
     c ,fourpi,omegst,omegr,omegexb,flr,sf6,sf8,kplotm,kzs,zedge
     c ,cpuo,cio,syso,valfk,xu,xv,n,pi
      common/mesh/psi(jrx),z(izx),u(izx),v(jrx),dpsi(jrx),dz(izx)
     c ,vpsi(jrx),uuz(izx),vpsih(jrx),uuzh(izx)
      common/graf/  xrtime(nplt),xrspz(izx,nps),xrsppsi(jrx,2*nps)
     c ,time(nplt)
      common/curvco/cr,lb,rp,rw,beta0,delrho,stable,en0,cee,r0,
     c echarg,omeg1,omeg2,en1,besarg,zol,dtrel,p0,omeg0
     c ,omana1,omana2,groana,theta0
      common/forzed/nfour,nfourx,nfourmax,nfourp,jfour,ixp,iocv

      real lb
      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.
```

```
      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(l)-xi(n-1)) , and
c..... rhs2(k)=2*a3*xi(n)-a2*xi(n-1)+b1*(xr(l)-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)
```

```
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 ( 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 )
      use param
      use fstor
      use matrix
      use const

      data tim/1,e6/
      integer tallyb(2000b)
      common / q8iocs/iocf(0:15)
      data itally/1/

c,,,,,,,call link call here
      call link('unit59=terminal,unit2=(inflora,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)
```

```
   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)*(xiol(k+koff)-xioo(k+koff))
   120  rhs2(k)=rhs2(k)+fac2*b1(k,mbar)*(xrol(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))
        omegci=echarg*b0/(amass*cee)
        omegp2=fourpi*en0*echarg**2/amass
        omegst=2.*beta0*omegci*cee**2/(omegp2*r0sq)
        vomeg=echarg*sqrt(en0*fourpi/amass)*r0sq*.5/(beta0*cee)
        u(ix)=(pi*valf/(2*omegst*valfk))/(1-.5/(ix-1.5))
        du=(u(ix)-u0)/(ix-1.5)
        targ=cosh(rw**2/r0sq+cr)*sqrt(beta0)
        v(jx)=b0*r0sq*.5*alog(targ)/sqrt(fourpi)
        dv=(v(jx)-v0)/(jx-1.5)
        return
        end

        subroutine mymove(a,b,len)
        dimension a(1),b(1)
        do 10 i=1,len
        a(i)=b(i)
   10   continue
        return
        end

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


c.....insert cliche for common  here
        use param
```

```
 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)
      call p100
      call input
c     call rigidcon
      call curvecon
      call grid
      call constant
c     call equilrot
      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)
c     call mymove(xiol(1),xio(1),kxx)
c     call mymove(xroo(1),xro(1),kxx)
      call mymove(xioo(1),xio(1),kxx)
      call banfac(kxp,ihbw,ww,1,-(kxp-1))
      t=0.
      do 100 n=1,nmax
      t=t+dt
      time(n)=t
      fac1=-1./dt
      fac2=1./dt
      do 90 l=0,lmax
      call rightvec
      call zmovewrd(ww1(nn),rhs1,kxx)
      call bansol(kxp,ihbw,ww,1,-(kxp-1),ww1(nn))
      do 10 j=2,jx-1
      kp=1+izxp*(j-2)
      call zmovewrd(xrol,ww1(nn),kxx)
  10  continue
      call zmovewrd(ww1(nn),rhs2,kxx)
      call bansol(kxp,ihbw,ww,1,-(kxp-1),ww1(nn))
      do 20 j=2,jx-1
      kp=1+kxp*(j-2)
      call zmovewrd(xiol,ww1(nn),kxx)
  20  continue
      fac1=-.5/dt
 90   fac2=.5/dt
      call zmovewrd(xioo,xio,kxx)
      call zmovewrd(xio,xiol,kxx)
      call zmovewrd(xroo,xro,kxx)
      call zmovewrd(xro,xrol,kxx)
c..... time array
      xrtime(n)=xro(kplot)
      use fstor
      use matrix
      use const

