# APPENDICES

# APPENDIX A

# ELLCHI: A PROGRAM FOR THE SIMULATION OF SUPERHUMP LIGHT CURVES

*'Great. What's the name of this cocktail?'*
*'Masetti and sons.'*
(Benni 1996)

## A.1. INTRODUCTION

During the study of the behaviour of X-ray Nova Ophiuchi 1993 (=V2293 Oph; see Ch. 4) in the outburst phase, the problem of interpreting the light curve shape of this object arose out. This point can also be extended to double systems, i.e. the SU UMa-type DNe, which present the superhump phenomenon during their stronger eruptions (the so-called superoutbursts; see Ch. 2).

The main reason which stimulated this search was the fact that, until now, the theoretical studies concerning the superhump phenomenon were devoted to the description of its genesis, its development and its decay (Whitehurst 1988a, 1988b, 1994; Hirose & Osaki 1990, 1993; Whitehurst & King 1991) without making any conjecture on the shape of the light curve, in particular on the reason for its asymmetry.

Therefore, a simplified model of an erupting system with ongoing superhumps has been constructed, so that a physical interpretation to the sawtooth-shaped light curve observed in these cases could be given and the lack of explanations to the light curve modulation asymmetry could be filled in, even if in an approximate way. As it will be clear further on, the described model uses the hypothesis by Vogt (1982) and places

the origin of the superhump modulation on the external edge of the disk, and precisely in the region located around the hot spot.

## A.2. THE ALGORITHM

As already mentioned earlier in Ch. 4 of this Thesis, it has been hypothesized that the superhump is produced by the periodic variation of the distance $R$ from the collapsed object of the impact point on the accretion disk of the gas stream coming from the secondary. All this according to Eq. (1.13) which, as already illustrated, gives the luminosity produced by means of the accretion mechanism.

As it can be easily understood, the distance $R$ is needed to vary cyclically if a periodic modulation is to be obtained, and this is possible only if the disk is elliptic in shape. Moreover, in order to get a superhump period slightly longer than the orbital one, the disk should precess slowly in the same direction of the orbital motion. Both these hypotheses are confirmed by the theoretical studies of superhumps by Whitehurst (1988a, 1988b).

