Optimization Homework 1: Introduction

Mark C. Anderson

January 27, 2021

1 Unconstrained Optimization: The Brachistochrone Problem

The Brachistochrone problem asks us to find the minimum-time trajectory for a bead to travel along a curve from a high point to a low point. The problem is further complicated by adding friction. The physics is discussed and defined in reference [1].

For the current study, I minimized the expression

$$f = \sum_{i=1}^{n-1} \frac{\sqrt{\Delta x_i^2 + \Delta y_i^2}}{\sqrt{h - y_{i+1} - \mu_k x_{i+1}} + \sqrt{h - y_i - \mu_k x_i}} \tag{1}$$

where

- n = total number of points
- Δx = the x-distance between adjacent points
- Δy = the. y-distance between adjacent points
- h = the height of the starting point above the ending point
- μ_k = the coefficient of kinetic friction
- x = each individual x-position along the curve
- y = each individual y-position along the curve

The minimization is accomplished by varying the y-positions of each of the nodes along the curve except for the starting and ending positions, which remained fixed at $(0, 1)$ and $(1, 0)$ respectively. The minimization was solved in MATLAB using a built-in unconstrained optimizer called 'fminunc' with the following options shown in Table 1:

Table 1: Options passed into MATLAB's fmincon function.

Option	**Value**
MaxFunctionEvaluations	100,000
MaxIterations	1,000
StepTolerance	1e-9

Once the minimal-time curve is determined, the travel time is calculated by

$$t = \sum_{i=1}^{n-1} \Delta t_i = \sum_{i=1}^{n-1} \sqrt{\frac{2}{g}} \frac{\sqrt{\Delta x_i^2 + \Delta y_i^2}}{\sqrt{h - y_{i+1} - \mu_k x_{i+1}} + \sqrt{h - y_i - \mu_k x_i}} \tag{2}$$

where $g = 9.81$ m/s^2. The results of this optimization for $n = 12$ are shown in Figure 1. Note that because the boundaries are fixed, the total number of design variables is really $n - 2 = 10$ for this case. The travel time for this case is 0.638 seconds. The starting guess for this function was a straight line at $y = 1$, with the endpoint adjusted to be at a height of 0.

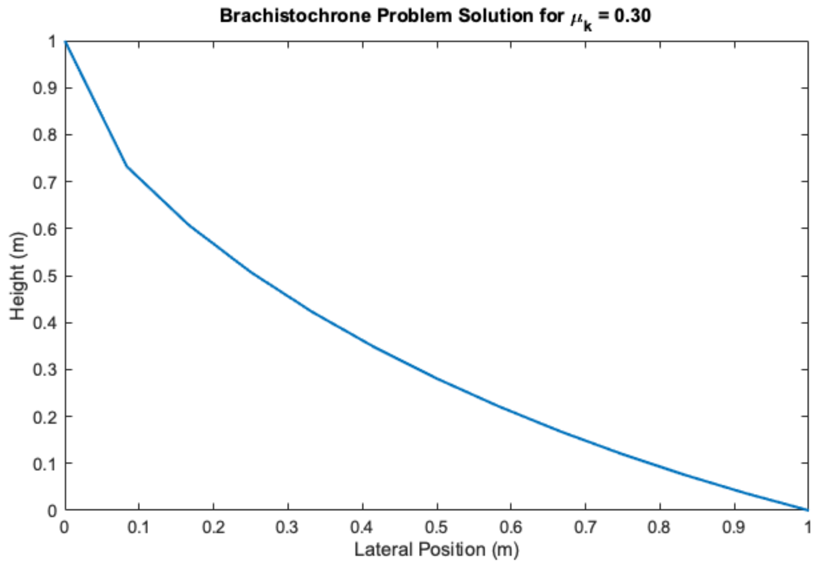

Figure 1: The unconstrained Bracistochrone problem with n = 12 (10 design variables)

1.1 Increasing the Number of Design Variables

We now investigate the impact of the number of design variables. The different paths calculated by each case are shown in Figure 2 and the travel time is shown in