      dimension 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
 102  format(10x,'this problem run by ',a8 )
      write (100,101) dt, ix,jx,nmax,lmax,bias,omeg1,omeg2,omeg0,
     c stable,flr,sf6,sf8,kplot kzs,cpuo,cio,syso
     c ,xu,xv,en0,en1,b0,omana1,omana2,groana
 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/'omega1=',e16.8,4x,
     c 'omega2 =',e16.8,4x,'omega0 =',e16.8/
     c 'stable =',e16.8,4x,'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/'en0=',e16.8,4x,'en1=',
     c e16.8,4x,'b0=',e16.8/'analytic freq (omana1)=',e16.8,3x,'analytic
     c freq (omana2)=',e16.8/'analytic growth (groana)=',e16.8)
c.....plot coordinate stretching

      kx='u$'
      iy(1)='z$'
      it(1)='z vs u, '
      it(2)='(z=const'
      it(3)='*u**xu)$'
      call pframe
      call pscale(0,u(2),u(ix),z(2),z(ix),1)
      call pcurve(0,u(2),z(2),ix-1,1,it,kx,iy,1h$,1h$)
      kx='v$'
      iy(1)='psi$'
      it(1)='psi vs v'
      it(2)=', (psi=c'
      it(3)='onst*v**'
      it(4)='xv)$'
      call pscale(0,v(2),v(jx),psi(2),psi(jx),2)
      call pcurve(0,v(2),psi(2),jx-1,2,it,kx,iy,1h$,1h$)
      iy(1)='r(   ,v)$'
      it(1)='r(   ,v) '
      it(2)='vs v$'
      ipld=ix/4
      call zcitoa(xout,0,ipld,2,0)
      call cmove(it(1),2,xout,0,2)
      call cmove(iy(1),2,xout,0,2)
      do 5 jj=1,jx-1
    5 rplot(jj)=r(ipld,jj+1)
      call pscale(0,v(2),v(jx),rplot(1),rplot(jx-1),3)
      call pcurve(0,v(2),rplot(1),jx-1,3,it,kx,iy,1h$,1h$)
      it(4)='each    '
      it(5)='th time$'
      call zcitoa(xout,0,ndiag,3,0)
      call cmove(it(4),5,xout,0,3)
      kx='z$'
      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 picture
      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)
      iy(1)='xr(z,psi'
      iy(2)='p)$'
      it(1)='xr(z,psi'
      it(2)='(    )) v'
      it(3)='s z      '
      jpl=jx/2
      call zcitoa(xout,0,jpl,3,0)
      call cmove(it(2),1,xout,0,3)
      do 20 np=1,npm
      lab(1)=5hn=  $
      call zcitoa(xout,2,np,2,0)
      call cmove(lab(1),2,xout,2,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=izx-2
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 40
      do 50 nnp=np,np+4
      nnpp=nnp-np+1
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 55 ip=2,ix-1
      ymax(nnpp)=amax1(ymax(nnpp),xrspz(ip,nnp))
55    ymin(nnpp)=amin1(ymin(nnpp),xrspz(ip,nnp))
50    continue
      ymin1=epp
      ymax1=-epp
      do 57 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
57    ymin1=amin1(ymin1,ymin(nn))
      call pscale(0,z(2),z(ix-1),ymin1,ymax1,n4)
40    continue
      call pcurve(0,z(2),xrspz(2,np),n3,n4,it,kx,iy,1h$,lab)
   20 continue
      kx='psi$'
      do 122 ll=1,2
      ipl=ix/2**ll
      iy(1)='xr(zp,ps'
      iy(2)='i)$'
      it(1)='xr(z(    '
      it(2)='),psi) v'
      it(3)='s psi    '
      call zcitoa(xout,0,ipl,3,0)
      call cmove(it(1),5,xout,0,3)
      do 120 np=1+(ll-1)*nps,npm+(ll-1)*nps
       npld=np-(ll-1)*nps
      lab(1)=5hn=  $
      call zcitoa(xout,0,npld,2,0)
      call cmove(lab(1),2,xout,0,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=jx-2
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 140
      do 150 nnp=np,np+4
      nnpp=nnp-np+1
```

