18

The Catenary

Why Study the Catenary?

- **Structural Engineering and Architecture**. The catenary is the optimal shape for a hanging cable and, when inverted, forms an arch that carries loads purely in compression, making it strong. Suspension bridge cables, power lines, and railway overhead wires naturally form catenaries, and the catenary is an ideal road surface for bicycles with square wheels.

- **Historical and Mathematical Significance**. The catenary played a key role in the development of early calculus and provides a natural geometric example of hyperbolic functions via $y = a\cosh(x/a)$. The catenary has unusual connections to other important curves such as the parabola and the tractrix.

- **Differential Geometry and Curvature**. Rotating a catenary produces the *catenoid*, the simplest nontrivial minimal surface, connecting the curve to curvature theory. Soap-film catenoids are used to study stability transitions (where a catenoid "snaps" into two disks), and the resulting mathematics can be utilized for investigating surface tension, Plateau's laws, and capillary action.

18.1 The Catenary Chain

The catenary is the shape of a perfectly flexible hanging chain, with its mass uniformly distributed along its length. It looks very much like a parabola, and was thought to be one by Galileo, although some authors argue that he only believed

it was 'approximately' parabolic. In any case, that belief was disproved by the 17-year-old Huygens in 1646, who showed that the parabola was the shape assumed by a flexible cord loaded by weights uniformly distributed in the horizontal direction. Fig. 18.1 demonstrates the difference between the two, and the curves are admittedly very similar.

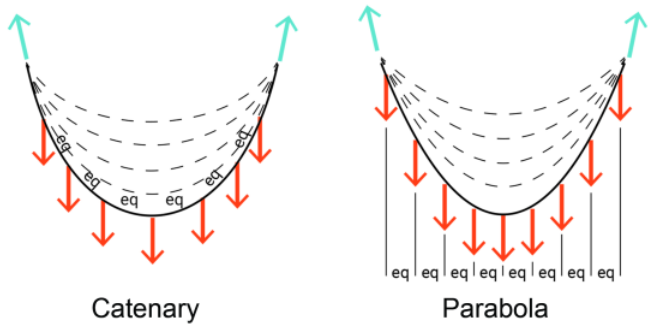


Figure 18.1. The Distribution of Weights in a Catenary and a Parabola

An equation for the catenary was proposed independently by Johann Bernoulli, Huygens, and Leibniz in 1691, in response to a challenge from Jakob Bernoulli the previous year. Johann showed that the curve satisfies the differential equation $dy/dx = s/a$ where a is a constant derived from the chain's weight per unit length, and s is its arc length. He also rewrote it as the integral

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

This was as far as matters could be taken at the time, since x is a transcendental function of y. Nowadays, it can be solved

$$y = a\cosh(x/a),$$

or written in terms of cosh()'s exponential definition,

$$y = a(e^{x/a} + e^{-x/a})/2.$$

The curve was first called a catenaria by Huygens in a letter to Leibniz, based upon the Latin word *catena* for 'chain', and the anglicized equivalent, 'catenary', is due to the (future) US President Thomas Jefferson, coined in a letter to Thomas Paine in 1788, about the design of a bridge.

Two classic books on curves which discuss the catenary are *A Book of Curves* by Lockwood [**Loc67**] and *A Handbook on Curves and Their Properties* by Yates [**Yat47**]. Details with an engineering maths emphasis can be found in chapter 7

of *How Round Is Your Circle?* by Bryant and Sangwin [**BS24**], and the catenary is placed in its historical context by Stillwell in [**Sti10**].

18.1.1 Why cosh() **and** sinh()**?** In 1757, Vincenzo Riccati introduced the use of hyperbolic functions to obtain the roots of certain equations, particularly cubics. Soon after, in 1759, Daviet de Foncenex showed how to interchange circular and hyperbolic functions by using i:

$$\sin\theta = -i\sinh i\theta, \qquad \text{and} \qquad \cos\theta = \cosh i\theta.$$

In a 1768 paper, Johann Heinrich Lambert adopted the notation we use today, including sinh and cosh, and helped popularize hyperbolic trigonometry.

cosh() and sinh() play a similar role for hyperbolas that cos() and sin() satisfy for circles. A unit circle, $x^2 + y^2 = 1$, can be parameterized using $x = \cos t$ and $y = \sin t$ since these trigonometry functions satisfy both

$$\cos^2 t + \sin^2 t = 1,$$

and the differential equation

$$y'' + y = 0.$$

A unit hyperbola has the form $x^2 - y^2 = 1$, and the question arises as to what functions can be used to parameterize it in a similar manner. The hyperbolic functions are the answer:

$$\cosh t = \frac{e^t + e^{-t}}{2} \qquad \text{and} \qquad \sinh t = \frac{e^t - e^{-t}}{2},$$

linked by

$$\cosh^2 t - \sinh^2 t = 1,$$

Their derivatives,

$$\frac{d}{dx}\cosh x = \sinh x \qquad \text{and} \qquad \frac{d}{dx}\sinh x = \cosh x,$$

can be utilized to solve the hyperbolic analogue of the circular differential equation:

$$y'' - y = 0$$

sinhCosh.py plots cosh(x), sinh(x), e^x, and e^{-x} (see Fig. 18.2) to illustrate the relationships between the curves: sinh(x) is half the difference of e^x and e^{-x} and cosh(x) is their average.

For more on the hyperbolic functions, any good book on calculus, such as Thomas and Finney [**TFW96**] will be helpful. In particular, you may want to refer to it for details on how various hyperbolic derivatives and integrals are derived, which we're going to take as given.

Python's `math` module contains implementations for cosh() and other hyperbolic functions (`https://docs.python.org/3/library/math.html`).

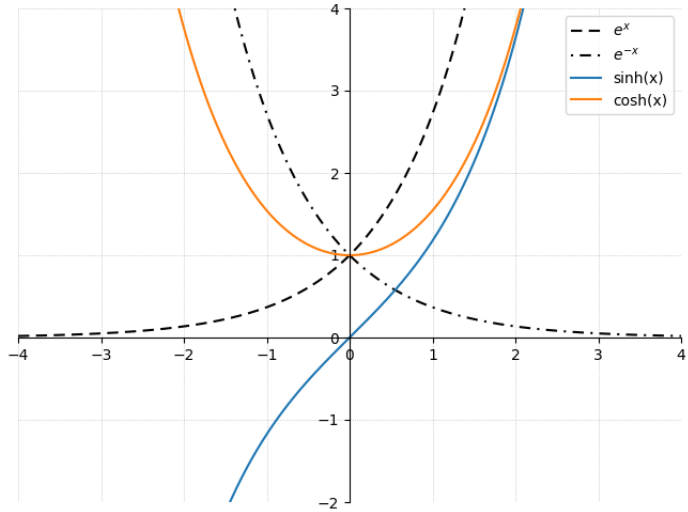


Figure 18.2. Plot of $\cosh(x)$, $\sinh(x)$, e^x, and e^{-x}

However, CPython doesn't implement these functions itself, instead calling the underlying platform's C math library, such as `libm` on Linux and macOS (have a look at CPython's `mathmodule.c` at `https://github.com/python/cpython/blob/main/Modules/mathmodule.c`). A relatively simple version, musl `libm` for embedded systems, can be found at `https://git.musl-libc.org/cgit/musl/tree/src/math`. The C implementation of cosh() is at `https://git.musl-libc.org/cgit/musl/tree/src/math/cosh.c`, and here's a pseudo-Pythonic version which seems very distant from $\cosh(x) = \frac{1}{2}(e^x + e^{-x})$:

```
def cosh(x):
  absX = abs(x)
  if absX < math.log(2):
    # for extremely small x, cosh(x) is approx 1
    if absX < 2**-26:
      return 1
    t = math.exp(absX) - 1
    return 1 + t * t / (2 * (1 + t))

  maxLog = 709.782712893384    # approx log() of MAX FLOAT
  if absX < maxLog:
    t = math.exp(absX)
    return 0.5 * (t + 1 / t)

  # for extremely large x
  return math.exp(absX)
```

Edge cases are dealt with by separate branches, and `1 + t*t/(2*(1+t))` is utilized for the general case rather than $\frac{1}{2}(e^x + e^{-x})$. The trouble with the standard definition is that it can suffer from catastrophic cancellation for small values of x due to the subtraction of nearly equal quantities in

$$\exp(x) \approx 1 + x + \frac{x^2}{2} \qquad \text{and} \qquad \exp(-x) \approx 1 - x + \frac{x^2}{2}.$$

One solution is to use $\exp(x) - 1$ and the identity

$$\cosh(x) = 1 + \frac{(\exp(x) - 1)^2}{2\,\exp(x)}.$$

Employing $t = \exp(x) - 1$, such that $\exp(x) = 1 + t$, produces the equation used in the code:

$$\cosh(x) = 1 + \frac{t^2}{2(1+t)}.$$

This is numerically stable because when x is small:

$$t = \exp(x) - 1 \approx x + \frac{x^2}{2} + \cdots,$$

since it doesn't involve the subtraction of nearly equal numbers.

The final return statement utilizes `math.exp(absX)` because the e^{-x} term is negligible for large x. Also, e^x can overflow, but that's handled by `math.exp()` returning `math.inf`.

18.1.2 Deriving the Catenary. We'll derive equations for the catenary with the help of Fig. 18.3. The tension T_0 at the lowest point A is directed horizontally. s is the length of the chain measured from A to any point P, and the tension at that point is T inclined by ψ to the horizontal. The weight per unit length of the chain is w.

The segment AP will be in equilibrium under the action of three forces: its weight ws, T_0, and the tension T at P. Resolving vertically and horizontally,

$$T \sin\psi = ws, \quad T\cos\psi = T_0.$$

For convenience, we introduce the constant a, such that $T_0 = wa$, which allows us to write

$$T \sin\psi = ws, \quad T\cos\psi = wa.$$

Dividing these equations

$$\tan\psi = \frac{s}{a} \qquad \text{or} \qquad s = a\tan\psi. \tag{18.1}$$

Equ. (18.1) is called the catenary's *intrinsic* equation, and a is its *parameter*.