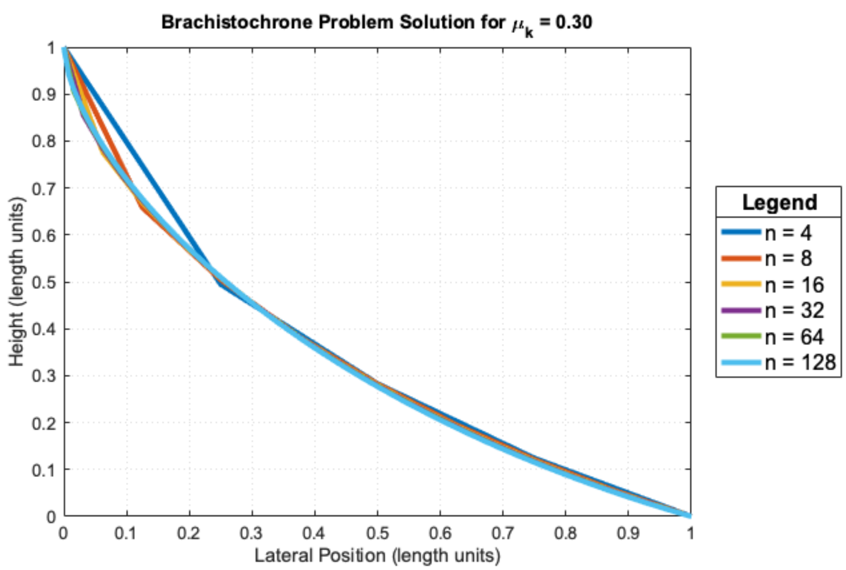

Figure 2: The different curves calculated by increasing the number of design variables. Note that because the end points are fixed each the total number of design variables is $n - 2$.

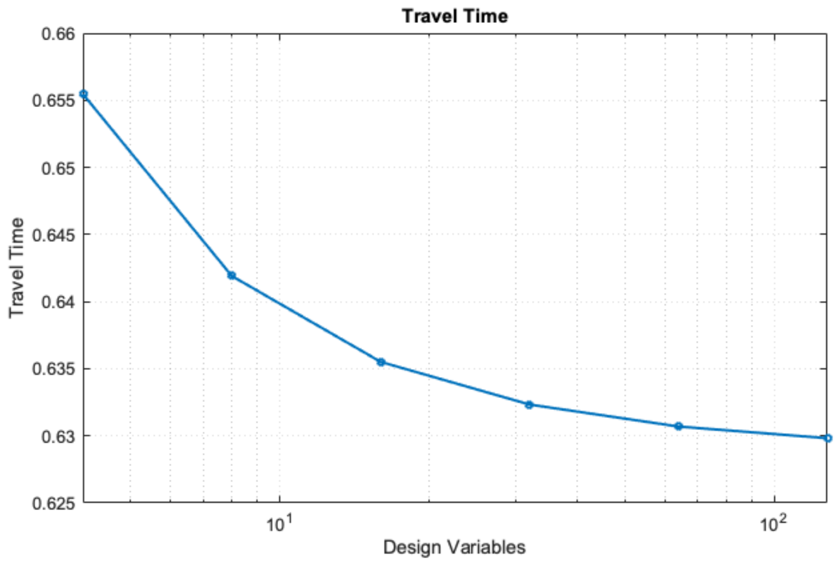

Figure 3: The travel time calculated by each calculation involving different numbers of design variables. Notice that the answer appears to be converging as the number of design variables is increased.

Improved precision comes at a cost. As the number of design variables is increased, the computational cost increases. Figure 4 shows the increase in the number of function calls with design variable number and Figure 5 shows a similar variation in the number of iterations. Both show what appears to be an exponential increase in computational cost with the number of design variables.

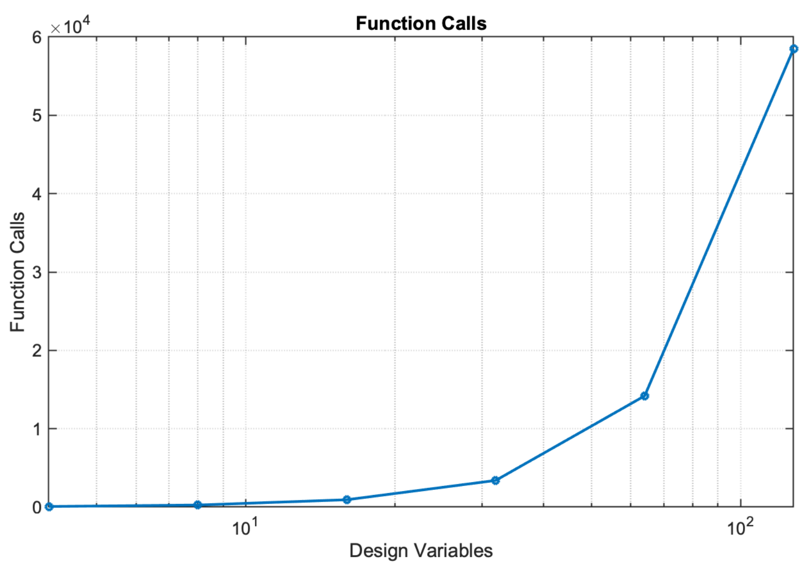

Figure 4: The number of function calls as a function of design variable number.

