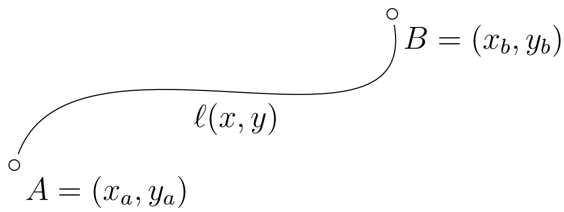
Finding a Minimum Time Path

Suppose we wish to find the path between two points, $A = (x_a, y_a)$ and $B = (x_b, y_b)$, that takes minimum time to traverse. For any particular path, the traversal time T depends on the speed of travel $v(x, y)$ as a function of location on the path, $\ell(x, y)$. That is, "time equals distance divided by rate":

$$T = \int_A^B \frac{d\ell}{v(x, y)}.$$



To help paint a more complete picture consider the special case where $\ell = \ell(x)$, that is where ℓ is a function represented as a graph in $x - y$ coordinate space. Then,

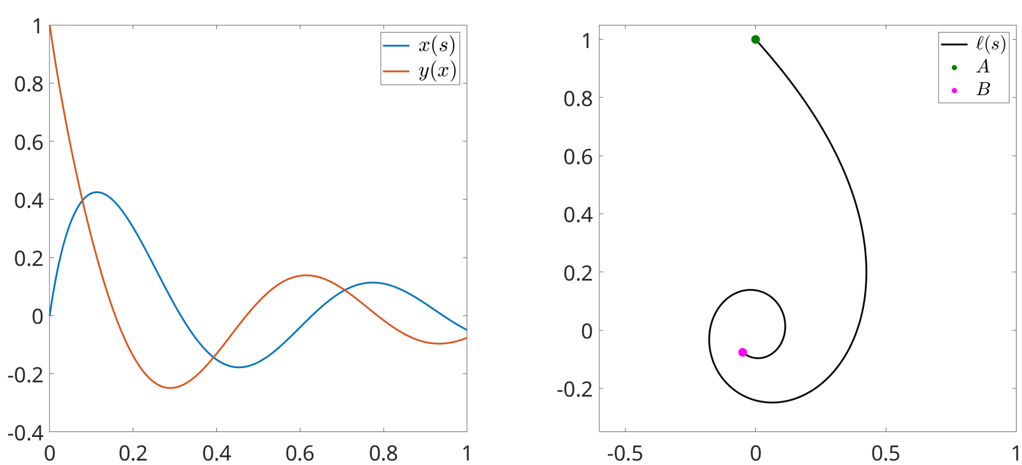
$$T = \int_{x_a}^{x_b} \frac{\sqrt{1 + \ell'(x)^2}}{v(x, \ell(x))} dx.$$

Further, if $v = 1$ we recover the familiar arc length formula. Or, for variable v, we can also consider the Brachistochrone problem. In this case $y_a > y_b$, $x_a < x_b$ and $v(x, y) = \sqrt{2g(y_a - \ell(x))}$, where g is the local gravitational acceleration constant ($g \approx 9.81 m/s^2$).

A more general formulation which does not assume that ℓ can be represented as a function is given by a parametric representation with variable $s \in [0, 1]$:

$$\ell = \ell(s) = (x(s), y(s)),\ \ell(0) = A,\ \ell(1) = B.$$

That is, $x(0) = x_a,\ x(1) = x_b,\ y() = y_a,\ y(1) = y_b$. If we specify continuous functions $x(s)$ and $y(s)$, then we have established a continuous path $\ell(s)$ from A to B. To illustrate that not every continuous path can be represented as a function, consider the specific example:



In this general case, we can write the total time as an integral in variable s:

$$T = \int_0^1 \frac{\sqrt{x'(s)^2 + y'(s)^2}}{v(x(s), y(s))} ds.$$

We will next take the step of optimizing over a family of functions defined as

$$x(s) = x_a + s(x_b - x_a) + \sum_{j=1}^{n} w_j \sin(j\pi s)$$
$$y(s) = y_a + s(y_b - y_a) + \sum_{j=1}^{n} z_j \sin(j\pi s)$$