To find the Cartesian version, start with

$$\tan\psi = \frac{dy}{dx}, \quad \text{and so} \quad \frac{dy}{dx} = \frac{s}{a}. \tag{18.2}$$

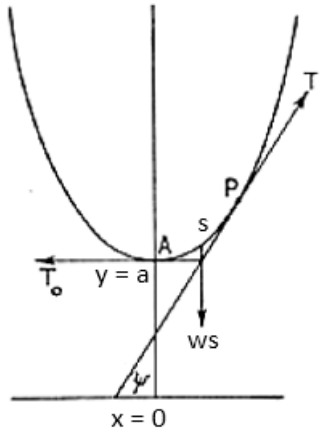


Figure 18.3. A Catenary Curve

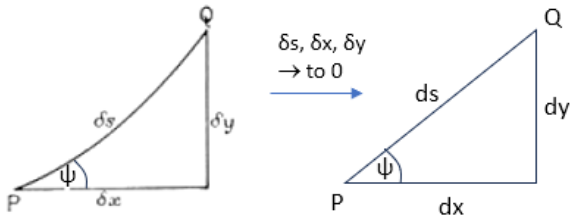


Figure 18.4. A Small Element δs of the Curve

Consider a small element δs of the curve joining two points P and Q, as in Fig. 18.4. Let the coordinates of P and Q be (x, y) and $(x+\delta x, y+\delta y)$ respectively. Then

$$(\delta s)^2 \simeq (\delta y)^2 + (\delta x)^2$$

so that

$$\left(\frac{\delta s}{\delta x}\right)^2 \simeq \left(\frac{\delta y}{\delta x}\right)^2 + 1$$

and

$$\left(\frac{\delta s}{\delta y}\right)^2 \simeq 1 + \left(\frac{\delta x}{\delta y}\right)^2$$

When δs, δx and $\delta y \to 0$, these become

$$\left(\frac{ds}{dx}\right)^2 = \left(\frac{dy}{dx}\right)^2 + 1 \tag{18.3}$$

and

$$\left(\frac{ds}{dy}\right)^2 = 1 + \left(\frac{dx}{dy}\right)^2 \tag{18.4}$$

Rearranging equ. (18.3) with the help of equ. (18.2):

$$\begin{aligned}\left(\frac{ds}{dx}\right)^2 &= 1 + \frac{s^2}{a^2} \\ \frac{ds}{dx} &= \frac{\sqrt{(a^2+s^2)}}{a} \\ dx &= \frac{a\,ds}{\sqrt{(a^2+s^2)}}\end{aligned}$$

Integrating:

$$x = a\sinh^{-1}\frac{s}{a} \qquad \text{or} \qquad s = a\sinh\frac{x}{a} \tag{18.5}$$

provided that $x = 0$ when $s = 0$.

Rearranging equ. (18.4) with the help of equ. (18.2):

$$\begin{aligned}\left(\frac{ds}{dy}\right)^2 &= 1 + \frac{a^2}{s^2} \\ \frac{ds}{dy} &= \frac{\sqrt{(s^2+a^2)}}{s} \\ dy &= \frac{s\,ds}{\sqrt{(s^2+a^2)}}\end{aligned}$$

Integrating:

$$y = \sqrt{(s^2+a^2)} \qquad \text{or} \qquad y^2 = s^2 + a^2 \tag{18.6}$$

provided that $y = a$ when $s = 0$.

Combining Equs. (18.5) and (18.6):

$$\begin{aligned}y^2 &= a^2\left(\sinh^2\frac{x}{a} + 1\right) \\ &= a^2\cosh^2\frac{x}{a}\end{aligned}$$

This gives us the Cartesian equation for a catenary:

$$\boxed{y = a\cosh\frac{x}{a}} \tag{18.7}$$

Since $T\sin\psi = ws$ and $T\cos\psi = wa$, then

$$T^2 = w^2(s^2 + a^2)$$

which, with the help of Equ. (18.6), is

$$\begin{aligned} T^2 &= w^2 y^2 \\ T &= wy \end{aligned}$$

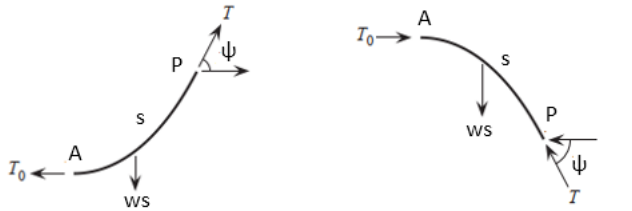
Thus the tension at any point in the catenary is proportional to the height of that point above the x-axis.

When a is large, Equ. (18.7) can be expanded as,

$$\begin{aligned} y = a\cosh\frac{x}{a} &= \frac{a}{2}(e^{x/a} + e^{-x/a}) \\ &= \frac{a}{2}\left\{1 + a + \frac{x^2}{2a^2} + \ldots + \left(1 - a + \frac{x^2}{2a^2} + \ldots\right)\right\} \\ &= a + \frac{x^2}{2a} + \ldots \end{aligned}$$

In this case, the curve is approximately a parabola.

18.1.3 Turning the Catenary Upside Down. If we turn Fig. 18.3 upside down to form an arch (producing Fig. 18.5), what was in tension is now in compression. This was first pointed out by Robert Hooke in 1675. The arch is only in compression, with no tension between the links, making an inverted catenary the strongest curve for building an arch.



a) Forces on a Hanging Chain b) Forces on a Section of an Arch

Figure 18.5. From Chain to Arch

18.1.4 The Parametric Form for the Catenary. We can differentiate Equ. (18.1) ($s = a\tan\psi$) to produce

$$\frac{ds}{d\psi} = a\sec^2\psi. \tag{18.8}$$

Also, from Fig. 18.4, we have

$$\frac{dx}{ds} = \cos\psi, \quad \frac{dy}{ds} = \sin\psi.$$

The dx/ds expression can be combined with Equ. (18.8)

$$\begin{aligned}\frac{dx}{d\psi} &= \frac{dx}{ds}\cdot\frac{ds}{d\psi} \\ &= \cos\psi\cdot a\sec^2\psi.\end{aligned}$$

Integrating both sides:

$$\begin{aligned}x &= \int a\sec\psi d\psi \\ &= a\log(\sec\psi+\tan\psi).\end{aligned}$$

Note that if x is measured from the catenary's lowest point (its *vertex*), then no constant term is required in the integral.

We can do a similar manipulation of the dy/ds expression and Equ. (18.8):

$$\begin{aligned}\frac{dy}{d\psi} &= \frac{dy}{ds}\cdot\frac{ds}{d\psi} \\ &= \sin\psi\cdot a\sec^2\psi.\end{aligned}$$

Integrating both sides:

$$\begin{aligned}y &= a\int\frac{\sin\psi}{\cos^2\psi}d\psi \\ &= a\sec\psi+\text{constant}.\end{aligned}$$

If y is measured from the vertex where ψ is zero, then the constant becomes $-a$. However, the usual convention is to place the origin at a distance a below the vertex so that $y=a$ when $\psi=0$, and then a constant isn't needed.

The collected parametric equations for the catenary are:

$$x = a\log(\sec\psi+\tan\psi) \qquad \text{and} \qquad y = a\sec\psi. \tag{18.9}$$

18.2 A Parabolic Chain

Fig. 18.6 depicts a chain with n equal weights (w) uniformly spaced out at horizontal intervals of h. This causes it to take on a parabolic shape rather a catenary, as we'll show below.

The weights are attached in arc length steps of $a_1, a_2, \ldots, a_n$, each inclined at $\theta_1, \theta_2, \ldots, \theta_n$ to the horizontal, and with tensions $T_1, T_2, \ldots, T_n$. The chain is tied to a point at (l,k) relative to the origin at the vertex O.

The equilibrium of the weights means that

$$T_1\cos\theta_1 = T_2\cos\theta_2 = T_3\cos\theta_3 = \ldots = T_n\cos\theta_n = H \text{ (say)}, \tag{18.10}$$

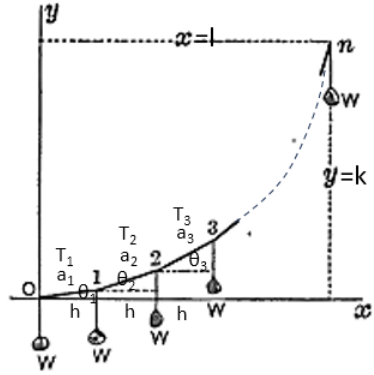


Figure 18.6. A Equally Weighted and Spaced Chain

and

$$\begin{aligned} T_2 \sin\theta_2 - T_1 \sin\theta_1 &= w, \\ T_3 \sin\theta_3 - T_2 \sin\theta_2 &= w, \\ &\vdots \\ T_n \sin\theta_{n+1} - T_{n-1} \sin\theta_n &= w. \end{aligned} \tag{18.11}$$

We have geometrical equations,

$$\begin{aligned} a_1 \cos\theta_1 + a_2 \cos\theta_2 + \ldots + a_n \cos\theta_n &= l, \\ a_1 \sin\theta_1 + a_2 \sin\theta_2 + \ldots + a_n \sin\theta_n &= k. \end{aligned}$$

Since the weights are equally spaced horizontally, we also have

$$a_1 \cos\theta_1 = a_2 \cos\theta_2 = \ldots = a_n \cos\theta_n = h \qquad \text{and} \qquad nh = l.$$

Dividing each of the equations in Equ. 18.11 by the relevant H equality in Equ. 18.10, we get

$$\begin{aligned} \tan\theta_2 - \tan\theta_1 &= \frac{w}{H}, \\ \tan\theta_3 - \tan\theta_2 &= \frac{w}{H}, \\ &\vdots \\ \tan\theta_n - \tan\theta_{n-1} &= \frac{w}{H}. \end{aligned}$$

These can be rearranged and simplified:

$$\begin{aligned}
\tan\theta_2 &= \tan\theta_1 + w/H, \\
\tan\theta_3 &= \tan\theta_2 + w/H \\
&= \tan\theta_1 + 2w/H, \\
\tan\theta_4 &= \tan\theta_3 + w/H \\
&= \tan\theta_1 + 3w/H, \\
&\vdots \\
\tan\theta_n &= \tan\theta_{n-1} + w/H \\
&= \tan\theta_1 + (n-1)w/H.
\end{aligned}$$

The nth coordinate (x, y) in the chain at step a_n and angle θ_n can be expressed as

$$\begin{aligned}
x &= nh \\
y &= h(\tan\theta_1 + \tan\theta_2 + \ldots + \tan\theta_n).
\end{aligned} \tag{18.12}$$

Each θ_r is replaced by its θ_1 expression, to get

$$y = h[n\tan\theta_1 + \frac{1}{2}n(n-1)w/H].$$

Use $x = nh$ (Equ. 18.12) to eliminate n, and express y in terms of x:

$$\frac{2Hh}{w}y = x^2 + \frac{2\tan\theta_1 H - w}{w}x,$$

which is an equation for a parabola.

18.3 Visually Comparing the Catenary and Parabola

sagCatParab.py and arclenCatParab.py plot a catenary and parabola together on the same graph, and include sliders to adjust parameters to see how the curves are affected.

18.3.1 Comparing Curves Using Sag. sagCatParab.py 'hangs' the curves from two endpoints at an equal height above the x-axis, and has them share the same vertex at the origin. This is only possible by allowing the two curves to have different lengths. A third dotted line shows the difference between the catenary's and parabola's y-values. The slider allows the sag (or dip, d) of the curves to be adjusted by changing the heights of the endpoints. Fig. 18.7 shows two screenshots of the plot.