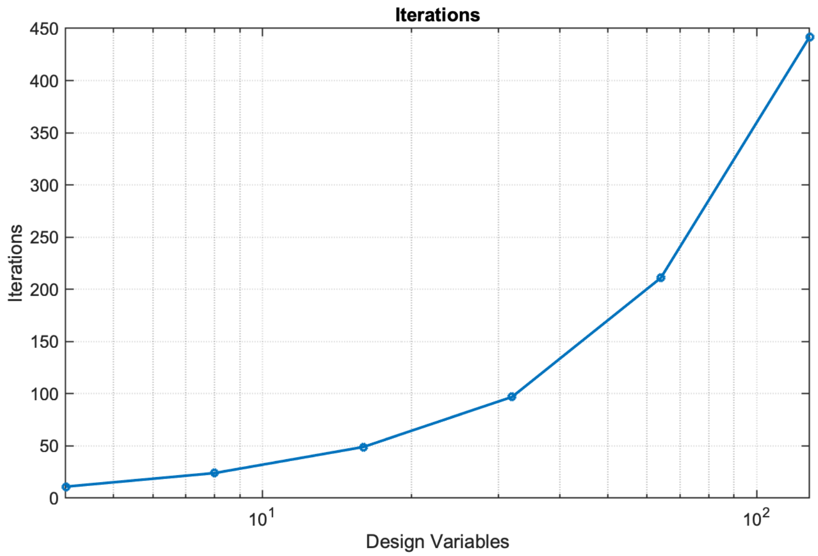

Figure 5: The number of iterations as a function of design variable number.

2 Constrained Optimization: Ten-bar Truss Structure

The ten-bar truss problem is defined in the reference [1]. The structure is shown in Figure 6. Each bar is subject to the constraint that it can have 25,000 psi on it, except for bar number 9, which can have 75,000 psi on it because it is made of a stronger material. Additionally, for manufacturing reasons, the bars must each have a cross-sectional area of no less than 0.1 in^2.

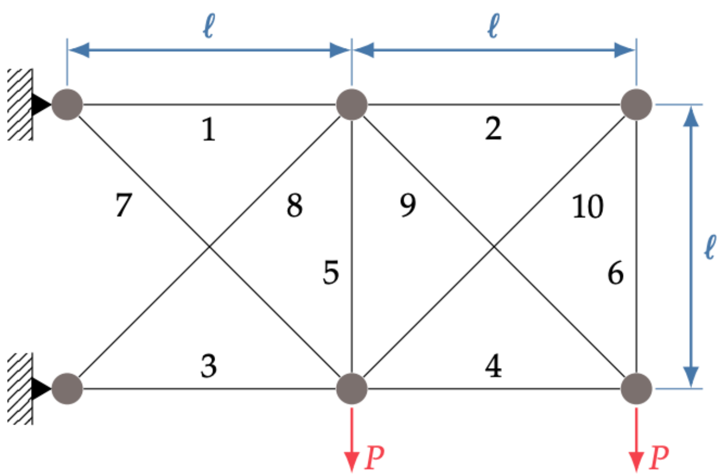

Figure 6: The ten-bar truss structure. Each bar can have 25,000 lbs of stress on it, except for bar number 9, which can have 75,000 lbs of stress on it.

The initial guess was that every bar would have a cross-sectional area of $0.1 + 1e10^{-9}$, which is just barely above the lower bound on the area for each bar. This was chosen because larger guesses did not lead to convergence in a reasonable amount of time, and if the initial guess exactly equaled the lower bound then MATLAB gave a warning that the initial guess was automatically shifted to be within the bounds. Therefore, an initial guess was chosen that was barely within the bounds.

The optimal mass was found to be 1,500 lbm, when rounded to two significant digits. The cross-sectional areas are in Table 2. A convergence plot of the first-order optimality is shown in Figures 7 and 8. The solver seems to have gotten stuck at a first-order optimality of around 50 for a while before coming across a path that led to sufficiently small first-order optimalities. A first-order optimality of about 50 was a common value for the optimizer to get stuck at without converging when the initial guess for the cross-sectional areas was larger than what was used here. The total number of function calls in this successful case to the ‘truss.m‘ function was 1,449.