These definitions provide sinusoidal curves of order n which satisfy our boundary conditions (check!). The weights w_j and z_j define the particular curve. We seek weights that minimize the total travel time. Our optimization problem is one of $2n$ variables:

$$\min_{w,z \in \mathbb{R}^n} \quad T(w, z).$$

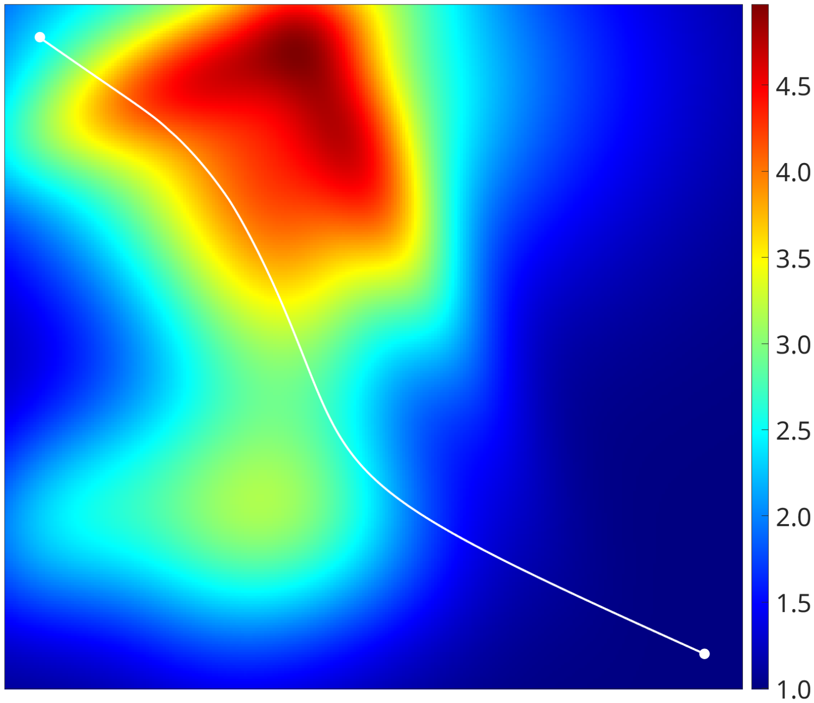
Notice that this problem is an approximate solution to the original travel time problem because the actual minimum time path may not be representable by our $x(s)$ and $y(s)$.

Next, we must be able to compute the integral $T(w, z)$. We will employ a finite differencing scheme similar to that of the aerodynamic drag problem. However, the discretization of the integral is *not* the optimization problem. Instead, we choose a *fixed* sampling set $\{s_0, s_1, \ldots, s_m\}$ satisfying $s_0 = 0$, $s_m = 1$ and $s_{k+1} - s_k = 1/m$, for some large m. Then we define $x_k = x(s_k)$ and $y_k = y(s_k)$. The time integral is then approximated as

$$T(w, z) = \sum_{k=1}^{m} \frac{\sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2}}{v\left(\frac{x_k + x_{k-1}}{2}, \frac{y_k + y_{k-1}}{2}\right)},$$
$$x_k = x_a + \frac{k}{m}(x_b - x_a) + \sum_{j=1}^{n} w_j \sin(jk\pi/m)$$
$$y_k = y_a + \frac{k}{m}(y_b - y_a) + \sum_{j=1}^{n} z_j \sin(jk\pi/m)$$

The numerator of each summand of T is the linear approximation of the path length from (x_{k-1}, y_{k-1}) to (x_k, y_k). The corresponding denominators are the speeds at the midpoint of each linear approximation.

An example of a minimum time path is illustrated in the figure below. The color scale shows the velocity function on the area if interest. The white line is the minimum time path between the two end points. Notice that the path favors regions of high speed, but not at the expense of creating a very long path.



This example uses $n = 12$ and the optimization was performed using BFGS. The speed in T is not computed directly from a given function, but is interpolated from a 256×256 array of given values.

Complete the following tasks.

1. Discuss how one might choose useful and effective problem parameters m and n.
2. Download the provided objective computation code and speed matrix. Verify that the minimum time path of the example is as shown in this document. That is, verify that your code is working correctly.
3. Find minimum time paths for $n = 1, 2, 3, 5, 8, 13, 21$. Compare the results and discuss.
4. Discuss how you might include obstructions in your area of interest.
5. Create an interesting speed function (or array) of your own and find the minimum time path between two points of your choosing.