It's apparent that as the sag d decreases that the differences between the two curves also decreases. It's not surprising that early investigators were fooled into thinking they were the same curve.

The subtraction in $y = a\cosh(x/a) - a$ lowers the catenary so its vertex is at the origin. The more difficult problem is to determine a given only the curve's

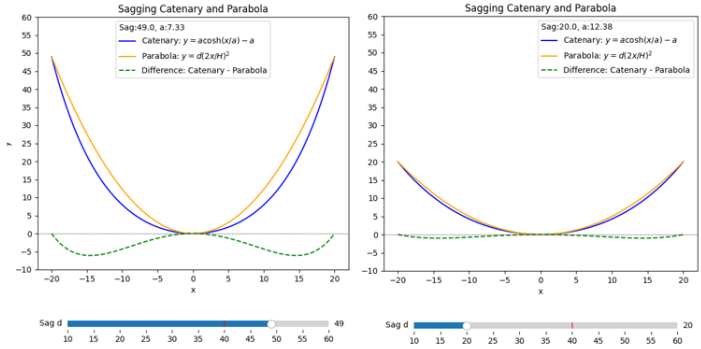


Figure 18.7. Adjusting the Sag of a Catenary and a Parabola

span H and sag d. This is handled by calling `catA()` in catSolver.py, which is explained in the next section.

18.3.2 The Parabola's Equation. If we know that three distinct points lie on the parabola then its equation can be determined by solving a linear system. Let the points be (x_1, y_1), (x_2, y_2), and (x_3, y_3), with x_1, x_2, x_3 pairwise distinct, and the quadratic that passes through those points be $y(x) = ax^2 + bx + c$. The coefficients satisfy

$$\begin{pmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}.$$

which can be solved for (a, b, c) by Gaussian Elimination (see section 6.8).

Fortunately, one of these points is the parabola's vertex, which simplifies matters. Suppose the vertex is (x_v, y_v), and the other two points are (x_1, y_1) and (x_2, y_2). The parabola's equation in terms of the vertex is

$$y = y_v + k(x - x_v)^2,$$

with only one unknown, k. Substitute one of the other points into the equation

$$y_1 = y_v + k(x_1 - x_v)^2,$$

and solve for k,

$$k = \frac{y_1 - y_v}{(x_1 - x_v)^2}.$$

The parabola becomes

$$y = y_v + \frac{y_1 - y_v}{(x_1 - x_v)^2}(x - x_v)^2.$$

In our case, $(x_v, y_v) = (0, 0)$, and so

$$y = \frac{y_1}{x_1^2} x^2.$$

Also, y_1 is equivalent to the curve's sag, d, and due to the symmetry of the two end points y_1 is half of the curve's span, H, away from the vertex. Using these values gives us:

$$y = \frac{4d}{H^2} x^2 \qquad \text{or} \qquad y = d\frac{2x}{H}^2$$

which is used by `sagCatParab.py` for plotting the parabola.

18.3.3 Comparing Curves Using their Length. `arclenCatParab.py` keeps the lengths of both curves the same, and suspends their ends between two fixed points at (-2,0) and (2,0). The slider allows the user to adjust the curves' length, as shown in Fig. 18.8. This time it's the two curve's vertices (lowest points) which vary over time. However, a vertex always remains midway between the two endpoints, at $x = 0$, so its only its y-coordinate that changes.

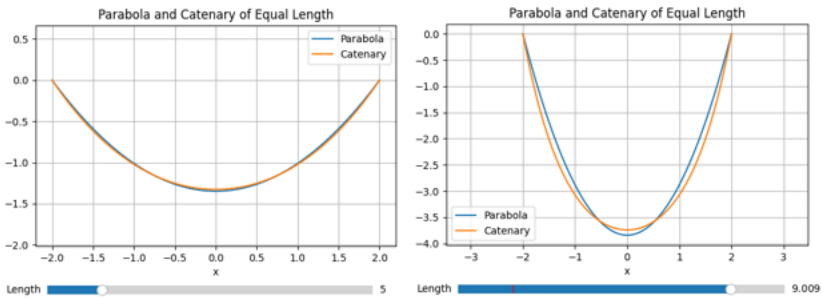


Figure 18.8. Adjusting the Length of a Catenary and a Parabola

The same relationship between the curves seen in `sagCatParab.py` also occurs in Fig. 18.8 – as the lengths decrease the curves become straighter, and the difference between them diminish.

The catenary's equation is derived from the two fixed symmetric endpoints and the arc length. Let the two endpoints be (x_1, y_1) and (x_2, y_1) so they're at the same height, and

$$h = \frac{x_1 + x_2}{2}$$

so the curve is symmetric about the line $x = h$, and the span as $H = x_2 - x_1$. Let $H_2 = H/2$, so the endpoints are at $x = h \pm H_2$.

A symmetric catenary can be written as

$$y = a\cosh\left(\frac{x-h}{a}\right) + k,$$

where k is a vertical shift that moves the catenary up or down so its endpoints remain at the same positions even as the curve's length changes.

The right-hand endpoint at $(h + H_2, y_1)$ satisfies

$$y_1 = a\cosh\left(\frac{H_2}{a}\right) + k.$$

so:

$$k = y_1 - a\cosh\left(\frac{H_2}{a}\right).$$

Substitute this into the catenary equation:

$$y(x) = y_1 + a\left[\cosh\left(\frac{x-h}{a}\right) - \cosh\left(\frac{H_2}{a}\right)\right].$$

This can be simplified a little since in our case $y_1 = 0$ and $h = 0$:

$$y(x) = a\left[\cosh\left(\frac{x}{a}\right) - \cosh\left(\frac{H_2}{a}\right)\right].$$

The length of the catenary between $x_1 = h - H_2$ and $x_2 = h + H_2$ is

$$\begin{aligned} L &= \int_{-H_2}^{H_2} \cosh\left(\frac{u}{a}\right) du \\ &= 2\int_0^{H_2} \cosh\left(\frac{u}{a}\right) du = 2a\sinh\left(\frac{H_2}{a}\right) \\ &= 2a\sinh\left(\frac{H}{2a}\right) \end{aligned}$$

This is a transcendental equation that needs to be solved for a when we have values for the curve's length L and span H. Once again, this is handled by calling `catA()` in catSolver.py, which is explained in the next section.

The code for generating the curve is located in `getCatenary()` in arclenCatParab.py:

```
def genCatenary(arcLen):
  a = getCatA(H=H, L=arcLen)              # H = horizontal width
  k = yCoord - a * math.cosh(H2 / a)      # H2 = H/2
  xs = [xLeft + i*dx for i in range(steps+1)]
  ys = [a*math.cosh(x/a) + k for x in xs]
  return xs, ys
```

18.3.4 The Parabola's Equation. Although the derivation of the parabola's equation starts out simply, it encounters a roadblock. Let the two endpoints be (x_1, y_1) and (x_2, y_1) and define h and H_2 as before so that the parabola's vertex is at (h, k) and the endpoints are at $x = h \pm H_2$. Because the parabola is symmetric about $x = h$, its equation can be written as

$$y = a(x-h)^2 + k.$$

Substituting in $(h + H_2, y_1)$ gives

$$y_1 = aH_2^2 + k,$$

so that

$$k = y_1 - aH_2^2.$$

Thus, the parabola may be written entirely in terms of a:

$$y = a(x - h)^2 + y_1 - aH_2^2.$$

Let $u = x - h$, which makes the derivative

$$y' = 2au.$$

The length L of the parabola between $x = h - H_2$ and $x = h + H_2$ is therefore

$$L = \int_{h-H_2}^{h+H_2} \sqrt{1 + (y')^2}\, dx = 2 \int_0^{H_2} \sqrt{1 + 4a^2u^2}\, du. \tag{18.13}$$

This has a closed form integral, but it's ugly, and it's more interesting to see how a can be obtained by summing the y' integral, as shown below.

Once a is found, the parabola is calculated using

$$y = a(x - h)^2 + y_1 - aH_2^2$$

This can be simplified to $y = a(x^2 - H_2^2)$ since $y_1 = 0$ and $h = 0$.

The code for generating the curve is located in `getParabola()` in arclenCatParab.py:

```
def genParabola(arcLen):
  a = solveParabolaA(arcLen)
  xs = [xLeft + i*dx for i in range(steps+1)]
  ys = [a*(x*x - H2*H2) + yCoord for x in xs]  # H2 = H/2
  return xs, ys
```

The actual hard work is carried out inside `solveParabolaA()`. It relies on `paraArcLen()` which uses a value for a to calculate the curve length by summing the y' integral from equ. 18.13 from $x = h - H_2$ to $x = h + H_2$.

```
def paraArcLen(a):
  length = 0.0
  x = xLeft    # x = h-H2
  for _ in range(steps):   # stops at x = h+H2
    slope = 2*a*x          # the derivative y'
    ds = math.sqrt(1 + slope*slope) * dx
    length += ds
    x += dx
  return length
```

The a passed to `paraArcLen()` is obtained by a binary search through a range of possible values until the length calculated by `paraArcLen()` is close enough to the known length of the curve (`arcLen`):

```
def solveParabolaA(arcLen):
  lowA = 0.0
  highA = 5.0
  for _ in range(40):
    midA = 0.5 * (lowA + highA)
    if paraArcLen(midA) > arcLen:
      highA = midA
    else:
      lowA = midA
  return 0.5 * (lowA + highA)
```

We coded things this way to illustrate another approach to finding a, but in the next section we'll use SciPy's `brentq()` solver for the equations associated with the catenary.

18.4 Determining the Catenary Parameter a

Determining the a parameter for a particular catenary can vary from being very simple to requiring numerical root-finding which can be quite tricky due to rounding and overflow errors. For these more complicated cases, we'll utilize the dimensions in Fig. 18.9. The ends of the catenary are at the same height, H is its span, d the dip (or sag), L the total length, and the vertex is a distance a above the x-axis.