```
     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 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
      chi(i,j)=rho(i,j)*(omeg1+omeg2)*sf6
      yep(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
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 155 ip=2,jx-1
      ymax(nnpp)=amax1(ymax(nnpp),xrsppsi(ip,nnp))
  155  ymin(nnpp)=amin1(ymin(nnpp),xrsppsi(ip,nnp))
  150  continue
      ymin1=epp
      ymax1=-epp
      do 157 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
  157  ymin1=amin1(ymin1,ymin(nn))
      call pscale(0,psi(2),psi(jx-1),ymin1,ymax1,n4)
  140  continue
      call pcurve(0,psi(2),xrsppsi(2,np),n3,n4,it,kx,iy,1h$,lab)
  120 continue
 122  continue
      kx='r$'
      iy(1)='xr(zp,r)'
      iy(2)='$'
      it(1)='xr(z(    '
      it(2)='),r) vs  '
      it(3)='r,       '
      do 220 lp=1,2
      ipl=ix/2**lp
      call zcitoa(xout,0,ipl,3,0)
      call cmove(it(1),5,xout,0,3)
      do 300 jj=1,jx-2
      rplot(jj)=r(ipl,jj+1)
 300  continue
      do 220 np=1+(lp-1)*nps,npm+(lp-1)*nps
      lab(1)=5hn=  $
      npld=np-(lp-1)*nps
      call zcitoa(xout,0,npld,2,0)
      call cmove(lab(1),2,xout,0,2)
      n1=(np-1)/5
      n2=mod(n1,4)
      n3=jx-2
      n4=n2+1
      n5=mod(np,5)
      if(n2.eq.0.and.n5.eq.1)call pframe
      if(n5.ne.1)go to 240
      do 250 nnp=np,np+4
      nnpp=nnp-np+1
      ymax(nnpp)=-epp
      ymin(nnpp)=epp
      do 255 ip=2,jx-1
      ymax(nnpp)=amax1(ymax(nnpp),xrsppsi(ip,nnp))
  255  ymin(nnpp)=amin1(ymin(nnpp),xrsppsi(ip,nnp))
  250  continue
      ymin1=epp
      ymax1=-epp
      do 257 nn=1,5
      ymax1=amax1(ymax1,ymax(nn))
  257  ymin1=amin1(ymin1,ymin(nn))
      call pscale(0,rplot(1),rplot(jx-2),ymin1,ymax1,n4)
  240  continue
      call pcurve(0,rplot(1),xrsppsi(2,np),n3,n4,it,kx,iy,1h$,lab)
  220 continue
      call pframe
c..... time plots of xrtime
```

```
      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)
      chi(i,j)=0.
      yep(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)
       chi(i,j)=rho(i,j)*(2.*omegexb+omeggb-omegst)
      yep(i,j)=-rho(i,j)*(omegexb+omeggb)*(omegexb-omegst)
      chi(i,j)=chi(i,j)*flr
      yep(i,j)=yep(i,j)*flr
  5   continue
      do 30 i=1,ix
      do 30 j=2,jx-1
      qv(i,j)=.5*(qub(i,j+1)*b(i,j+1)-qub(i,j-1)*b(i,j-1))
 30   continue
      return
      end
      subroutine initial

c.....set up initial displacement vectors, xro and xio
c..... test case 1, cos(kz) in z, flat in psi
```

```
      kx='t$'
      iy(1)='xr(kplot'
      iy(2)=')$'
      it(1)='xr(kplot'
      it(2)=') vs t$'
      call pcurve(Ü,time,xrtime,nmax,12,it,kx,iy,1h$,1h$ )
      do 70 ll=1,nmax
  70  dum1(lll)=abs(xrtime(lll))
      call pcurve(2,time,dum1,nmax,34,1h$,1h$,1h$,1h$,1h$)
      call pclose
      return
      end

      subroutine diagno
c.....sets up arrays for spatial plots

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

      np=np+1
      do 10 i=2,ix-1
      kp1=i-1+(jx/2-1)*(ix-2)
      kp2=jx/2-1+(jx-2)*(i-2)
      kpp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
      xrspz(i,np)=xro(kpp)
  10  continue
      do 20 ll=1,2
      ixpl=ix/2**ll
      do 20 j=2,jx-1
      kp1=ixpl-1+(j-2)*(ix-2)
      kp2=j-1+(jx-2)*(ixpl-2)
      kpp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
      xrsppsi(j,np+(ll-1)*nps)=xro(kpp)
      xrsppsi(j,np+(ll-1)*nps)=xro(kpp)
  20  continue
      npm=np
      return
      end
      subroutine fourier

c.....controls the fourier mode analyses of xr(i,j), which
c.....uses rcpft and cpft from B. Langdon a la
c..... N Maron

