APPENDIX 1

A computer program in Fortran 77, which solves the system of ordinary differential equations. It calculates the evolution of φ and B contained in it from inflationary stage of the Universe till B-conservation epoch. The Runge-Kutta 4th order method is used and the exact routine is taken from [77].

```
      PROGRAM BARYON NUMERICALLY CALCULATED GAMA

      INTEGER*4 N,J,s,z,w
      PARAMETER (N=4)
      REAL*8 x,h,y(N),dydx(N),yout(N),yold(N),ymax(N),Au(3),Av(3)
      REAL*8 lam1,lam2,lam3,a,mH,b,field,omu,omv,p
      LOGICAL kraj1,kraj2
      EXTERNAL derivs,rk4,omega_u,omega_v


      OPEN(2,file='num_m350_h9.dat')
      WRITE(*,*) 'CALCULATIONS IN PROGRESS...'

c     step size
      h=dble(1.e-6)

c     parameters
      x=dble(2.)
      lam1=dble(0.01)
      lam2=dble(0.0001)
      lam3=lam2
      a=dble(0.001)
      mh=dble(3.5e-10)

c     angle
      p=dble(30.)
      p=p*acos(dble(-1.))/dble(180.)

c     initial conditions
      y(1)=dble(lam1+lam2)**(-0.25)*dble(2.)**(-0.25)*dcos(p)
      y(2)=(dble(3.)/dble(2.)**(1.5))+y(1)
      y(3)=dble(lam1+lam2)**(-0.25)*dble(2.)**(-0.25)*dsin(p)
      y(4)=(dble(3.)/dble(2.)**(1.5))+y(3)

      omu=sqrt(lam1)*sqrt(y(1)**(2.))/dble(2.)
      omv=sqrt(lam1)*sqrt(y(3)**(2.))/dble(2.)

c     counters
      s=0
      z=2
      w=2
      kraj1=.false.
      kraj2=.false.

c     initial values of the massives
      do 20 j=1,3
         Au(j)=dble(0.)
         Av(j)=dble(0.)
```

```
20    continue

      do 30 j=1,4
         yold(j)=y(j)
         ymax(j)=y(j)
30    continue

c     initial values of the Baryon charge and the scalar field
      b=dble(2.)*(y(2)*y(3)-y(4)*y(1))
      field=dble(4.)*sqrt(y(1)*y(1)+y(3)*y(3))/x*x

      WRITE(2,'(2x,f15.6,2x,f15.6,2x,f17.10,2x,f17.10,2x,f17.10,
     *  2x,f17.10,2x,f17.10)') x,field,B,y(1),y(3),omu,omv

c     the program is going to stop when m^2*fi^2 become compatible with
c     lam1*fi^4/2
      do 10 while ((mh**(2.)*x**(4.))/(dble(8.)*lam1*ymax(1)**(2.))
     *      .LE.dble(1.0))

c     new step
       x=x+h
       s=s+1

c     call procedure for calculating the differencial equations
      call derivs(x,y,dydx,omu,omv)
      call rk4(y,dydx,n,x,h,yout,derivs,omu,omv)

c     finding maximum of y1 and call procedure for calculating
c     omega_u
      if (yout(2)*y(2).LE.dble(0.).AND.dydx(2).LE.dble(0.)) then
         ymax(1)=yout(1)
         call omega_u(x,omu,z,Au,kraj1)
      endif

c     call procedure for calculating omega_v
      if (yout(4)*y(4).LE.dble(0.).AND.dydx(4).LE.dble(0.)) then
         call omega_v(x,omv,w,Av,kraj2)
      endif

c     send back program to start calculation with the new omega
      if (kraj1.AND.kraj2) then
       z=1
       w=1
       x=Au(2)
       s=int((x-int(x))/h)
      endif

c     calculating and saving B and fi only for each 1/h step
        if (z.LE.2.AND.w.LE.2.AND.s.GE.dble(1.)/h) then

           b=dble(2.)*(yout(2)*yout(3)-yout(4)*yout(1))
           field=dble(4.)*sqrt(yout(1)*yout(1)+yout(3)*yout(3))/x*x
           WRITE(2,'(2x,f15.6,2x,f15.6,2x,f17.10,2x,f17.10,2x,f17.10,