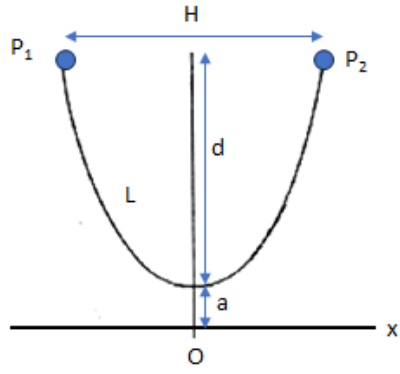


Figure 18.9. The Dimensions of a Catenary

The most common ways to determine a utilize:

(1) **Horizontal span (H) and sag (d).** This requires you to numerically solve

$$d = a\left(\cosh\left(\frac{H}{2a}\right) - 1\right)$$

for a. Unfortunately, it's quite difficult to catch all the edge cases that can produce incorrect answers or cause the solver to fail. One concern is highlighted by the Taylor expansion of cosh():

$$\cosh(u) \approx 1 + \frac{u^2}{2} + \frac{u^4}{24} + \cdots$$

the equation becomes

$$d = a\left(\cosh\left(\frac{H}{2a}\right) - 1\right) \approx a\left(\frac{(H/(2a))^2}{2}\right) = \frac{H^2}{8a}$$

$\cosh(u) - 1$ involves subtracting two nearly equal numbers, potentially causing catastrophic cancellation in the floating-point arithmetic. Also, if the sag (d) is small compared to the span (H), as when the catenary is nearly horizontal, then any slight error in d's measurement will produce huge variations in a.

Another concern is that certain numerical methods, such as Newton-Raphson and bisection, aren't well suited to solving transcendental equations. This drawback is avoided in `catA()` in catSolver.py by using Brent's method imported from the SciPy module (`https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.brentq.html`). However, that solver introduces its own concerns regarding setting the bounds around the root.

(2) **Horizontal span (H) and curve length (L).** Numerically solve:

$$L = 2a\sinh\left(\frac{H}{2a}\right)$$

for a.

(3) **Sag (d) and curve length (L).** This has an explicit solution,

$$a = \frac{L^2/4 - d^2}{2d}$$

18.5 Using a Catenary to Calculate Logarithms

'How to Find the Logarithm of Any Number Using Nothing but a Piece of String' by Viktor Blasjö [**Bla16**] describes how Leibniz utilized the catenary to calculate logarithms. His recipe is summarized in Fig. 18.10

One half of a catenary is extended horizontally so that a line can be drawn from its endpoint M to the chain's top midpoint at H. The line is bisected at B, and a normal to HM constructed through that point. Meanwhile, a perpendicular is dropped from H, and where it and the normal intersect is marked as the origin O. The distance from the origin to the lowest point of the catenary (its vertex at A) is considered to be a unit length. In that case, the catenary's equation will

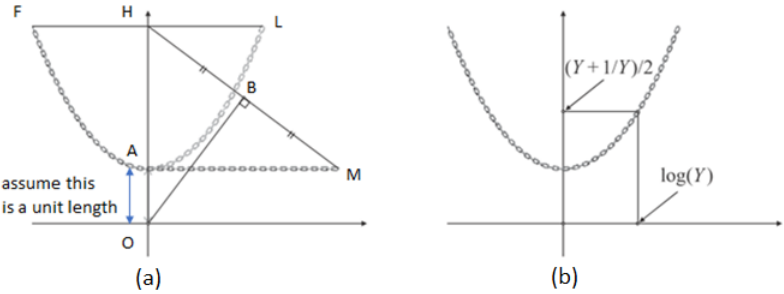


Figure 18.10. Leibniz's Recipe for Determining Logarithms from a Catenary

be $y = (e^x + e^{-x})/2$, and the logarithm tool is ready for use in Fig. 18.10(b). To determine $\log(Y)$, locate $(Y + 1/Y)/2$ on the y-axis and draw a line across to the curve, then down to the x-axis.

Why does this work? We have $y = a\cosh(\frac{x}{a})$, where $a = OA$ in Fig. 18.10(a). Since we've assigned this a value of 1, then $y = \cosh(x) = \frac{1}{2}(e^x + e^{-x})$. We also know that $y^2 = a^2 + s^2$ (see equ. (18.6)), or $y = \sqrt{1 + s^2}$. In Fig. 18.10(a), $y = OH$ and $s = AL$, and the horizontal segment AM has the same length. It follows by Pythagoras that $OM = \sqrt{1 + s^2}$, and $y = \sqrt{1 + s^2}$, which means that $OH = OM$. Thus, $\triangle OHM$ is isosceles, and the perpendicular bisector of its base HM will pass through O. So the construction does indeed give set up the coordinate system for $y = (e^x + e^{-x})/2$.

It may not be immediately obvious that the relationship in Fig. 18.10(b) is true. Let $x = \log(Y)$, and use it to simplify $y = \frac{1}{2}(e^x + e^{-x})$. It's also possible to test things in Python, as in the following examples:

```
>>> Y = 5
>>> math.acosh((Y + (1/Y))/2)
1.6094379124341003
>>> math.log(Y)
1.6094379124341003

>>> Y = 8
>>> math.acosh((Y + (1/Y))/2)
2.0794415416798357
>>> math.log(Y)
2.0794415416798357
```

18.6 Rolling a Parabola

The catenary and parabola are related in another way apart from looking very similar. If a parabola is rolled along a straight line, its focus traces out a catenary. rollParabola.py animates this relationship: the parabola keeps rolling right and left along the x-axis, reversing direction when the tip of an arm touches the axis. The position of its focus is recorded and its path displayed as a red curve. In addition, $y = \frac{1}{4a}\cosh(4ax)$ is drawn as a dashed gray line, revealing that the focus is following that curve (see Fig. 18.11).

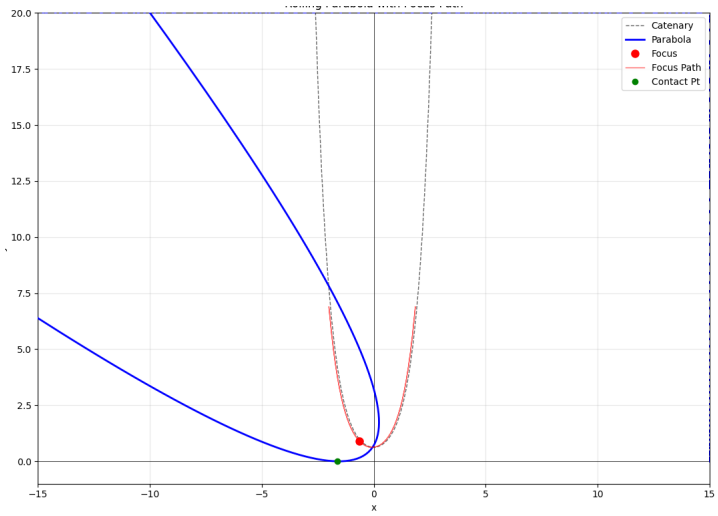


Figure 18.11. An Animated Rolling Parabola Tracing out a Catenary

rollParabola.py starts by generates all of the points for the parabola at the origin. Then the arc length of each of these points from the vertex is calculated: `calcLengths()` measures the distance between adjacent points and increments a total length value.

Matplotlib calls `animate()` each time an animation frame needs to be drawn. The parabola at the origin is rotated around the origin and its new coordinates are searched to find the lowest one. The curve is shifted up the y-axis so this 'contact' point is resting on the x-axis, and the point's corresponding arc length is used to move the curve the correct distance to the right or left.

The rotation and translation of the curve are also applied to the focus, which starts at $(0, 1/(4a))$. Its new position is recorded in a list which is used to draw its path.

The reason for the focal point being at $(0, 1/(4a))$ is due to the parabola having the equation $y = ax^2$. The link between a parabola and its focus is that any point

on the curve will be an equal distance from the focus and the length of a vertical line dropped to the directrix at $y = -f$ (see Fig. 18.12). This is discussed in more detail in section 16 on conics.

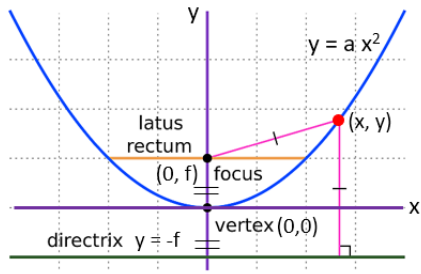


Figure 18.12. Linking f and a for a Parabola

Proving that the path of the focus is a catenary is a bit too involved to be investigated here, but we'll give some pointers to information on that topic at the end of this section. Instead, we'll follow Sánchez-Reyes [**SR12**] and demonstrate the equivalence of the path to the equation for cosh.

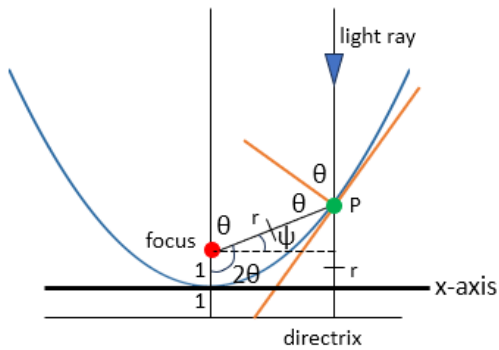


Figure 18.13. The Parabola Before Rolling

The parabola in Fig. 18.13 displays some useful facts – point P is an equal distance r from the focus and directrix, and a beam of light hitting the parabola vertically is reflected through the focus. We've labeled the angle of incidence and reflection as θ, and the focus is fixed to be 2 units above the directrix. $\psi =$

$2\theta - \pi/2$, and so

$$\begin{aligned} r &= 2 + r\sin(2\theta - \pi/2) \\ &= 2 + r(-\cos(2\theta)) \\ r(1 + \cos(2\theta)) &= 2 \end{aligned}$$

With the help of the double-angle identity, $1 + \cos(2\theta) = 2\cos^2\theta$:

$$\begin{aligned} r \cdot 2\cos^2\theta &= 2 \\ r &= \sec^2\theta \end{aligned}$$

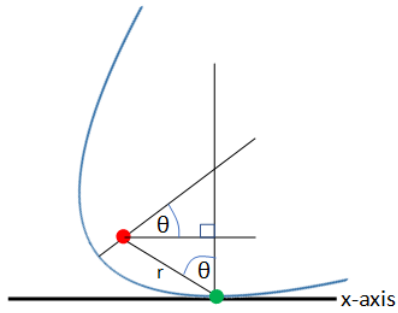


Figure 18.14. The Parabola After Rolling

Fig. 18.14 shows the parabola after it has rolled a short distance to the right. The coordinate axes now pass through the green point, which has become the center of rotation. This allows us to write

$$\frac{dy}{dx} = \tan\theta \qquad \text{and} \qquad y = r\cos\theta.$$

The second equation gives us the height of the focus as it rolls along the x-axis, and can be rewritten using the r equation and the derivative:

$$\begin{aligned} y &= r\cos\theta \\ &= \sec^2\theta \cdot \cos\theta \\ &= 1/\cos\theta \quad = \sec\theta \end{aligned}$$

Recall that $1 + \tan^2\theta = \sec^2\theta$, so

$$\begin{aligned} y &= \sec\theta \\ &= \sqrt{1 + \tan^2\theta} \\ &= \sqrt{1 + (dy/dx)^2} \end{aligned}$$

This equation specifies how the focus moves around the current center of rotation. Is there another equation with a similar relationship between y and dy/dx?