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

      dimension ahist(ixpf+1,nfourxf),xrfour(ixpf,2)
     c ,cs(ixpf),si(ixpf),iahist(ixpf+1,nfourxf)
     c ,aa(ixpf+2,2)
      equivalence (ahist,iahist)
      nfour=nfour+1
      nf1=mod(nfour,2)+1
      timf0=timf
```

```
c..... set up 1/4/82 by r. freis

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

      data pi/3.1415926/

      do 10 j=2,jx-1
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
      do 10 i=2,ix-1
      if(kzs.eq.0)go to 5
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
    5 continue
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      xro(k)=ex0*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xio(k)=ex0*(r3+r4-1.)+ex1*cos(.5*pi*z(i)/zedge)
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)go to 15
      r1=ranf(b1)
      r2=ranf(b1)
      r3=ranf(b1)
      r4=ranf(b1)
   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*(r1+r2-1.)+ex1*cos(.5*pi*(z(i))/zedge)
      xioo(k)=ex0*(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

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
```

```
      timf=time(n)
      do 5 i=2,ix-1
      kp1=i-1+(jfour-1)*(ix-2)
      kp2=jfour-1+(jx-2)*(i-2)
      kpp=.5*(1+isw)*kp1+.5*(1-isw)*kp2
      kpp1=i+1-1+(jfour-1)*(ix-2)
      kpp2=jfour-1+(jx-2)*(i+1-2)
      kppp=.5*(1+isw)*kpp1+.5*(1-isw)*kpp2
 5    xrfour(i,nf1)=xro(kpp)
      xrfour(1,nf1)=fi1*xrfour(2,nf1)
 100  continue
c.....if z symetry imposed by quarter wave boundary conditions,
c.....the xr array must be expanded to half wave
      do 7 i=1,ix-1
 7    xrfour(ix-1+i,nf1)=+xrfour(ix-i,nf1)*sign(1.,fizx)
      if(n.gt.0)go to 101
      ixp1=2*(ix-1)
      ixp2=4*(ix-2)
      ixp3=2*(ix-2)
      do 200 l=1,ixp2
      cs(l)=cos(pi*(l-1)/ixp2)
      si(l)=sin(pi*(l-1)/ixp2)
 200  continue
 101  continue
      do 9 i=1,ixp1-3
  9   xrfour(ixp1-1+i,nf1)=xrfour(ixp1-i,nf1)
 8    if(nf1.eq.2)return
      if(ixp1.gt.ixp2)go to  1000
      if(ixp1.gt.ixp2+1)go to 1010
      call cpft(xrfour(1,1),xrfour(1,2),ixp2,1,+1)
 1000  continue
      do 300 l=1,ixp2
      aa(l,1)=xrfour(l,1)*cs(l)+xrfour(l,2)*si(l)
      aa(l,2)=-xrfour(l,1)*si(l)+xrfour(l,2)*cs(l)
 300  continue
      call rpft2(aa(1,1),aa(1,2),ixp2,1)
 1010  continue
      call zmovewrd(ahist(2,nfour-1),aa(1,2),ixp2)
      call zmovewrd(ahist(2,nfour),aa(1,1),ixp2)
      iahist(1,nfour-1)=nfourx*nfourinc+nfour-1
      iahist(1,nfour)=nfourx*nfourinc+nfour
      if(mod(nfour,nfourx).ne.0)return
      nfourinc=nfourinc+1
      nfour=0
      if(nfourinc.eq.1)call history(ahist)
      if(nfourinc.gt.1)call history1(ahist)
      return
      end
      subroutine fourplay

c..... initialize quantities for fourier analyses and zed
c..... history file

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

      nfour=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./,dt/1./
     c ,ndiag/100/,fi1/1./,fizx/1./,fj1/1./,fjrx/1./,flr/1./
     c ,sf6/1./, sf8/1./, kplotm/0/, kzs/1/

c.....forced data loaded for testing fourier analyses and zed file
c.....maker
      data jfour/1/,nfourp/1/,ntourmax/5/
      namelist/now1/mm, bias, lmax, nmax,dv, dt, du,ndiag
     c ,fi1,fizx,fj1,fjrx,rho0,b0,ex0,ex1,u0,v0,fpsi,fu,fv,fz
     c ,azm,apsim,flr,sf6,sf8,kplotm,kzs,jfour,nfourp,nfourmax