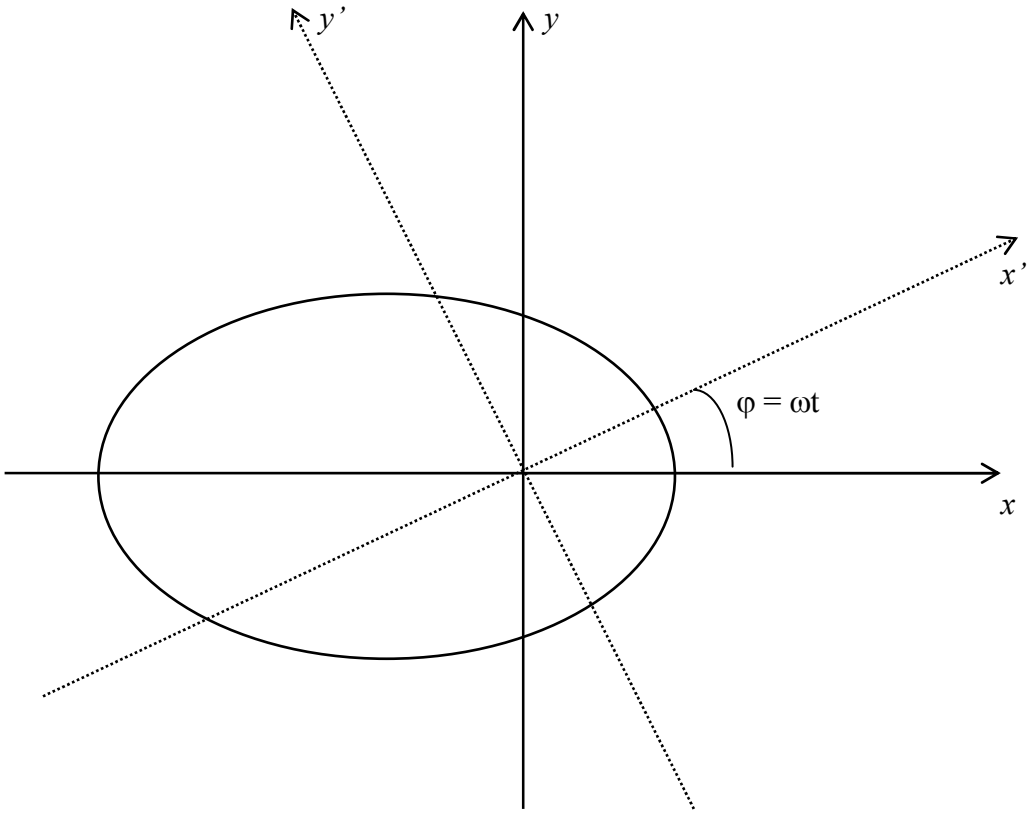


**Fig. A.1.** Geometry of the superhump model described in the text.

Therefore, in order to analyze this variation, the program *ELLCHI* has been written: it computes the distance $R$ in correspondence of 1000 points, equidistant in phase and placed all along the elliptic disk border. Then, the program calculates the intersection between an ellipse with focus coincident with the origin of axes and a parabola branch in uniform rotation with constant angular velocity ω around the origin itself (which is thus the place in which the compact object is located). Clearly, in order to shift from the rotating system *x'y'* to the fixed system *xy* of Fig. A.1 the following coordinate transformation for a rotated system is needed:

$$\begin{cases} x' = x\cos\varphi + y sen\varphi \\ y' = -x sen\varphi + y\cos\varphi \end{cases}. \tag{A.1}$$

Figure A.1 shows the geometry of the problem. We chose a parabola brach to represent the gas stream coming from the secondary because it is the curve that better fits, at a first approximation, the trajectory of a test particle coming from infinity with negligible initial velocity.

The orbital separation $a$ of the system was chosen as measurement unit for the distances. The "string method" (see e.g. Serotti & Sturlese 1984) is used to compute the intersection between the ellipse and the parabola branch, determined by solving the system of equations given by these two curves. This is

$$\begin{cases} \dfrac{x^2}{a^2} + \dfrac{y^2}{b^2} = 1 \\ -x sen\varphi + y\cos\varphi = d(x\cos\varphi + y sen\varphi)^2 + cc \end{cases} \tag{A.2}$$

where φ = ωt represents the angle of which the parabola is rotated, at time t, with respect to the initial position (when the fixed and the rotated systems are coinciding), while $a$, $b$, $d$ and $cc$ are the geometric parameter describing the shapes of the ellipse (the first two) and of the parabola (the other two). Once that the coordinates $x_{int}$ and $y_{int}$ of the intersection are known, its distance from the origin is computed; from it, the accretion luminosity is evaluated by means of Eq. (1.13), here slightly modified in order to take into account that the superhump emission comes only from a fraction

of the disk (i.e. the hot spot region). This modification is such that the **hot spot parameter β** is introduced: this parameter represents the ratio between the luminosity of the stream impact zone and that of the whole accretion disk. Moreover, it was decided that the luminosity value is normalized to the mean luminosity value of the modulation, so that the mean magnitude of the light curve is equal to zero.

The result may then be compared to an observed light curve in order to find the ellipse eccentricity, the phase shift between the synthetic and the observed light curves, the value of β and the mean magnitude; the parameters of the theoretical light curve that better fit the observed data are computed by means of a least-squares method.

## A.3. THE PROGRAM

Going more into the details of the program *ELLCHI*, it can be said that it reads from the ancillary file *ELLIPSE.INPUT* the values of the parabola parameters, the number of bins dividing the abscissa interval over which each intersection is to be found and the uncertainty associated with the coordinates of each intersection. Then, the semimajor axis and the eccentricity of the ellipse, as well as the value of β, the mean magnitude value and the phase shift of the light curve are requested as keyboard input.