Table 2: The cross-sectional areas for each of the ten bars in the structure. Notice that several of the bars are fixed at the lower bound of manufacturability.

Bar Number	**Area (in^2)**
1	7.90
2	0.10
3	8.10
4	3.90
5	0.10
6	0.10
7	5.80
8	5.52
9	3.68
10	0.14

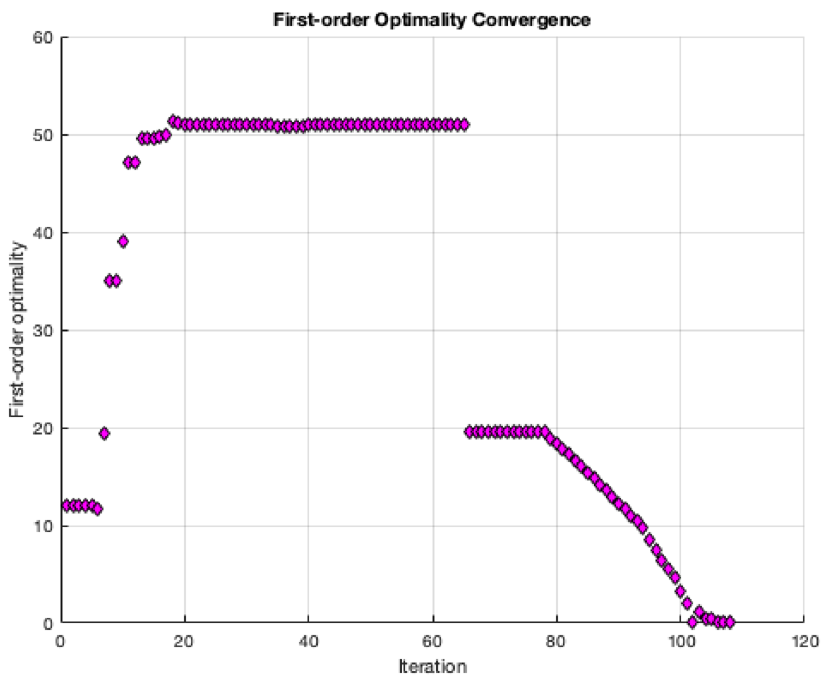

Figure 7: Convergence of the first-order optimality on a linear y-scale.

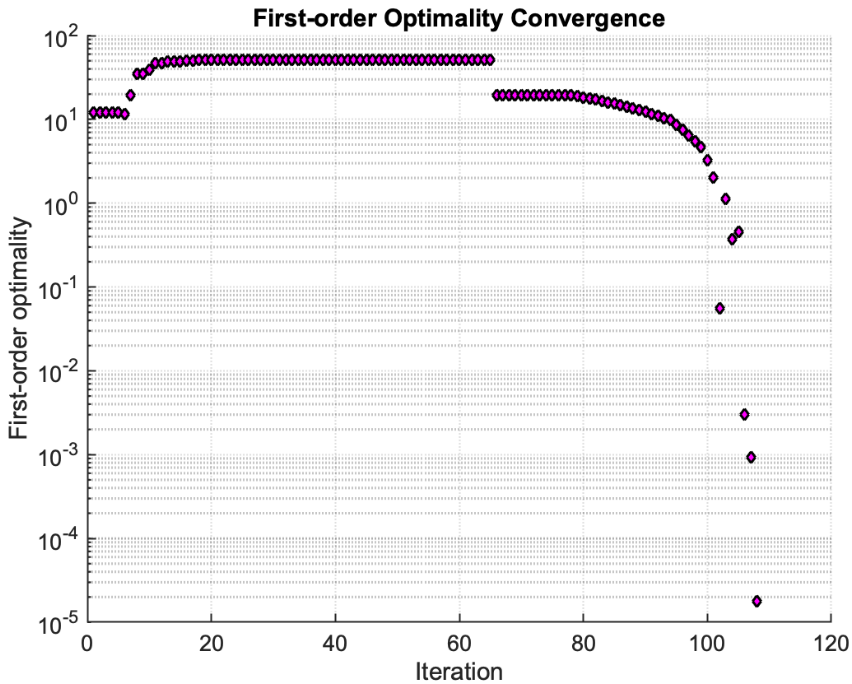

Figure 8: Convergence of the first-order optimality on a logarithmic y-scale for additional detail at small values.

References

[1] Martins, J. R. R. A., & Ning, A. (2021). Engineering Design Optimization. http://flowlab.groups.et.byu.net/mdobook.pdf