      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
      nfourinc=0
      if(jfour.eq.0)jfour=jx/4
      ixp=ix-1
      if(fi1*fizx.lt.0)ixp=2*(ix-2)+1
      nfourx=nfourxf
      return
      end
      subroutine history(ahisto)

c.....sets up and creates the file for zed to process
c.....assumes ixpf-1 =2**n

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

      dimension ikx(ixpf), iky(ixpf),ahisto(1),modewrd(ixpf+2)
      integer dskadd
      data itv/31b/

c.....set up the history file using iopack from J. Stewart
      iowrt=59
      ntry=1
      nsus=0
      name='florfour'
      idcu=0
      nchs=4
      nsiz=(ixpf+1)*(nfourmax*nfourx+1)+2
      call iopack(iowrt,ntry,nsus)
      call crtdsk(name,nsiz,iocu,idcu,nchs)
      iocv=iocu
      do 1 i=1,ixp-1
      iky(2*(ixp-1)+1-i)=-i
      ikx(2*(ixp-1)+1-i)=0
      iky(i)=i-1
      ikx(i)=0
 1    continue
      iky(ixp)=ixp-1

c.....pack modewords for history file
      data mask/0000000000000000777777b/, mask2/00000000000000077777777b/
      do 2 i=1,2*(ixp-1)
      itemp1=(iky(i).and.mask)
      itemp2=(ikx(i).and.mask)
      itemp3=shift(itemp2,18)
      itemp4=(itv.and.mask2)
      itemp5=shift(itemp4,36)
      modewrd(i+2)=((itemp1.or.itemp3).or.itemp5)
  2   continue
      modewrd(1)=1
      modewrd(2)=2*(ixp-1)
      dskadd=0
c..... write modewords to history file
      nwords=2*(ixp-1)+2
      call wrsdsk(iocu,modewrd,nwords,0)
      entry history1(ahisto)
      dskadd=dskadd+nwords
      nword1=(2*(ixp-1)+1)*nfourx
      psip=psi(j)
  20  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*chi(i,j)*r4*bb
      f4(i,j)=(1.-m2)*mm*chi(i,j)/bb
      f5(i,j)=-m2*yep(i,j)*r4*bb
      f7(i,j)=(1.-m2)*(-m2)*yep(i,j)/(bb*vp)
      g4(i,j)=qub(i,j)*r(i,j)**2
      g3(i,j)=r(i,j)*b(i,j)
      g2(i,j)=qub(i,j)/(r(i,j)*b(i,j))**2
      g1(i,j)=+(mm*uuz(i))**2*r(i,j)*(r(i+1,j)+r(i-1,j)-2*r(i,j))
     c*qv(i,j) /du**2
c..... special g1 to test b.c. on flute test case
      g1(i,j)=-(mm*uuz(i))**2*r(i,j)*r(i,j)/lb**2
     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.
```

```
      call wrsdsk(iocu,ahisto,nword1,dskadd)
      call chkdsk(iocu)
      nwords=nword1
      return
      end
      subroutine cpft(r,i,n,incp,signp)
      parameter (log2nx=15)
      real r(1),i(1)
      integer signp,span,rc
      double precision qt,qq
      real sines(log2nx),i0,i1
      if(sines(1).eq.1.) go to 1
      sines(1)=1.
      qt=1.
      qt=datan(qt)
      do 2 is=2,log2nx
      qq=dsin(qt)
      sines(is)=qq
    2 qt=qt*.5
    1 continue
      if(n.eq.1) return
      inc=incp
      sgn=signp
      ninc=n*inc
      span=ninc
      it=n/2
      do 1000 is=1,log2nx
c...  (2000=recur)
      if(it.eq.1) go to 2000
 1000 it=it/2
c
c  if truncated rather than rounded arithmetic is used,
c  singleton's magnitude correcton should be applied to cos and sin.
 1500 t=sin+(s*cos-c*sin)
      cos=cos-(c*cos+s*sin)
      sin=t
c...  (3000=repl)
 3000 k1=k0+span
      r0=r(1+k0)
      r1=r(1+k1)
      i0=i(1+k0)
      i1=i(1+k1)
      r(1+k0)=r0+r1
      i(1+k0)=i0+i1
      r0=r0-r1
      i0=i0-i1
      r(1+k1)=cos*r0-sin*i0
      i(1+k1)=sin*r0+cos*i0
      k0=k1+span
      if(k0.lt.ninc) go to 3000
      k1=k0-ninc
      cos=-cos
      k0=span-k1
      if(k1.lt.k0) go to 3000
      k0=k0+inc
      k1=span-k0
      if(k0.lt.k1) go to 1500
 2000 continue
      span=span/2
      k0=0
```

