Brachystochrone

James Emery

11/12/92, (Last edit 11/9/13)

Contents

1 The Curve of Fastest Descent

We are to find the Brachystochrone curve, which is the path of fastest descent under gravity from one point to a lower point. Let a mass slide on a frictionless wire under the force of gravity. The path starts at $(0,0)$ and ends at (x_f, y_f). Energy conservation gives

$$\frac{1}{2}mv^2 = mgy,$$

where v is the velocity, m is the mass, and g is the acceleration of gravity.

Then

$$v = \sqrt{2gy}.$$

$$dt = \frac{ds}{v}$$

$$t = \int_0^{x_1} \frac{ds}{v} = \int_0^{x_1} \frac{\sqrt{1+y'^2}dx}{\sqrt{2gy}}.$$

The integrand is

$$F(x,y,y') = \frac{\sqrt{1+y'^2}dx}{\sqrt{2gy}}.$$

From the calculus of variation, a necessary condition for a minimum time is that Euler's equation holds. Euler's equation is

$$\frac{\partial}{\partial x}\frac{\partial F}{\partial y'} - \frac{\partial F}{\partial y} = 0$$

$$\frac{\partial F}{\partial y'} = \frac{1}{\sqrt{2gy}}\frac{y'}{\sqrt{1+y'^2}}$$

We have

$$\frac{d}{dx}(F - y'\frac{\partial F}{\partial y'}) =$$

$$y'\frac{\partial F}{\partial y} + \frac{\partial F}{\partial y'}y'' - y''\frac{\partial F}{\partial y'} - y'\frac{d}{dx}\frac{\partial F}{\partial y'} =$$

$$y'(\frac{\partial F}{\partial y} - \frac{d}{dx}\frac{\partial F}{\partial y'}) = 0.$$

So

$$F - y'\frac{\partial F}{\partial y'} = c.$$

where c is a constant. Then

$$\frac{\sqrt{1+y'2}}{\sqrt{2gy}} - \frac{y'2}{\sqrt{2gy}}\frac{1}{\sqrt{1+y'^2}} = c.$$

Simplifying

$$1 = c\sqrt{2gy(1+y'^2)}.$$

Squaring,

$$1 = c^2(2gy(1+y'^2)).$$

$$1/(c^2 2gy) - 1 = y'^2.$$

$$y'^2 = \frac{1-2c^2gy}{2c^2gy}.$$

So

$$\frac{dy}{dx} = \sqrt{\frac{k-y}{y}}$$

for some new constant k. Then

$$dx = \frac{y}{\sqrt{ky - y^2}} dy.$$

$$x = \int \frac{y}{\sqrt{ky - y^2}} dy.$$

$$= -\sqrt{ky - y^2} + \frac{k}{2} \arccos(\frac{k - 2y}{k}) + M,$$

where M is a constant. When $y = 0$, then $x = 0$, so M is zero. Let

$$\arccos(1 - (2y/k)) = \theta.$$

Then

$$y = (k/2)(1 - \cos(\theta).$$

$$x = \frac{k}{2}(\theta - \sqrt{2(2y/k) - (2y/k)^2})$$

$$= \frac{k}{2}(\theta - \sqrt{1 - (1 - (2y/k))^2})$$

$$= \frac{k}{2}(\theta - \sqrt{1 - \cos^2(\theta)})$$

$$= \frac{k}{2}(\theta - \sin(\theta))$$

This is a cycloid, the curve generated by a point on a circle rolling on a line. The radius of the circle is $r = k/2$. Given x and y, then k and θ are determined by these two nonlinear equations. We may use Newton's method. The parametric equations of our cycloid are