After having acquired these data, the program checks if the intersection between the ellipse and the parabola branch is verified (otherwise it requires a modification of the parameters of the curves) and computes it over 1000 different points, equidistant in phase, by rotating the parabola branch by an angle $2\pi/1000$ every new cycle of intersection search. In order to simplfy this task, the search method has been separately applied on each of the four cartesian plane quadrants.

The method used to calculate the intersection is, as said in the previous Section, the "string method": it is based on the computation of the difference between the ordinate values of the two curves (the ellipse and the parabola branch) on the left and right borders of the initial abscissa bin $[x_1, x_2]$; then, the two values are compared.

This procedure is applied until the program finds the bin inside which the difference between the values changes its sign: then, the procedure is repeated on this bin which will be divided into smaller bins until the requested precision is achieved.

Once the values of the intersections have been obtained, for each of them the distance to the origin is computed and, from it, the accretion luminosity (taking also into account the hot spot parameter β) and its corresponding magnitude by using the Pogson's formula

$$\mathrm{m}(R) = -2.5 \cdot \mathrm{Log}\frac{L(R)}{L_{\mathrm{ave}}}. \qquad \text{(A.2)}$$

As the luminosity is normalized to the mean light curve luminosity, the modulation has a mean magnitude equal to zero.

After all this, the program writes 3 output files with names *ELLISSE.TAB*, *PARABOLA.TAB* and *ELLIPSE.TAB*, the first one containing phases and magnitudes of the superhump synthetic light curve, the second the coordinates of the points on the parabola at the initial epoch of computation, and the third the coordinates of the points belonging to the ellipse.

Finally, the program asks if one wants to compare the synthetic light curve with a phase-folded series of observational data points. If so, the synthetic light curve is adapted to the phase shift and magnitude level of the observational one by means of the values for these two quantities as given in input at the beginning of the program. Then, an error parameter from the comparison between the two curves is computed by using a least-squares method. In this way, playing with the values of eccentricity, phase shift, mean magnitude and β it is possible to construct the synthetic light curve which best fits the observational data.

## A.4. THE RESULTS

Thus, by modeling the superhump phenomenon with this simple FORTRAN code, one can obtain theoretic light curves which are qualitatively and quantitatively

similar to the real ones and, by means of a comparison between them, one can extract important parameters concerning the outbursting accretion disk.

In particular, the modulation amplitude $\Delta m$, the light curve asymmetry $\xi$ (that is the phase ratio between the rising and the declining branches of the light curve; see Ch. 2) and the phase shift $\Delta\varphi$ (that is, the distance between the inital light curve phase and that of the minimum light) are affected by 5 parameters: the semimajor axis *a* of the elliptic disk, the hot spot parameter $\beta$, the disk eccentricity *e*, as well as the concavity *d* and the distance *cc* of the vertex from the origin of axes which characterize the parabolic stream of matter.

The effects of each of these 5 quantities on the three main characteristics of the synthetic superhump light curve were then evaluated by varying each one of these single parameters. The results are reported in Table A.I.

**Table A.I.** Summary of the effects produced on the synthetic superhump light curve by varying the parameters describing the ellipse-parabola system geometry and the properties of the elliptic disk

| increasing parameter | $\Delta m$ | $\xi$ | $\Delta\varphi$ |
|---|---|---|---|
| *a* | ↙ | ↙ | ↗ |
| *e* | ↗ | ↗ | ↗ |
| $\beta$ | ↗ | ←→ | ←→ |
| *d* | ←→ | ↙ | ↗ |
| *cc* | ←→ | ↗ | ↗ |

As it can be seen, and as it was expected, the increase of both $\beta$ and *e* makes the light curve amplitude wider, and this because the impact area and the difference between the maximum and minimum distance of it from the origin of axes (where the compact object is located) get larger, respectively. As *a* increases, instead, $\Delta m$ reduces because the disk gets larger and all the impact points are therefore more distant from the origin of axes. The parameters *d* and *cc* does not produce any effect on $\Delta m$. Concerning the light curve asymmetry, it increases as *e* and *cc* increase, while gets smaller as *a* and *d* increase, for geometric reasons in both cases. On the contrary, the parameter $\beta$ does not produce any effect on $\xi$. Finally, the phase shift increases as

each one of these 5 parameters increase but β, which does not have any influence on Δφ.