```
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))*.5*uuzh(i)
      g4iphjmh=(g4(i+1,j-1)+g4(i,j))*.5*uuzh(i)
      g4imhjph=(g4(i-1,j+1)+g4(i-1,j))*.5*uuzh(i-1)
      g4imhjmh=(g4(i-1,j-1)+g4(i-1,j))*.5*uuzh(i-1)
      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))
```

```
c... (4000=zero)
 4000 k1=k0+span
      r0=r(1+k0)
      r1=r(1+k1)
      i0=i(1+k0)
      i1=i(1+k1)
      r(1+k0)=r0+r1
      i(1+k0)=i0+i1
      r(1+k1)=r0-r1
      i(1+k1)=i0-i1
      k0=k1+span
      if(k0.lt.ninc) go to 4000
      if(span.eq.inc) go to 5000
      k0=span/2
 4500 k1=k0+span
      r0=r(1+k0)
      r1=r(1+k1)
      i0=i(1+k0)
      i1=i(1+k1)
      r(1+k0)=r0+r1
      i(1+k0)=i0+i1
      r(1+k1)=(i1-i0)*sgn
      i(1+k1)=(r0-r1)*sgn
      k0=k1+span
      if(k0.lt.ninc) go to 4500
      k1=inc+inc
      if(span.eq.k1) go to 2000
      c=2.*sines(is)**2
      is=is-1
      sin=sign(sines(is),sgn)
      s=sin
      cos=1.-c
      k0=inc
      go to 3000
c
 5000 n1=ninc-inc
      n2=ninc/2
      ij=0
      ji=0
      rc=0
      if(n2.eq.inc) return
      go to 5020
c... (5010=even)
 5010 ij=n1-ij
      ji=n1-ji
      t=r(1+ij)
      r(1+ij)=r(1+ji)
      r(1+ji)=t
      t=i(1+ij)
      i(1+ij)=i(1+ji)
      i(1+ji)=t
      if(ij.gt.n2) go to 5010
c... (5020=odd)
 5020 ij=ij+inc
      ji=ji+n2
      t=r(1+ij)
      r(1+ij)=r(1+ji)
      r(1+ji)=t
      t=i(1+ij)
      i(1+ij)=i(1+ji)
```

```
       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
      i(1+ji)=t
      it=n2
c... (6000=incrv)
 6000 it=it/2
      rc=rc-it
      if(rc.ge.0) go to 6000
      rc=rc+2*it
      ji=rc
      ij=ij+inc
      if(ij.le.ji) go to 5010
      if(ij.lt.n2) go to 5020
c
      return
      end
      subroutine rpft2(a,b,n,incp)
      real a(1),b(1),ip,im
      inc=incp
      ninc=n*inc
      a(1)=a(1)+a(1)
      b(1)=b(1)+b(1)
      lp=inc
      lm=ninc-lp
c... n=1 or 2 (200=nyq)
      if(lp.ge.lm) go to 200
  100 rp=a(1+lp)
      rm=a(1+lm)
      ip=b(1+lp)
      im=b(1+lm)
      a(1+lp)=-rm+rp
      b(1+lm)=-rm-rp
      b(1+lp)=ip-im
      a(1+lm)=ip+im
      lp=lp+inc
      lm=ninc-lp
c... until lp=lm=n/2
      if(lp.lt.lm) go to 100
c... n=1
  200 if(lp.gt.ninc) return
      a(1+lp)=a(1+lp)+a(1+lp)
      b(1+lp)=b(1+lp)+b(1+lp)
      return
      end
      subroutine rpfti2(a,b,n,incp)
      real a(1),b(1)
      inc=incp
      ninc=n*inc
      lp=inc
      lm=ninc-lp
c... n=1 or 2
      if(lp.ge.lm) return
  100 ca=a(1+lp)
      sb=b(1+lm)
      cb=b(1+lp)
      sa=a(1+lm)
      a(1+lp)=ca-sb
      a(1+lm)=ca+sb
      b(1+lp)=cb+sa
      b(1+lm)=cb-sa
      lp=lp+inc
      lm=ninc-lp
      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-2)*(jx-3)
      k1=.5*(1+isw)*k1i+.5*(1-isw)*k1j
      fac1=-1.
      if(sfj1.eq.1.and.sfizx.eq.1)fac1=1.
      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=-1.
      if(sfjrx.eq.1.and.sfi1.eq.1)fac3=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=1.
      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=-1.
      if(sfjrx.eq.1.and.sfizx.eq.1)fac4=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
      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
      k1=i-1+(j-2)*(ix-2)
```