Turning to the hyperbolic functions, and using the identities:

$$\frac{d}{dx}\cosh x = \sinh x \qquad \text{and} \qquad \cosh^2 x - \sinh^2 x = 1$$

then

$$\begin{aligned} \cosh x &= \sqrt{1 + \sinh^2 x} \\ &= \sqrt{1 + \frac{d}{dx}(\cosh x)^2} \end{aligned}$$

If we set $y = \cosh(x)$, then the equation becomes:

$$y = \sqrt{1 + (dy/dx)^2}$$

In other words, the same general form. This is not conclusive proof that the focus' path follows a catenary, but it's certainly suggestive. A complete proof can be found in [**AM10**], where Agarwal and Marengo show that if the parabola is $y = ax^2$ then the path's equation is

$$y = \frac{1}{4a}\cosh 4ax$$

Another proof can be found at
`https://www.anirdesh.com/math/calculus/parabola-rolling-proof.php`, with a lot of additional analysis.

18.7 The Tractrix

In 1676, Frenchman Claude Perrault placed his pocket watch on a table, pulled the end of its chain along the table's edge, and asked: What's the shape of the curve traced out by the watch? The first known solution was given by Christiaan Huygens in a letter to a friend in 1693 where he also coined the name "tractrix", from the Latin *tratus* for something pulled along.

We've discussed the tractrix before, in the context of curves of pursuit in section 6.14.2, and return to it here because it can also be generated by taking the *involute* of a catenary.

Fig. 18.15 shows the geometrical relationships involved. The catenary's equation is $y = a\cosh(x/a)$, with its vertex C located at $(0, a)$. Imagine a string wrapped tightly over the right half of the catenary, ending at C, which is gradually pulled away from the curve so that the untethered string forms a tangent PQ. Q is a point on the tractrix, and P lies on the catenary. PQ has the same arc length as PC, labeled as s in Fig. 18.15, and the line segment of length a normal to PQ always touches the x-axis (currently at R). This can be confirmed by using the catenary equation $y^2 = s^2 + a^2$ (equ. (18.6) from section 18.1). Another equation applicable to Fig. 18.15 is $\tan\psi = s/a$ (equ. 18.1). Q corresponds to Perrault's watch, QR is the chain, and OR is the pulling direction.

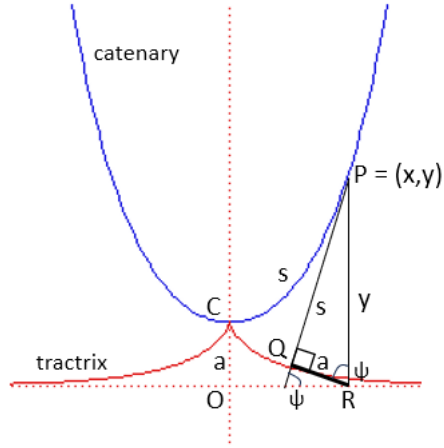


Figure 18.15. The Tractrix as an Involute of a Catenary

The left branch of the tractrix can be constructed in a similar way, but by starting with a string wrapped around the left half of the catenary, ending at C, and then pulling the string to the left.

Section 6.14.2 gives a Cartesian equation for the tractrix:

$$x = a \ln\left(\frac{a + \sqrt{a^2 - y^2}}{y}\right) - \sqrt{a^2 - y^2},$$

assuming that when $x = 0$, $y = a$. If we don't mind using hyperbolic functions, this can be rewritten. Let $y = a \text{ sech } u$ so that

$$\sqrt{a^2 - y^2} = a\sqrt{1 - \text{sech}^2 u} = a \tanh u,$$

and

$$\frac{a + \sqrt{a^2 - y^2}}{y} = \frac{a + a \tanh u}{a \text{ sech } u} = \cosh u + \sinh u = e^u.$$

Hence

$$\ln\left(\frac{a + \sqrt{a^2 - y^2}}{y}\right) = u,$$

which allows us to rewrite

$$\begin{aligned} x &= a \ln\left(\frac{a + \sqrt{a^2 - y^2}}{y}\right) - \sqrt{a^2 - y^2} \\ &= a\, u - a \tanh u \end{aligned}$$

paraTractrix.py plots $x = u - \tanh u$ and $y = \text{sech } u$ as shown in Fig. 18.16.

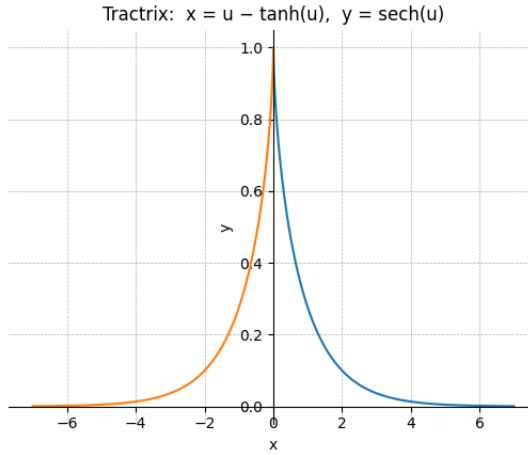


Figure 18.16. Plotting a Parametric Tractrix

18.7.1 Generating a Catenary with Tractrix Normals. Fig. 18.15 relates a point Q on a tractrix to a point P on the catenary by the normal QP to the tangent QR. This gives us a simple way to generate a catenary as an *envelope* of normals as shown in Fig. 18.17. It was drawn by `tractrixNormals.py`, with a little help from our Vector class, `Vec.py` (see section 14.4).

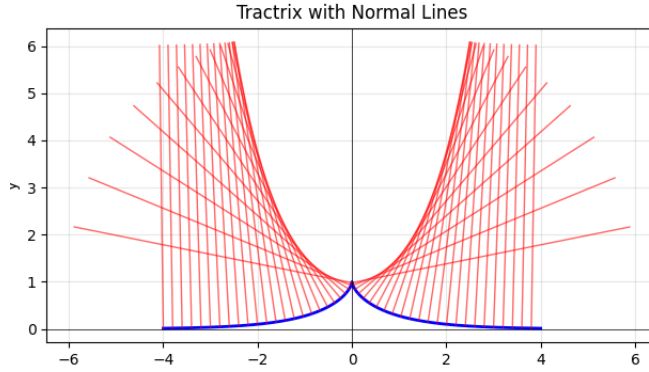


Figure 18.17. Drawing a Catenary with Tractrix Normals

To obtain a normal, we need a tangent, which can be produced by differentiating the parametric equations:

$$x(u) = u - \tanh u, \qquad y(u) = \operatorname{sech} u.$$

Differentiating with respect to u:

$$\begin{aligned}\frac{dx}{du} &= 1 - \frac{d}{du}(\tanh u) \\ &= 1 - \operatorname{sech}^2 u \\ &= \tanh^2 u.\end{aligned}$$

and

$$\begin{aligned}\frac{dy}{du} &= \frac{d}{du}(\operatorname{sech} u) \\ &= -\operatorname{sech} u \tanh u.\end{aligned}$$

These equations are utilized in `tractrix()` and `tractrixDerive()` in tractrixNormals.py to obtain a tractrix point and its tangent:

```
def tractrix(u, sign):
  x = sign * (u - math.tanh(u))
  y = 1 / math.cosh(u)
  return Vec(x, y, 0)

def tractrixDeriv(u, sign=+1):
  duDt = sign * math.tanh(u)**2
  duDt = -1/math.cosh(u) * math.tanh(u)
  return Vec(duDt, duDt, 0)
```

The tangent is converted to a normal in `getNormal()`:

```
def getNormal(tangent, sign):
  tangent = tangent.normalize()
  # For left branch (sign=-1), rotate clockwise;
  # for right branch, counterclockwise
  if sign == -1:
    normal = Vec(tangent.y, -tangent.x, 0)
  else:
    normal = Vec(-tangent.y, tangent.x, 0)
  normal *= NORMAL_LEN      # scale
  return normal
```

One slight quirk is that we have to check the tractrix branch to decide which way the normal vectors should point so they're aimed up the y-axis.

18.7.2 Generating a Tractrix with Circles. Yet another way to generate a tractrix, that doesn't involve a catenary, but is still interesting. Draw a large group of identical circles whose centers all lie along a straight line. Starting at the top of the first circle, draw a line to its center, stopping when your reach the edge of the next circle to the right. The process now repeats but linking the stopping point to its circle's center. The collection of points will *approximate* a tractrix, with the quality of the match improving with the number of circles along the line.

tractrixOrtho.py generates Fig. 18.18. The lower curve is generated in the same manner, but starting from the bottom of the leftmost circle.

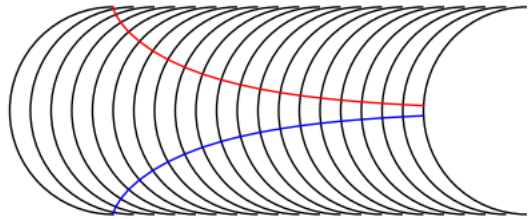


Figure 18.18. Drawing a Tractrix with Circles

The coding problem here is determining the intersection between a line segment and a circle. Inside `lineCircleIntersection()` in tractrixOrtho.py, the line (supplied as two points p_1 and p_2) is written in parametric form:

$$(x, y) = p_1 + t(p_2 - p_1)$$

To find the intersection, this is substituted into the equation for a circle:

$$(x - h)^2 + (y - k)^2 = r^2,$$

where (h, k) is the circle's origin, and r is its radius.

The components are rearranged as a quadratic in terms of t: $at^2 + bt + c = 0$, which is solved. In fact, the values for a, b, and c are hardwired into the code as:

```
a = dx*dx + dy*dy
b = 2 * (fx*dx + fy*dy)
c = fx*fx + fy*fy - r*r
```

where `dx` and `dy` are the coordinates of $p_2 - p_1$, and `fx` and `fy` are the coordinates of $p_1 - (h, k)$. Also, the quadratic's determinant, $D = b^2 - 4ac$, is examined to see if there are 0, 1, or 2 intersections. Each valid t is turned into an intersection point, and `collectPoints()` chooses the correct point if two intersections were found, using the one nearest to the other point forming the line segment.

18.8 Riding a Bike with Square Wheels

Fig. 18.19 shows a bike with square wheels at Macalester College, constructed by mathematician Stan Wagon and bicycle mechanic Loren Kellen in 1997 (`https://www.macalester.edu/mscs/multimedia/squarewheeledbike/squarewheelbike/`).



Figure 18.19. A Bicycle with Square Wheels

The most relevant question here is what road surface is required to make riding this bike a reasonable possibility. Of course, the answer is a catenary, as we'll show below.