Therefore, if one examines Table A.I one last time, it can be said that the system parameters can be divided into three categories: *horizontal* parameters, which act on ξ and Δφ and then modify the superhump light curve shape by altering quantities which are measurable along the phase axis (*x* axis); *vertical* parameters, which modify Δm, and then their effect is measurable on the magnitude axis (*y* axis); and *plane* parameters, which can modify the light curve along both axes.

In conclusion, despite the simple idea at the basis of this model, the synthetic light cuves obtained from it are quite similar to the observed ones. It should be noted that this result comes from purely geometric considerations. In any case, the code illustrated here can be improved by introducing other important parameters (such as, for example, the mass ratio $q$, the mass loss rate $\dot{M}$ from the secondary star, the temporal evolution of the superoutburst, etc.) which were not considered in this treatment in order to keep the program reasonably simple.

# REFERENCES OF APPENDIX A


- Hirose M., Osaki Y.,1990, PASJ, 42, 135
- Hirose M., Osaki Y.,1993, PASJ, 45, 243
- Serotti L., Sturlese A., 1984, Introduzione all'informatica - Elementi di calcolo numerico e programmazione. Pitagora Editrice, Bologna *(in italian)*
- Vogt N., 1982, ApJ, 252, 653
- Whitehurst R., 1988a, MNRAS, 232, 35
- Whitehurst R., 1988b, MNRAS, 233, 529
- Whitehurst R., 1994, MNRAS, 266, 35
- Whitehurst R., King A.J., 1991, MNRAS, 249, 25

# ALPHABETICAL LIST OF THE VARIABLES USED IN THE PROGRAM

`a` : semimajor axis of the elliptic disk

`alpha1` : value obtained by inserting in the parabola equation the point with coordinates (`x1`, `y1`) belonging to the ellipse

`alpha2` : value obtained by inserting in the parabola equation the point with coordinates (`x2`, `y2`) belonging to the ellipse

`b` : semiminor axis of the elliptic disk

`beta` : ratio between the hot spot and the disk emissions

`c` : position of the elliptic disk focus occupied by the compact object

`cc` : distance of the parabola vertex from the origin of axes

`chi` : value of the least-squares parameter computed from the comparison between the synthetic and the observed light curves

`conf` : answer to the request of comparison between the synthetic and the observed light curves

`curve` : name of the file containing the observations folded with the superhump period

`d` : parabola concavity index (coefficient of $x^2$)

`delphi` : phase difference between the observed and the synthetic light curves

`delta` : determinant of the equation which allows computing `xint`

`dummy1` : ancillary variable for the `alpha1` value computation

`dummy2` : ancillary variable for the `alpha2` value computation

`dw` : phase increment of the synthetic light curve

`dx` : width of the single abscissa bin inside which the abscissa of the parabola-ellipse intersection point is searched

`dxe` : increment along the abscissas used for computing the ellipse points

`dxp` : increment along the abscissas used for computing the parabola points

`e` : eccentricity of the elliptic disk

`eps` : precision in the computation of the parabola-ellipse intersection

`i` : cycle counter indicating the actual bin in which the parabola-ellipse intersection is searched

`ii` : cycle counter for the comparison between the two light curves (least-squares parameter computation)

`j` : cycle counter for the computation of ellipse and parabola points

`k` : cycle counter for the phase increment of the synthetic light curve

`l` : cycle counter for the comparison between the two light curves

`m` : vector containing the magnitudes of the synthetic light curve

`mag` : magnitude of the synthetic light curve at a given phase `w`

`mmean` : mean magnitude of the observed light curve

`np` : starting number of bins in which is divided the interval of abscissas over which the abscissa of the intersection is searched

`nstep` : actual number of bins in which is divided the interval of abscissas over which the abscissa of the intersection is searched

`ph` : vector reporting the phases of the magnitudes containing in `m`

`rip` : answer to the request of restarting the program

`ro` : distance (at phase `w`) from the origin of axes of the intersection (hot spot location) between the parabola an the ellipse