```
c... until lp=lm=n/2
      if(lp.lt.lm) go to 100
      return
      end
      subroutine cartmm(n,fn,fx,f,inc)
      dimension f(1)
      j=0
      ninc=n*inc
      gn=f(1)
      gx=f(1)
  100 j=j+inc
      if(j.ge.ninc) go to 200
      g=f(1+j)
      if(g.gt.gx) gx=g
      if(g.lt.gn) gn=g
      go to 100
  200 fn=gn
      fx=gx
      return
      end
      subroutine logmm(m,n,fn,fx,f,inc)
      dimension f(1)
c  find maximum
      gx=f(1)
      j=0
      ninc=n*inc
  100 j=j+inc
      if(j.ge.ninc) go to 200
      g=f(1+j)
      if(g.gt.gx) gx=g
      go to 100
c  check to see if max is positive
  200 if(gx.le.0.) go to 500
      an=gx*10.**(-m)
      gn=gx
      j=-inc
  300 j=j+inc
      if(j.ge.ninc) go to 400
      g=f(1+j)
      if(g.le.0.) go to 300
      if(g.le.an) go to 350
      if(g.gt.gn) go to 300
      gn=g
      go to 300
  350 gn=an
      go to 300
  400 fn=gn
      fx=gx
      return
c  no positive values min=max=1.
  500 fn=1.
      fx=1.
      return
      end
      subroutine pack1(a,b,c)
c  pack a and b into c
      c=(a.and.1777777777740000000000b).or.shiftr(b,32)
      return
      end
      subroutine upack1(a,b,c)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 12 m=2,8,3
      a1(k,m)=a1(k,m)+sfizx*a1(k,m+1)
      a2(k,m)=a2(k,m)+sfizx*a2(k,m+1)
      a3(k,m)=a3(k,m)+sfizx*a3(k,m+1)
 12   continue
 20   continue
      i=2
      do 21  j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 21  m=1,7,3
      a1(k,m)=0.
      a2(k,m)=0.
      a3(k,m)=0.
21    continue
      i=ix-1
      do 22 j=2,jx-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 22 m=3,9,3
      a1(k,m)=0.
      a2(k,m)=0.
      a3(k,m)=0.
22    continue
      j=2
      do 31 i=2,ix-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 30 m=4,6
      a1(k,m)=a1(k,m)+sfj1*a1(k,m-3)
      a2(k,m)=a2(k,m)+sfj1*a2(k,m-3)
      a3(k,m)=a3(k,m)+sfj1*a3(k,m-3)
30    continue
      b1(k,2)=b1(k,2)+b1(k,1)
31    continue
32    continue
      j=jx-1
      do 35 i=2,ix-1
      k1=i-1+(j-2)*(ix-2)
      k2=j-1+(jx-2)*(i-2)
      k=.5*(1+isw)*k1+.5*(1-isw)*k2
      do 34 m=4,6
      a1(k,m)=a1(k,m)+sfjrx*a1(k,m+3)
      a2(k,m)=a2(k,m)+sfjrx*a2(k,m+3)
      a3(k,m)=a3(k,m)+sfjrx*a3(k,m+3)
34    continue
      b1(k,2)=b1(k,2)+b1(k,3)
35    continue
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
```