        *  2x,f17.10,2x,f17.10)') x,field,B,y(1),y(3),omu,omv
           s=0
        endif
```

```
c     preparing for a new step
35    do 40 j=1,4
             yold(j)=y(j)
             y(j)=yout(j)
40    continue

10    continue

c     saving final results
        WRITE(2,'(2x,f15.6,2x,f15.6,2x,f17.10,2x,f17.10,2x,f17.10,
     *  2x,f17.10,2x,f17.10)') x,field,B,y(1),y(3),omu,omv

c     saving all parameters at the end of the data file
      WRITE(2,*) 'lam1=',lam1, '   lam2=',lam2
      WRITE(2,*) 'step=',h, '    alfa=',a, '    m/H=',mH
      WRITE(2,*) 'eta=',x, '  field=',field, '  B=',B
      WRITE(2,*) 'NUMERICALLY CALCULATED OMEGA'
      WRITE(*,*) 'END OF CALCULATION PROCESS'


      STOP
      END

c     calculating derivatives procedure
      SUBROUTINE derivs(x,y,dydx,omu,omv)
      REAL*8 x,y(*),dydx(*),lam,lamp,lam1,lam2,lam3,a
      REAL*8 omu,omv,mH

c     the same parameters as in the main program
      mH=dble(3.5e-10)
      lam1=dble(0.01)
      lam2=dble(0.0001)
      lam3=lam2
      a=dble(0.001)
      lam=lam1+lam2
      lamp=lam1-dble(3.)*lam2

c     differencial equations
      dydx(1)=y(2)
      dydx(3)=y(4)
      dydx(2)=dble(2.)*y(1)/x**(2.)-(lam+lam3)*y(1)**(3.)
     *  -lamp*y(1)*y(3)**(2.)
     *  -mH**(2.)*x**(4.)*y(1)
     *  -dble(0.75)*a*omu*(y(2)-dble(2.)*y(1)/x)
      dydx(4)=dble(2.)*y(3)/x**(2.)-(lam-lam3)*y(3)**(3.)
     *  -lamp*y(3)*y(1)**(2.)
     *  -mH**(2.)*x**(4.)*y(3)
     *  -dble(0.75)*a*omv*(y(4)-dble(2.)*y(3)/x)

      RETURN
      END

c     calculating omega_u procedure
        SUBROUTINE omega_u(x,omu,z,Au,kraj1)
        REAL*8 x,omu,Au(3)
        INTEGER z
        LOGICAL kraj1
```

```
          Au(z)=x
          if (z/3.EQ.1) then
        omu=dble(2.)*acos(dble(-1.))/(dble(1.)*(Au(3)-Au(2)))
        kraj1=.true.
          else
        z=z+1
        kraj1=.false.
        endif
          return
        END

c     calculating omega_v procedure
        SUBROUTINE omega_v(x,omv,w,Av,kraj2)
        REAL*8 x,omv,Av(21)
        INTEGER w
        LOGICAL kraj2
          Av(w)=x
        if (w/3.EQ.1) then
        omv=dble(2.)*acos(dble(-1.))/(dble(1.)*(Av(3)-Av(2)))
        kraj2=.true.
        else
          w=w+1
        kraj2=.false.
        endif
        return
        END

c     Runge-Kutta 4 method
      SUBROUTINE rk4(y,dydx,n,x,h,yout,derivs,omu,omv)
       INTEGER n,NMAX
       REAL*8 h,x,dydx(n),y(n),yout(n),omu,omv
       EXTERNAL derivs
       PARAMETER (NMAX=50)
       INTEGER i
       REAL*8 h6,hh,xh,dym(NMAX),dyt(NMAX),yt(NMAX)
       hh=h*dble(0.5)
       h6=h/dble(6.)
       xh=x+hh
       do 11 i=1,n
         yt(i)=y(i)+hh*dydx(i)
11     continue
       call derivs(xh,yt,dyt,omu,omv)
       do 12 i=1,n
         yt(i)=y(i)+hh*dyt(i)
12     continue
       call derivs(xh,yt,dym,omu,omv)
       do 13 i=1,n
         yt(i)=y(i)+h*dym(i)
         dym(i)=dyt(i)+dym(i)
13     continue
       call derivs(x+h,yt,dyt,omu,omv)
       do 14 i=1,n
         yout(i)=y(i)+h6*(dydx(i)+dyt(i)+dble(2.)*dym(i))

14    continue
        RETURN
        END
```