$$x = r(\theta - \sin(\theta)),$$

and

$$y = r(1 - \cos(\theta)).$$

Suppose x and y are given, then dividing x by y, we get an equation

$$x/y = \frac{1 - \cos(\theta)}{\theta - \sin(\theta)},$$

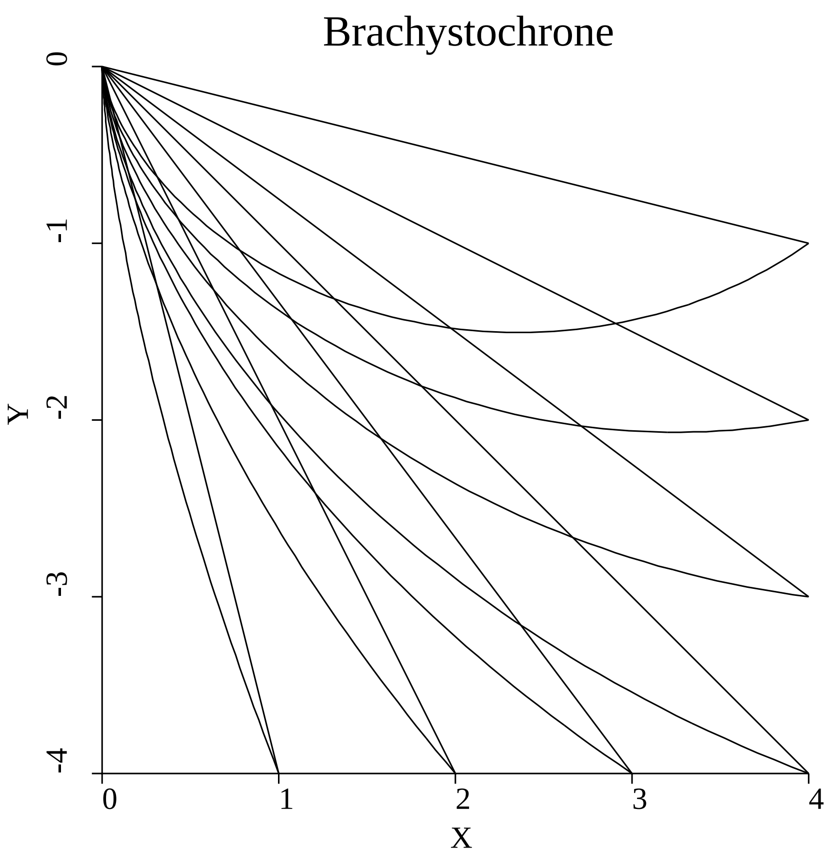

Figure 1: Cycloid curves of quickest descent, ending respectively at (1,4),(2,4),(3,4),(4,4),(4,3),(4,2), and (4,1).

which may be solved numerically for θ. Solving for θ is equivalent to finding a zero of the function

$$f(\theta) = \frac{1-\cos(\theta)}{\theta - \sin(\theta)} - \frac{x}{y}.$$

We have

$$\frac{df}{d\theta} = 1 - \frac{\theta - \sin^2(\theta)}{(1-\cos(\theta))^2}.$$

We shall find the zero by using Newton's iteration. This is

$$\theta_{n+1} = \theta_n - \frac{f(\theta_n)}{f'(\theta_n)}.$$

We construct a Fortran program that reads a point (x_f, y_f) and finds an r, defining a cycloid that passes through the origin and point (x_f, y_f). It also finds the angle parameter θ_f of (x_f, y_f). Points on the cycloid are written to a file. These points may be plotted or imported to a CAD system using IGES (for original program on a unix machine).