Gerson Robison is credited with this discovery in 1960 [**Rob60**], although it became more widely known through the efforts of Stan Wagon and Leon Hall [**Wag99, HW92**]. Often forgotten is the very early work on Hengist Pod, inventor of the square wheel in around 54 BC as documented in *Carry on Cleo* [**Tho64**], and shown in Fig. 18.20.



Figure 18.20. Hengist Pod Demonstrating his Invention

We obviously want the bike's wheels to roll smoothly, which means that the center of each wheel should move horizontally. There should be no slipping, which implies that a wheel will form a tangent to the road's surface at the point of contact, and its center is directly above that contact point.

Fig. 18.21 shows a wheel of side length $2a$ at the top of a crest in the road. As the wheel rolls, the center will always remain a height h above the x-axis.

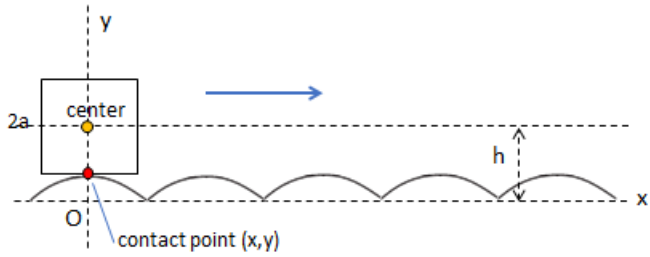


Figure 18.21. The Initial Square Wheel

This height requirement means that even when the wheel has rolled into a valley that its center will still be h above the axis (see Fig. 18.22). Since the wheel has a side length of $2a$, then $h = a\sqrt{2}$.

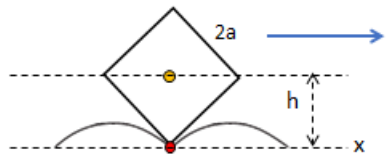


Figure 18.22. The Square Wheel in a Cusp

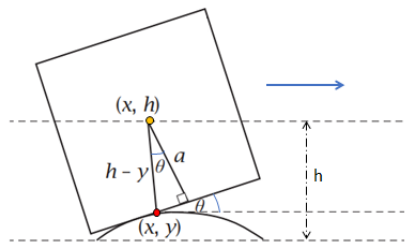


Figure 18.23. The Wheel in a General Position

We're now ready to consider the general case shown in Fig. 18.23. The wheel is inclined at an angle θ to the horizontal, with the point of contact at some (x, y) coordinate, so that:

$$a = (h - y)\cos\theta = \frac{h - y}{\sec\theta} = \frac{h - y}{\sqrt{1 + \tan^2\theta}} = \frac{h - y}{\sqrt{1 + (dy/dx)^2}}.$$

Therefore,

$$\frac{dy}{dx} = \pm\sqrt{((h-y)/a)^2 - 1}. \tag{18.14}$$

We'll use the positive part of this equation, which implies that the wheel is rolling up the incline (as in Fig. 18.23),

$$\begin{aligned} \frac{dy}{dx} &= \sqrt{\left(\frac{(h-y)^2-a^2}{a^2}\right)} \\ \int \frac{1}{\sqrt{(h-y)^2-a^2}}\, dy &= \frac{1}{a}\int dx \end{aligned}$$

After integrating:

$$\begin{aligned} -\cosh^{-1}\frac{y-h}{a} &= \frac{x}{a} + c \\ y &= -a\cosh\left(\frac{x}{a}+c\right) + h. \end{aligned}$$

Fig. 18.21 indicates that when the wheel first starts rolling that its contact point $(x, y) = (0, h-a)$. Substituting this into the equation makes $\cosh(c) = 1$ and so $c = 0$. The equation for the road's surface becomes,

$$y = h - a\cosh\left(\frac{x}{a}\right) \tag{18.15}$$

Since $h = a\sqrt{2}$, then

$$y = a\sqrt{2} - a\cosh\left(\frac{x}{a}\right), \tag{18.16}$$

an inverted catenary.

At the joint between two curves in the road, as in Fig. 18.22, the contact point is touching the x-axis so $y = 0$ and Equ. 18.14 can be written as

$$\frac{dy}{dx} = \sqrt{((a\sqrt{2}-0)/a)^2 - 1} = 1,$$

This implies that $\tan\theta = 1$, so the angle between the wheel and x-axis is $\pi/4$ in Fig. 18.22. Of course, each corner takes up $\pi/2$, so the wheel fits snugly into the gap between the crests.

To determine the length of a curved section of the road, let's measure from $x = 0$ (the crest in Fig. 18.21) to the next valley at $(x_1, 0)$. The total length, s, will be double this distance:

$$\begin{aligned} s &= 2\int_0^{x_1} \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx \\ &= 2\int_0^{x_1} \sqrt{1 + \left(\frac{h-y}{a}\right)^2 - 1}\, dx \\ &= 2\int_0^{x_1} \frac{h-y}{a}\, dx \end{aligned}$$

Use Equ. 18.15 to equate $(h-y)/a = \cosh(\frac{x}{a})$:

$$\begin{aligned} s &= 2\int_0^{x_1} \cosh\frac{x}{a}\,dx \\ &= 2\left[a\sinh\frac{x}{a}\right]_0^{x_1} \\ &= 2a\sinh\frac{x_1}{a}, \end{aligned}$$

Plug the coordinate of the second cusp, $(x_1, 0)$, into Equ. 18.16:

$$\begin{aligned} a\sqrt{2} - a\cosh\left(\frac{x_1}{a}\right) &= 0 \\ \cosh\left(\frac{x_1}{a}\right) &= \sqrt{2} \end{aligned}$$

Using the identity $cosh^2u - sinh^2u = 1$:

$$\begin{aligned} cosh^2(x_1/a) - sinh^2(x_1/a) &= 1 \\ 2 - sinh^2(x_1/a) &= 1 \\ \sinh(x_1/a) &= 1. \end{aligned}$$

Substitute this into the s equation to obtain:

$$s = 2a\sinh\frac{x_1}{a} = 2a$$

So, the length of each curved section is the same as the length of the side of the square, which means that the wheel can rotate between cusps without any slippage.

Hall and Wagon [**HW92**] go into considerably more detail than we have, considering a variety of road surfaces (what Robison terms 'rockers') and wheel shapes ('rollers'). For example, the wheel can be any regular polygon *except* an equilateral triangle. The corner of a triangle hits the road surface before it enters a valley, and scrapes against it again on the way out. This is quite surprising since the angle between the two sides of the adjacent catenaries is 90°, more than enough room for the triangle's corners. As the number of sides in the polygon gets larger, it begins to look more and more like a circle and the line of catenaries will get flatter, looking increasingly like a flat road.

Although a ride over inverted catenaries feels smooth, it's not *quite* as smooth as a regular bicycle moving over a flat surface. For an ordinary bike, the horizontal distance traveled, x, is linearly proportional to the angle θ pedaled – pedal twice as much and you travel twice as far. This is almost true of the square wheel, but not exactly, as shown by Fig. 18.24.

If the distance traveled is $y = f(x)$, then $d\theta/dx = -1/f(x)$. We know $y \approx -\cosh(x)$, and so after integration $\theta(x) \approx 2\arctan(-e^{-x})$. An examination of the Taylor series for arctan() highlights its small discrepancy from linearity.

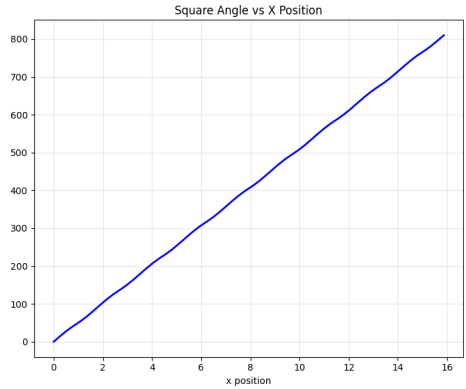


Figure 18.24. θ versus x for the Rolling Wheel

Fig. 18.24 is generated by rollingSquare.py, which also animates the path of a square wheel over a series of inverted catenaries in Fig. 18.25. `rollingSquare.py` was inspired by a much more feature-rich program in Wagon's *Mathematica in Action* [**Wag10**].

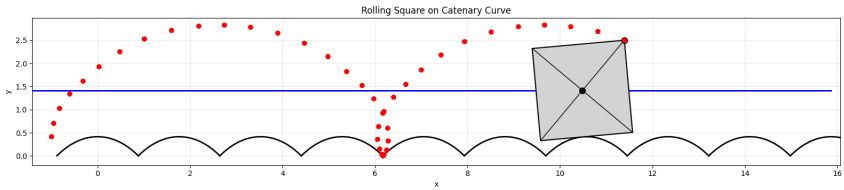


Figure 18.25. Animating the Journey of a Square Wheel

The series of red dots record the journey of one of the corners of the square, while the center of the square never deviates from the blue horizontal line. Note, that the corner occasionally goes backwards, reminiscent of the classic puzzle about the locus of the point at the bottom of the flanged wheel of a train [**Dar04**]. This is a *prolate* cycloid, the path traced out by a fixed point at a radius $b > a$, where a is the radius of a rolling circle; in our case, $b = h = a\sqrt{2}$.

An important point to remember when viewing `rollingSquare.py`'s animation is that no kind of collision detection is being carried out. A square shape is being rotated around its origin as it moves across the window by following the blue line. Its smooth traversal is *not* the result of it rolling over the catenaries. At best, the animation is an indication that the edges of the rotating square always seem to be just above the catenaries that it passes over.

What are the parameters required to produce this effect? There are three: `ir`, `cr`, and `hArch`, as shown in Fig. 18.26.

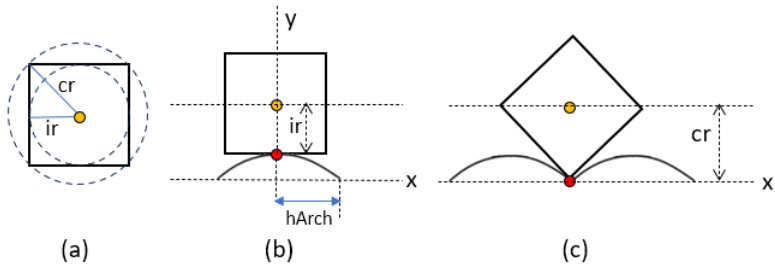


Figure 18.26. The Parameters used by `rollingSquare.py`

The four corners of the square are defined in terms of a circumradius (`cr`) and inradius (`ir`) (see Fig. 18.26(a)), which are set to 1 and $\sqrt{2}$ respectively. These variables are also used to determine `hArch`, the half-width of each catenary arch drawn along the floor (see Fig. 18.26(b)).

Each 'bump' on the floor is an inverted catenary, with an equation based on $y = -a\cosh(x/a)$ like the one in Fig. 18.27.