`twopi` : $2\pi$

`w` : phase of the synthetic light curve

`x` : vector containing the observed magnitudes (column 1 of file `curve`)

`x1` : starting abscissa of bin `dx`

`x2` : ending abscissa of bin `dx`

`x2bis` : ancillary variable for the computation of the precision of the solution abscissa

`xe` : abscissa of a generic point on the ellipse

`xfin` : ending abscissa of the interval over which the abscissa of the intersection is searched

`xin` : starting abscissa of the interval over which the abscissa of the intersection is searched

`xint` : value of the intersection between the parabola and the abscissas axis at a given phase `w`

`xmin` : ancillary variable for the comparison between the synthetic and the observed light curves

`xmed` : (approximated) abscissa of the intersection between parabola and ellipse at phase `w`

`xp` : abscissa of a generic point on the parabola

`y` : vector containing the phases of the observed magnitudes (column 2 of file `curve`)

`y1` : ordinate of the point with abscissa `x1` and located on the ellipse; this value is to be compared with the ordinate of the point of abscissa `x1` and belonging to the parabola

`y11` : ordinate of the point with abscissa `x1` and placed on the ellipse where $y > 0$

`y12` : ordinate of the point with abscissa `x1` and placed on the ellipse where $y < 0$

`y2` : ordinate of the point with abscissa `x2` and located on the ellipse; this value is to be compared with the ordinate of the point of abscissa `x2` and belonging to the parabola

`y21` : ordinate of the point with abscissa `x2` and placed on the ellipse where $y > 0$

`y22` : ordinate of the point with abscissa `x2` and placed on the ellipse where $y < 0$

`ye` : ordinate of a generic point on the ellipse

`ymed` : (approximated) ordinate of the intersection between parabola and ellipse at phase `w`