```
c  unpack c into a and b
      a=c.and.1777777777740000000000b
      b=shiftl(c,32)
      return                        /
      end
      subroutine iopack (niowrt,ntrymx,nsus)

      common/q8iocs/ iocs(16)
      dimension arry(2),ibeta(9)
      integer zadjname

      ioerr=niowrt
      itrymax=ntrymx
      nsuspend=nsus
      ibit=0
      ichar=2
      call zmovebit (ibit,58,ichar,58,1)
      ichar=8 - ibit
      iadj=1 + ibit/2
      ibit=56 - ibit

      return

      entry crtdsk (name,nsize,nioc,ndcu,nchs)
      indic=1
      ifunc=6hcreate
      itry=1
      go to 100

      entry opndsk (name,nsize,nioc)
      indic=2
      ifunc=5h open
  100 ishft=zadjname(name,name,iadj)
c.... find an open ioc unit
      nioc=-16
      do 110 i=16,2,-1
      if(iocs(i).eq.0) nioc=i-1
  110 continue

c.... if nioc=-16 no ioc available
      if(nioc.eq.-16) 120,130
  120 write (ioerr,111) ifunc
  111 format ("no ioc available, disk ",a8," inhibited")
      return

  130 go to (140,230) ,indic

  140 ibeta(1)=nioc
      ibeta(2)=name
      ibeta(3)=nsize
      ibeta(4)=0
      ibeta(5)=0
      if(nchs .eq. 0) ibeta(5)=9
      ibeta(6)=0
      ibeta(7)=0
      ibeta(8)=ndcu
     ntmp=izlcreat(ibeta(1),8)
      if(ntmp - 1) 150,160,155
  150 if(ndcu .ne. 0) ndcu=ibeta(8)
```

```
      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=itbw*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(l)-xi(n-1)) , and
c..... rhs2(k)=2*a3*xi(n)-a2*xi(n-1)+b1*(xr(l)-xr(n-1)) .

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

c.... if file index full or no file space , then suspend at 1 min.
c.... intervals and retry , do for a max. of 15 minutes then give up.
  155 if(ntmp - 3) 156,156,370
  156 if(itry - 15) 157,157,370
  157 call suspend (60)
      go to 140

  160 if(nchs) 190,370,190
c.... increment disk file name till name no longer matches files on disk
c....                then go back and create new file with new name
  190 itry=itry+1
      name=itrplnmr(name,nchs,1)
      if(itry-64) 140,140,370

  230 if(izopen(nioc,name,nsize,nacss).ne.0) 370,240
c.... is access level rw(3) or rwe(7) , if not close file
c.... with access level set to rwa(7) , i.e. nacss=7
c.... then reopen file with new access level
  240 if((nacss-3)*(nacss-7)) 250,400,250
  250 nacss=7
      if(izclose(nioc,nacss).ne.0) 370,230

      entry clsdsk (nioc,nacs)
c.... close disk file on ioc unit=nioc
c.... nacs=access level of file just closed
c.... see baselib manual page 8
      ifunc=5hclose
      itry=0
  305 itry=itry + 1
      if(izclose(nioc,nacs).ne.0) 310,315
  310 if(itry - itrymax) 305,305,370
  315 nioc=0
      go to 400

      entry dstdsk (nioc,name)
c.... destroy disk file "name"
c.... nioc set to -1 if error occurred in call
      nioc=1
      ifunc=7hdestroy
      itry=0
  320 itry=itry + 1
      if(izdstroy(name,0).ne.0) 325,400
  325 if(itry - itrymax) 320,320,370

      entry wrsdsk (nioc,arry,nwords,idskaddr)
c.... writes nwords of array arry to file connected to ioc unit=nioc
c....      starting at disk address idskaddr
      ifunc=5hwrite
      itry=0
  335 itry=itry + 1
      if(izdkout(nioc,arry,idskaddr,nwords).ne.0) 340,400
  340 if(itry - itrymax) 345,345,370
  345 if(nsuspend .ne. 0) call suspend (nsuspend)
      go to 335

      entry rdsdsk (nioc,arry,nwords,idskaddr)
c.... reads nwords from disk file connected to ioc unit=nioc
c.... starting at disk address idskaddr into array arry
```