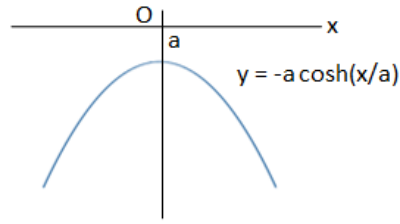


Figure 18.27. An Inverted Catenary

If the x-axis was the horizontal line through the center of the square, then a would equal `ir` as indicated best by Fig. 18.27(b). However, since the x-axis is `cr` units down from that line (as seen in Fig. 18.27(c)) then a corresponding shift must be applied to the catenary, producing:

$$y = \mathrm{cr} - \mathrm{ir} \cdot \cosh(x/\mathrm{ir})$$

The square is shown at the top of a crest in Fig. 18.26(b), when $(x, y) = (0, 0)$. y will next be 0 when $x =$ `hArch`. Plug this y value into the catenary equation, and

see what x is returned:

$$\begin{aligned} \text{cr} - \text{ir} \cdot \cosh(x/\text{ir}) &= 0 \\ \sqrt{2} - \cosh(x) &= 0 \\ \cosh(x) &= \sqrt{2} \end{aligned}$$

Using the identity, $cosh^2 x - sinh^2 x = 1$:

$$\begin{aligned} 2 - \sinh^2(x) &= 1 \\ x &= asinh(1) \end{aligned}$$

and so the code sets `hArch = math.asinh(1)`.

The first inverted catenary is drawn at $x = -\text{hArch}$, and subsequent catenaries are `2*hArch` apart. `catenaryPts()` in rollingSquare.py generates the coordinates for the ith catenary along the x-axis:

```
def catenaryPts(i, cr, ir, hArch, numPts=100):
  xs = []; ys = []
  xMin = -hArch + 2*hArch*i
  xMax = hArch + 2*hArch*i
  for j in range(numPts):
    x = xMin + (xMax - xMin)*j/(numPts - 1)
    y = cr - ir*math.cosh((x - 2*hArch*i)/ir)
        # catenary equation, offset by 2*hArch*i
    xs.append(x)
    ys.append(y)
  return xs, ys
```

18.9 The Catenoid

A catenoid is the surface of revolution produced when a catenary like the one in Fig. 18.3 is rotated about its x-axis. The catenoid, first described by Leonhard Euler in 1740, is the oldest known *minimal surface* – a shape of least area when bounded by a given closed space. This can be demonstrated nicely by a soap film stretched between two rings (see Fig. 18.28).

The catenoid can be defined by the parametric equations:

$$\begin{aligned} x &= a\cosh(\tfrac{v}{a})\cos u \\ y &= a\cosh(\tfrac{v}{a})\sin u \\ z &= v \end{aligned}$$

where $u \in [-\pi, \pi)$, $v \in \mathbb{R}$ and a is a non-zero real constant.

catenoid.py uses these equations to plot the 3D surface shown in Fig. 18.29. The somewhat mysterious a constant can be adjusted using a slider, revealing that it controls the size of the catenoid's 'waist' – reducing a makes it thinner, while a larger a makes the shape more cylindrical.

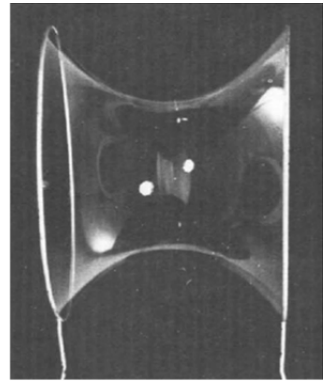


Figure 18.28. A Soap-film Catenoid

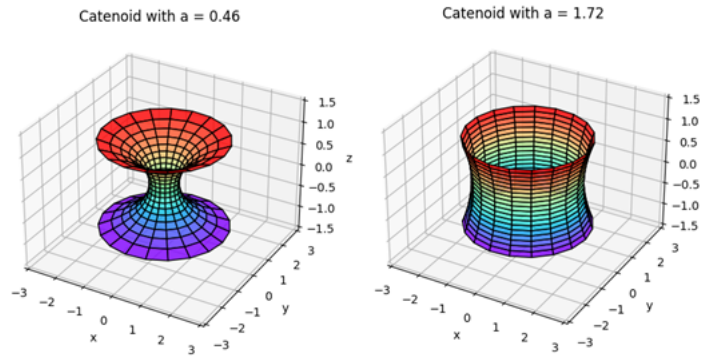


Figure 18.29. An Adjustable Catenoid

18.9.1 The Smallest Waist. Fig. 18.30 is a version of the soap-film catenoid in Fig. 18.28, but with various dimensions labeled. We've also reassigned the main axis to be the z-axis.

As the rings are pulled apart (i.e. H increases), the 'neck' gets longer and the 'waist' gets thinner (i.e. a decreases). However, at a critical distance, which depends on the ring's radius R, the soap film breaks, leaving planar films on each ring.

The catenoid's radius at any point along the z-axis, can be expressed using the equation for a catenary:

$$r(z) = a\cosh\left(\frac{z}{a}\right),$$

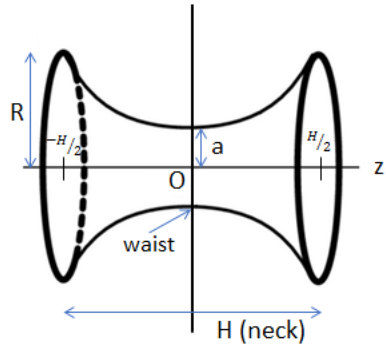


Figure 18.30. The Dimensions of a Catenoid

where a is the minimum neck radius at $z = 0$.

At either ring, say the one at $z = H/2$, the radius must equal the ring radius:

$$r(H/2) = a\cosh(\frac{H}{2a}) = R.$$

This equation allows us to determine a value for a when the values for R and H are known. Unfortunately, the equation is transcendental, so numerically methods (e.g. Brent's method) have to be employed to solve it.

It's also useful to plot the curve, which is simpler if we rearrange it. Multiply through by $2/H$:

$$\frac{2a}{H}\cosh(\frac{H}{2a}) = \frac{2R}{H}$$

and let $t = H/2a$, so that:

$$Y = (2R/H) = \cosh(t)/t$$

coshPlot.py plots $Y = cosh(t)/t$ for positive t values (see Fig. 18.31), and finds the minimum Y using SciPy's `brentq()` root finder (`https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.brentq.html`).

The graph in Fig. 18.31 tells us that there are two soap films that satisfy $Y = 2R/H > 1.5089...$, and nothing for $2R/H < 1.5089$. This is easily demonstrated by starting with a stable soap film, and gradually increasing the H separation between the rings. The rings don't change in size, so R is constant, which means that has H gets bigger that $2R/H$ decreases. At the minimum point, the film collapses into two.

When $2R/H > 1.5089...$, two catenoid shapes are possible – the common, 'stable' version with a thin neck that corresponds to the curve to the left of the minimum point. There's also a small chance of obtaining a bulging catenoid with

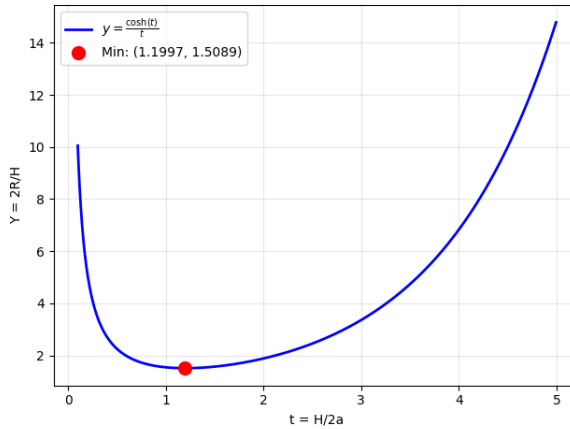


Figure 18.31. Plot of $Y = \cosh(t)/t$

a thick neck, which is unstable and likely to switch to the slim form at any time. This corresponds to the curve to the right of the minimum.

In soap-film demonstrations, R is usually fixed, so its useful to rephrase H and a in terms of R. The minimum point (t^*, Y^*) is

$$t^* = \frac{H}{2a} \approx 1.997, \quad \text{and} \quad Y^* = \frac{2R}{H} \approx 1.5089$$

Let's define H_c, the *critical* neck length, in terms of R:

$$\begin{aligned} 2R/H &= \cosh(t)/t \\ H_c &= (2Rt^*)/\cosh(t^*) \approx (2 \cdot 1.1997)R/\cosh(1.1997) \approx 1.325R \end{aligned}$$

So the critical neck length is about $\frac{4}{3}$ of the ring's radius. Any larger, and the catenoid will become unstable.

In terms of R, the critical waist *diameter*, $2a_c$, is

$$\begin{aligned} H/2a &= t^* \\ 2a_c &= H_c/t^* \approx 1.3255R/1.1997 \approx 1.105R \end{aligned}$$

The critical waist diameter is about the same as the ring radius. Any smaller and the catenoid will become unstable.

18.9.2 A Minimal Surface. The minimal surface property for a catenoid means that the concave surface of our soap film has a smaller area any other shape stretched between the two rings, such as a cylindrical film.

In 1744, Euler obtained a differential equation for determining the minimum area contained by a fixed boundary, although his analysis was later simplified by Lagrange. The Euler-Lagrange equation assumes that the problem surfaces are

continuous and have continuous derivatives. Sadly, most problems don't satisfy these criteria, although our soap film can be solved, but the details are a little advanced requiring concepts from the Calculus of Variations. One fairly approachable derivation can be found in Appendix III of Isenberg's *The Science of Soap Films and Soap Bubbles* [**Ise92**].

18.9.3 Comparing Surface Areas. Although we won't prove that the catenoid possesses the smallest surface area between two rings, we can calculate its area, and compare it to the surface area of a cylinder.