`yp` : ordinate of a generic point on the parabola

`yy1` : squared value of the ordinate of the ellipse point with abscissa `x1`

`yy2` : squared value of the ordinate of the ellipse point with abscissa `x2`

`yye` : squared value of `ye`

```
      PROGRAM ELLCHI
      real*8 ro,w,a,b,c,cc,d,yy1,alpha1,y11,y12,xin,xfin
      real*8 dx,alpha2,y1,y2,delphi
      real*8 dw,twopi,x1,x2,yy2,y21,y22,xmed,ymed,xint
      real*8 dxp,dxe,xp,yp,xe,yye,ye,dummy1,dummy2,delta
      real*8 mmean,e,x(10000),y(10000),ph(1000),m(1000)
      real*8 chi,eps,x2bis,beta,mag,xmin,w,ro
      character*1 conf,rip,curve*20
      integer nstep,i,k,j,np,l,ii
      parameter(twopi=6.283185307)
2000  open(unit=10,file='ellipse.input',status='old')
      read (10,fmt=*) cc,d,np,eps
      close(unit=10)
      write(*,*) ' '
      write(*,*) 'Insert a, e, beta, magmean & delta phi'
      write(*,*) '(e=0.64 per Whitehurst)'
      write(*,*) ' '
      read(*,*) a,e,beta,mmean,delphi
      write(*,*) ' '
      c=a*e
      if((a-c).lt.cc) then
        write(*,*) ' '
        write(*,*) 'In some parts no intersection'
        write(*,*) 'is found: you should change'
        write(*,*) 'something (a, e, or cc).'
        write(*,*) ' '
        stop
      end if
      b=sqrt(a**2-c**2)
      cc=-cc
      open(unit=1,file='ellisse.tab',status='new')
c--------------------------
      dw=1./1000.d0
      do 10 k=999,0,-1
        w=0.0001+k*dw
        xin=-(a+c)
        xfin=a-c
        nstep=np
5       x1=xin
        do i=1,nstep
          dx=(xfin-xin)/nstep
          x2=x1+dx

c-----------  COMPUTATION OF ALPHA1

          yy1=b**2*(1.-((x1+c)/a)**2)
          if(yy1.lt.0) yy1=0.
            y11=sqrt(yy1)
            y12=-sqrt(yy1)
            if(w.ge.0..and.w.lt.0.25) then
              delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
              if(delta.lt.0) delta=0
              xint=(sin(twopi*w)+sqrt(delta))/
     &           (2.*d*cos(twopi*w)**2)
              if(abs(xint).lt.(a-c)) then
                y1=y11
```

```
            else
                 y1=y12
            end if
          end if
          if(w.ge.0.25.and.w.lt.0.5) then
            delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
            if(delta.lt.0) delta=0
              xint=(-sin(twopi*w)-
     &           sqrt(delta))/(2.*d*cos(twopi*w)**2)
            if(abs(xint).lt.(a+c)) then
              y1=y12
            else
              y1=y11
            end if
          end if
          if(w.ge.0.5.and.w.lt.0.75) y1=y12
          if(w.ge.0.75.and.w.lt.1.) then
            delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
            if(delta.lt.0) delta=0
            xint=(sin(twopi*w)+sqrt(delta))/
     &        (2.*d*cos(twopi*w)**2)
            if(abs(xint).lt.(a-c)) then
              y1=y11
            else
              y1=y12
            end if
          end if
          if(w.lt.0.5) then
            if(y1.lt.-x1/tan(twopi*w)) then
              x1=x1+dx
              if(i.lt.nstep) go to 333
              if(i.eq.nstep) then
                nstep=nstep*10
                go to 5
              end if
            end if
          end if
          if(w.ge.0.5.and.w.lt.1.) then
            if(y1.gt.-x1/tan(twopi*w)) then
              x1=x1+dx
              if(i.lt.nstep) go to 333
              if(i.eq.nstep) then
                nstep=nstep*10
                go to 5
              end if
            end if
          end if
          dummy1=(y1*cos(twopi*w)-x1*sin(twopi*w)-cc)/d
          if(dummy1.lt.0.) dummy1=0.
          alpha1=(x1*cos(twopi*w)+y1*sin(twopi*w))-
     &        sqrt(dummy1)
          if (alpha1.eq.0.) then
            xmed=x1
            ymed=y1
            go to 1000
          end if
```

```
c---------  COMPUTATION OF ALPHA2

              yy2=b**2*(1.-((x2+c)/a)**2)
              if(yy2.lt.0) yy2=0.
              y21=sqrt(yy2)
              y22=-sqrt(yy2)
              if(w.ge.0..and.w.lt.0.25) then
                delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
                if(delta.lt.0) delta=0
                xint=(sin(twopi*w)+sqrt(delta))/
     &            (2.*d*cos(twopi*w)**2)
                if(abs(xint).lt.(a-c)) then
                  y2=y21
                else
                  y2=y22
                end if
              end if
              if(w.ge.0.25.and.w.lt.0.5) then
                delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
                if(delta.lt.0) delta=0
                xint=(-sin(twopi*w)-
     &            sqrt(delta))/(2.*d*cos(twopi*w)**2)