The program given here is for a PC, using the Microsoft Fortran Power Station Compiler.

```
c       cycloid.ftn computes a cycloid curve starting at (0,0) and
c       ending at a specified point.  Written for Danielle H Kay in 1992
c       for a student demonstration.
c       Brachystochrone, see bra.tex
c original April 1992
c modified 4/3/2008,
c modified 11/7/2013
c There may be an original silicon graphics version different from this one somewhere.
        implicit real*8(a-h,o-z)
        dimension ain(5)
one=1
pi=4.*atan(one)
        open(2,file='p.eg',status='unknown')
        write(*,*)' See the document Brachistocrone, bra.pdf'
        write(*,*)' Writes curve points to p.eg '
        write(*,*)' The cycloid lies in the first quadrant,'
        write(*,*)' but is flipped when printed to p.eg '
        write(*,*)' so that the curves slope downward'
        write(*,*)' Use pltmerge to define plot window.'
        write(*,*)' pltmerge p.eg q.eg -w'
        write(*,*)' Use pltax to add axes and labels.'
        write(*,*)' pltax q.eg p.eg x y Cycloids '
        write(*,*)' Use eg2ps to make postscript file'
        write(*,*)' eg2ps p.eg p.ps'
        write(*,*)
        write(*,*)' Starting curve point is (0,0)'
        do
        write(*,*)' Enter the ending point of curve, x,y (x>0 y>0)'
```

```
        write(*,*)' <return> to end. '
        call readr(0,ain,nr)
        if(nr .le. 0)exit
        x=ain(1)
        y=ain(2)
        w = x/y
        write(*,*)x, y
        n = 20
        t = w
        if( w .gt.  6.d0) then
            t = 6
        endif
        if (w .lt.  .01d0) then
           t = 1./100
        endif
c        write(*,*)' t= ',t
        do  i = 1, n
c         write(*,*)' i= ',i
         f = (t - sin(t)) / (1 - cos(t)) - x / y
         a = 1 - cos(t)
         df = (a * a - (t - sin(t)) * sin(t)) / (a * a)
         t = t - f / df

        end do
        write(*,*)'t=',t
        r = y / (1 - cos(t))
        write(*,*)' theta= ', t * 180. / pi, ' r= ', r
       write(*,*)' x= ', x, ' cx=',r * (t - sin(t))
       write(*,*)' y= ', y, ' cy=',r * (1 - cos(t))
        g=9.81
        tcd=sqrt(r/g)*t
        write(*,*)' Time of descent on cycloid = ', tcd
        tld=sqrt(2.*(x**2+y**2)/(y*g))
         write(*,*)' Time of descent on line = ', tld
         write(2,'(1x,a,2(1x,g15.8))')'m',0,0
         write(2,'(1x,a,2(1x,g15.8))')'d',x,-y
         n=100
         do  i=1,n
          tp=(i-1)*t/(n-1)
          if( i .eq. 1)then
           x=r*(tp -sin(tp))
           y=-r*(1.-cos(tp))
           write(2,'(1x,a,2(1x,g15.8))')'m',x,y
          else
           x=r*(tp -sin(tp))
           y=-r*(1.-cos(tp))
           write(2,'(1x,a,2(1x,g15.8))')'d',x,y
          endif
         end do
        end do
        end
c+ readr read a row of floating point numbers
      subroutine readr(nf, a, nr)
      implicit real*8(a-h,o-z)
c   numbers are separated by spaces
c   examples of valid numbers are:
c   12.13 34 45e4 4.78e-6 4e2,5.6D-23,10000.d015
```

```
c  nf=file number, 0 for standard input file
c   a=array of returned numbers
c  nr=number of values in returned array,
c     or 0 for empty or blank line,
c     or -1 for end of file on unit nf.
c requires functions val and length
      dimension a(*)
      character*200  b
      character*200  c
      character*1 d
        c=' '
      if(nf.eq.0)then
         read(*,'(a)',end=99)b
      else
         read(nf,'(a)',end=99)b
      endif
      nr=0
      l=lenstr(b)
      if(l.ge.200)then
        write(*,*)' error in readr subroutine '
        write(*,*)' record is too long '
      endif
      do 1 i=1,l
       d=b(i:i)
       if (d.ne.' ') then
        k=lenstr(c)
        if (k.gt.0)then
           c=c(1:k)//d
        else
           c=d
        endif
       endif
       if( (d.eq.' ').or.(i.eq.l)) then
        if (c.ne.' ') then
          nr=nr+1
           call valsub(c,a(nr),ier)
          c=' '
        endif
       endif
1     continue
      return
99    nr=-1
      return
      end
c
c+ lenstr   nonblank length of string
      function lenstr(s)
c  length of the substring of s obtained by deleting all
c  trailing blanks from s.  thus the length of a string
c  containing only blanks will be 0.
      character   s*(*)
      lenstr=0
      n=len(s)
      do 10 i=n,1,-1
      if(s(i:i) .ne. ' ')then
         lenstr=i
         return
```

```
      endif
10    continue
      return
      end
c+ valsub converts string to floating point number (r*8)
      subroutine valsub(s,v,ier)
      implicit real*8(a-h,o-z)
c     examples of valid strings are: 12.13 34 45e4 4.78e-6 4E2
c     the string is checked for valid characters,
c     but the string can still be invalid.
c      s-string
c      v-returned value
c      ier- 0 normal
c           1 if invalid character found, v returned 0
c
      logical p
      character s*(*),c*50,t*50,ch*15
      character z*1
      data ch/'1234567890+-.eE'/
      v=0.
      ier=1
      l=lenstr(s)
      if(l.eq.0)return
      p=.true.
      do 10 i=1,l
      z=s(i:i)
      if((z.eq.'D').or.(z.eq.'d'))then
          s(i:i)='e'
      endif
      p=p.and.(index(ch,s(i:i)).ne.0)
 10   continue
      if(.not.p)return
      n=index(s,'.')
      if(n.eq.0)then
          n=index(s,'e')
          if(n.eq.0)n=index(s,'E')
          if(n.eq.0)n=index(s,'d')
          if(n.eq.0)n=index(s,'D')
          if(n.eq.0)then
             s=s(1:l)//'.'
          else
            t=s(n:l)
            s=s(1:(n-1))//'.'//t
          endif
          l=l+1
      endif
      write(c,'(a30)')s(1:l)
      read(c,'(g30.23)')v
      ier=0
      return
      end
```