For a radius curve $r(z)$ rotated around the z-axis like the one in Fig. 18.28, the surface area between $z = -H/2$ and $z = H/2$ is:

$$S = 2\pi \int_{-H/2}^{H/2} r(z)\sqrt{1 + \left(\frac{dr}{dz}\right)^2}\, dz$$

$r(z) = a\cosh\left(\frac{z}{a}\right)$ is differentiated:

$$\frac{dr}{dz} = a \cdot \frac{1}{a}\sinh\left(\frac{z}{a}\right) = \sinh\left(\frac{z}{a}\right)$$

which means that:

$$\sqrt{1 + \left(\frac{dr}{dz}\right)^2} = \sqrt{1 + \sinh^2\left(\frac{z}{a}\right)}$$

Using $\cosh^2(x) - \sinh^2(x) = 1$:

$$\sqrt{1 + \sinh^2\left(\frac{z}{a}\right)} = \cosh\left(\frac{z}{a}\right)$$

Plug this and $r(z)$ into the surface area equation:

$$\begin{aligned} S &= 2\pi \textstyle\int_{-H/2}^{H/2} a\cosh\left(\frac{z}{a}\right) \cdot \cosh\left(\frac{z}{a}\right) dz \\ &= 2\pi a \textstyle\int_{-H/2}^{H/2} \cosh^2\left(\frac{z}{a}\right) dz \end{aligned}$$

Using the identity: $\cosh^2(x) = (1 + \cosh(2x))/2$:

$$\begin{aligned} S &= 2\pi a \textstyle\int_{-H/2}^{H/2} \frac{1+\cosh\left(\frac{2z}{a}\right)}{2}\, dz \\ &= \pi a \textstyle\int_{-H/2}^{H/2} \left[1 + \cosh\left(\frac{2z}{a}\right)\right] dz \\ &= \pi a \left[z + \frac{a}{2}\sinh\left(\frac{2z}{a}\right)\right]_{-H/2}^{H/2} \end{aligned}$$

Evaluate this expression between $-H/2$ and $H/2$:

$$S = \pi a\left[\left(\frac{H}{2} + \frac{a}{2}\sinh\left(\frac{H}{a}\right)\right) - \left(-\frac{H}{2} + \frac{a}{2}\sinh\left(-\frac{H}{a}\right)\right)\right]$$

Since $\sinh(-x) = -\sinh(x)$:

$$\begin{aligned} S &= \pi a\left[\frac{H}{2} + \frac{a}{2}\sinh\left(\frac{H}{a}\right) + \frac{H}{2} + \frac{a}{2}\sinh\left(\frac{H}{a}\right)\right] \\ &= \pi a\left[H + a\sinh\left(\frac{H}{a}\right)\right] \end{aligned}$$

areas.py utilizes this expression in `catenoidArea()` after first using SciPy's Brent's method to find a.

```
def catenoidArea(R, H):
  # Solve for catenoid parameter a using brentq()
  z = H/2
  aMin = R * math.exp(-z/R)
  a = brentq(catenoidEq, aMin, R, args=(R, z))
  return math.pi * a * (H + a * math.sinh(H / a))


def catenoidEq(a, R, z):
  return a * math.cosh(z/a) - R
```

These functions are used to generate a sequence of catenoid surface areas, with R fixed at 1, and H varying between 0.1 and 1.32548 (recall that the critical neck length $H_c < 1.325R$, or the soap film will break).

The catenoid's surface area is plotted against a cylinder's area (calculated using $2\pi RH$) in Fig. 18.32. For small H's, the two shapes have a similar surface area, but the catenoid's area becomes smaller as the distance between the rings increases.

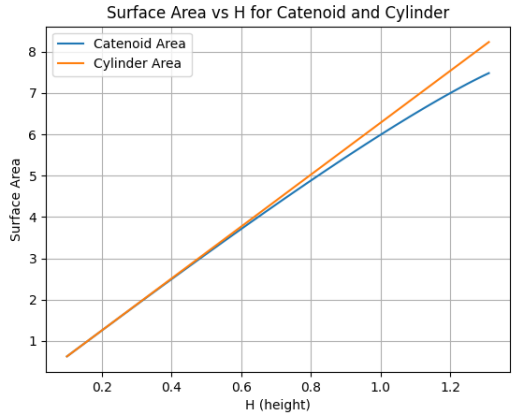


Figure 18.32. Surface Area vs. H for a Catenoid and Cylinder

Exercises

(1) A uniform chain of length $2l$ and weight w per unit length is suspended between two points at the same level and has a maximum sag of d. Prove that the tension at the lowest point is

$$w\frac{l^2-d^2}{2d}.$$

If $l = 50$ m and $d = 20$m, find the distance between the points of suspension.

(2) A uniform chain of length $2l$ has its ends fixed at two points at the same level. The sag at the middle is d. Prove that the span H is

$$\frac{l^2-d^2}{d}\log\frac{l+d}{l-d}.$$

(3) A chain is suspended from two points at the same level and $2l$ apart. The lowest point of the chain is d below the level of the supports. In each half of the chain, the load per unit horizontal distance varies uniformly with the horizontal distance from zero at the lowest point to w at the ends.

Taking the origin at the lowest point, show that the equation of the curve for the chain is

$$y = \frac{dx^3}{l^3}.$$

(4) Find the equation of the tractrix by evaluating the integral

$$-\int \frac{\sqrt{a^2-x^2}}{x}\,dx.$$

There are at least three substitutions which will work: $t^2 = a^2 - x^2$, $x = a\sin(\theta)$, and $x = a\cos(\theta)$.

Do the problem all three ways. Which of these substitutions do you think provides the easiest solution?

(5) In section 18.4, we discussed three equations that are often used to find the a value for a catenary:

(a) $d = a\left(\cosh\left(\frac{H}{2a}\right) - 1\right)$

(b) $L = 2a\sinh\left(\frac{H}{2a}\right)$

(c) $a = \frac{L^2/4-d^2}{2d}$

Derive these equations using the dimensions defined in Fig. 18.9, the catenary equation $y = a\cosh(x/a)$, and other identities from section 18.1.2.

(6) In section 18.6, it is stated without proof that a parabola $y = ax^2$ will have its focus at $(0, \frac{1}{4a})$. Show this to be true by utilizing Fig. 18.12.

(7) In section 18.8, it is stated that the corner of a square wheel fits snugly into the valley between two inverted catenaries. This implies that the angle each catenary makes with the x-axis is $\pi/4$. Show this to be true.

(8) Investigate the statement made near the end of section 18.8 that any polygonal wheel except a triangular one will roll without difficulty over a catenary.

(9) The description of the output of `rollingSquare.py` in section 18.8 mentions that if the distance traveled is $y = f(x)$, then $d\theta/dx = -1/f(x)$. Why is this true?

Answers

(1) For the catenary in Fig. 18.33:

$$y^2 = c^2 + s^2.$$

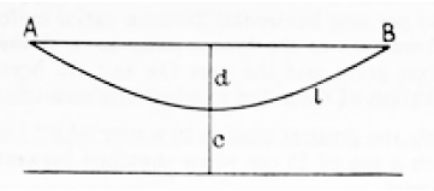


Figure 18.33. The Catenary in Q.1

At point B:

$$y = c + d, \qquad s = l,$$

so

$$(c + d)^2 = c^2 + l^2.$$

Thus,

$$2cd = l^2 - d^2, \qquad c = \frac{l^2 - d^2}{2d}.$$

The tension at the lowest point is therefore

$$wc = w\frac{l^2 - d^2}{2d}.$$

If $l = 50$ and $d = 20$:

$$c = \frac{2500 - 400}{40} = \frac{105}{2}.$$

Now, since

$$s = c\sinh\left(\frac{x}{c}\right),$$

we have if $AB = 2x$,

$$x = \frac{105}{2}\sinh^{-1}\left(\frac{20}{21}\right) = \frac{105}{2}\log\left[\frac{20}{21} + \sqrt{1 + \left(\frac{20}{21}\right)^2}\right].$$

Hence,

$$AB = 105\log\frac{49}{21} = 105 \times 2.303\log_{10}\frac{49}{21} \approx 89 \text{ m}.$$

(5) Deriving Equations:

(a) We'll show how $d = a(\cosh(H/(2a)) - 1)$ applies to a catenary by deriving $H = 2a\,\text{arcosh}((a + d)/a)$, and then rearranging its terms.
Consider the catenary in Fig. 18.9. Its lowest point is $(0, a)$ at $x = 0$, and its attachment points at P_1 and P_2 are at $x = \pm H/2$, height $= a + d$.
At the attachment point $x = H/2$, the height is $y = a + d$:

$$a + d = a\cosh(H/2a)$$

Divide both sides by a:

$$\frac{a + d}{a} = \cosh(H/2a))$$

Take the inverse hyperbolic cosine of both sides:

$$\text{arcosh}\left(\frac{a + d}{a}\right) = \frac{H}{2a}$$

Multiply both sides by $2a$:

$$H = 2a\,\text{arcosh}\left(\frac{a + d}{a}\right)$$

Rearrange the equation so the left side is in terms of d

$$d = a(\cosh(H/(2a)) - 1)$$

(b) $L = 2a\sinh\left(\frac{H}{2a}\right)$. Once again referring to the catenary in Fig. 18.9. The left attachment point P_1 is at $x = -H/2$ and P_2 is $x = H/2$. The arc length of a catenary from its vertex to a point P is $s = a\sinh\frac{x}{a}$, so the length between two points, with $x = x_1$ and $x = x_2$ is:

$$a\left[\sinh(x_2/a) - \sinh(x_1/a)\right]$$

But, we want the total arc length L from $x = -H/2$ to $x = H/2$:

$$L = a\left[\sinh(H/(2a)) - \sinh(-H/(2a))\right]$$

Use the property: $\sinh(-u) = -\sinh(u)$

$$\begin{aligned} L &= a\left[\sinh(H/(2a)) - (-\sinh(H/(2a)))\right] \\ &= a\left[\sinh(H/(2a)) + \sinh(H/(2a))\right] \\ &= 2a\sinh(H/(2a)) \end{aligned}$$

For small sags (large a relative to H), $\sinh(H/(2a)) \approx H/(2a)$, giving $L \approx H$ (nearly a straight line). For large sags (small a), sinh() grows exponentially, reflecting the much longer cable needed

(c) $a = (L^2/4 - d^2)/2d$. Back to Fig. 18.9 one more time. The attachment points, P_1 and P_2 are located at $(-H/2, a+d)$ and $(H/2, a+d)$ respectively. Plugging the latter into the catenary equation:

$$a + d = a\cosh(H/2a), \qquad \text{or} \qquad \cosh(H/2a) = 1 + \frac{d}{a}$$

The total arc length from $x = -H/2$ to $x = H/2$ is:

$$L = 2a\sinh(H/2a), \qquad \text{or} \qquad \sinh(H/2a) = L/(2a)$$

Due to the hyperbolic identity $\cosh^2(u) - \sinh^2(u) = 1$, this means that $\cosh^2(H/2a) - \sinh^2(H/2a) = 1$, and so:

$$\left(1 + \frac{d}{a}\right)^2 - \left(\frac{L}{2a}\right)^2 = 1$$

Expand and simplify;

$$\begin{aligned}
1 + \frac{2d}{a} + \frac{d^2}{a^2} - \frac{L^2}{4a^2} &= 1 \\
\frac{2d}{a} + \frac{d^2}{a^2} - \frac{L^2}{4a^2} &= 0 \\
2da + d^2 - \frac{L^2}{4} &= 0 \\
2da &= \frac{L^2}{4} - d^2 \\
a &= (L^2/4 - d^2)/2d
\end{aligned}$$