                if(abs(xint).lt.(a+c)) then
                  y2=y22
                else
                  y2=y21
                end if
              end if
              if(w.ge.0.5.and.w.lt.0.75) y2=y22
              if(w.ge.0.75.and.w.lt.1.) then
                delta=sin(twopi*w)**2-4.*cc*d*cos(twopi*w)**2
                if(delta.lt.0) delta=0
                xint=(sin(twopi*w)+sqrt(delta))/
     &            (2.*d*cos(twopi*w)**2)
                if(abs(xint).lt.(a-c)) then
                  y2=y21
                else
                  y2=y22
                end if
              end if
              dummy2=(y2*cos(twopi*w)-x2*sin(twopi*w)-cc)/d
              if(dummy2.lt.0.) dummy2=0.
              alpha2=(x2*cos(twopi*w)+y2*sin(twopi*w))-
     &          sqrt(dummy2)
              if (alpha2.eq.0.) then
                xmed=x2
                ymed=y2
                go to 1000
              end if

c---------  COMPARISON ALPHA1-ALPHA2

              if(y2.ge.0) then
                if(alpha2.gt.0..and.alpha1.lt.0..or.
     &            alpha2.lt.0..and.alpha1.gt.0.) then
```

```
                if(x2.eq.0) then
                   x2bis=(x1+x2)/2.
                else
                   x2bis=x2
                end if
                if (dabs((x2-x1)/x2bis).le.eps) goto 999
                xin=x1
                xfin=x2
                goto 5
              else if (alpha2.lt.0..and.alpha1.lt.0.) then
                if(i.eq.nstep) then
                   nstep=nstep*10
                   go to 5
                end if
                x1=x2
              else if (alpha2.gt.0..and.alpha1.gt.0..
     &             and.i.eq.nstep) then
                nstep=nstep*10
                go to 5
              endif
            else if(y2.lt.0) then
              if(alpha2.gt.0..and.alpha1.lt.0..
     &             or.alpha2.lt.0..and.alpha1.gt.0.) then
                if(x2.eq.0) then
                   x2bis=(x1+x2)/2.
                else
                   x2bis=x2
                end if
                if (dabs((x2-x1)/x2bis).le.eps) goto 999
                xin=x1
                xfin=x2
                goto 5
              else if (alpha2.gt.0..and.alpha1.gt.0.) then
                if(i.eq.nstep) then
                   nstep=nstep*10
                   go to 5
                end if
                x1=x2
              else if (alpha2.lt.0..and.alpha1.lt.0..
     &             and.i.eq.nstep) then
                nstep=nstep*10
                go to 5
              end if
            end if
333       end do
c---------------------------
999       xmed=(x1+x2)/2.
          ymed=(y1+y2)/2.
1000      ro=abs(sqrt(xmed**2+ymed**2))
          mag=-2.5*log10(1.-beta+
     &      beta*(a/(2*c))*(b**2/(2*a*ro)-1.))
          mag=mag+2.5*log10(1.-beta-beta*(a/(4*c)))
          write(1,'(1x,f6.4,1x,f10.5)') 1.-w,mag
          nstep=np
10     continue
c---------------------------
```

```
      close(unit=1)
      dxp=3./2.*(a+c)/1000.
      open(unit=2,file='parabola.tab',status='new')
      do j=0,2000
        xp=j*dxp
        yp=d*xp**2+cc
        write(2,'(1x,f10.5,2x,f10.5)') xp,yp
      end do
      close(unit=2)
      dxe=2.*a/1000.
      open(unit=3,file='ellisse.tab',status='new')
      do j=0,1000
        xe=-(a+c)+j*dxe
        yye=b**2*(1.-((xe+c)/a)**2)
        if(yye.lt.0) yye=0
        ye=sqrt(yye)
        write(3,'(1x,f10.5,2x,f10.5,2x,f10.5)') xe,ye,-ye
      end do
      close(unit=3)
      write(*,*) ' '
      write(*,*) 'COMPARISON WITH DATA POINTS? (y/n)'
      read (*,'(a)') conf
      if(conf.ne.'y') go to 3000
      write(*,*) 'Insert the data points file name'
      write(*,*) ' '
      read (*,'(a)') curve
      open(unit=20,file=curve,status='old')
      open(unit=21,file='ellipse.tab',status='old')
      do l=1,1000
        read(21,fmt=*) ph(l),m(l)
        m(l)=m(l)+mmean
        ph(l)=ph(l)+delphi
        if(ph(l).lt.0.) ph(l)=ph(l)+1.
        if(ph(l).gt.1.) ph(l)=ph(l)-1.
      end do
      do l=1,10000
        read(20,fmt=*,end=1999) x(l),y(l)
        do ii=1,1000
          xmin=abs(x(l)-ph(ii))
          if(xmin.lt.0.0005) then
            chi=chi+(y(l)-m(ii))**2
            go to 5000
          end if
        end do
5000  end do
1999  close(unit=20)
      close(unit=21)
      write(*,*) ' '
      write(*,'(a,f10.5)') ' Chi squared =', chi
      chi=0
      write(*,*) ' '
      write(*,*) 'ONCE MORE? (y/n)'
      write(*,*) ' '
      read (*,'(a)') rip
      if(rip.eq.'y') go to 2000
3000  continue
```

```
write(*,*) ' '
write(*,*) 'BYE BYE!!!!!!!!!!!!'
stop
end
```