2 The Time of Descent

We have

$$x = r(\theta - \sin(\theta)), y = r(1 - \cos(\theta)).$$

Then

$$\frac{dx}{d\theta} = y,$$
$$dx = y d\theta,$$

and

$$y' = \frac{dy}{dx} = \frac{dy}{d\theta}\frac{d\theta}{dx} = \frac{r\sin(\theta)}{y} = \frac{\sin(\theta)}{1 - \cos(\theta)}.$$

Then the time of descent on the cycloid is

$$t = \int_0^{\theta_f} \sqrt{\frac{1 + y'^2(\theta)}{2gy(\theta)}} y(\theta) d\theta$$

$$= \int_0^{\theta_f} \sqrt{\frac{(1 + y'^2(\theta))y(\theta)}{2g}} d\theta.$$

We have

$$yy'^2 = r(1 - \cos(\theta))\frac{\sin^2(\theta)}{(1 - \cos(\theta))^2}$$

$$= r\frac{1 - \cos^2(\theta)}{1 - \cos(\theta)} = r(1 + \cos(\theta)).$$

So the integrand becomes

$$\sqrt{\frac{r}{g}}.$$

The descent time is

$$t = \sqrt{\frac{r}{g}}\theta_f.$$

Let us now find the time of descent on a straight line path. The acceleration a, in the line direction, is constant. We have

$$a = \sin(\phi) g = \frac{y_f}{\sqrt{x_f^2 + y_f^2}} g,$$

where ϕ is the angle between the line and the horizontal. Then the distance traveled in time t along the line is

$$s = \frac{at^2}{2}.$$

We have

$$s = \sqrt{x_f^2 + y_f^2}.$$

Then the descent time is

$$t = \sqrt{\frac{2(x_f^2 + y_f^2)}{y_f g}}.$$

3 Run Example

I think the program was run on a Silicon Graphics Unix Workstation.

```
% ftn8 cycloid
no errors, no warnings, no informational messages,
Fortran 77 compiler 68K Rev 10.8(190) 1992/11/12 17:50:13 CDT (Thu)
All Globals are resolved.
% a.out
  writes points to p.cd
  enter point, <return> to end
2.2 1.2
 2.200000000000000 1.200000000000000
  theta=  193.6277113332691 r=  0.6085664327313063
  x=  2.200000000000000 2.200000000000000
  y=  1.200000000000000 1.200000000000000
  time of cycloid descent =  0.8421429774116072 seconds
  time of line descent =  1.033454019724319 seconds
  write points?
y
```