Assignment #1: Intro to Optimization — ME 575

due 1/27/2021 before midnight via Learning Suite — **50 possible points**

Overview: These two problems involve using existing optimization algorithms. The goal is to help you become familiar with the process before we start diving into algorithmic details. The first problem is unconstrained, and the second adds constraints.

1.1 Unconstrained Brachistochrone Problem: Solve the Brachistochrone Problem (with friction) using an unconstrained optimizer. The problem is defined in our textbook in Appendix C.1.7.

Complete the following (and as always be sure to describe your solution process along with showing results):

(a) Plot the optimal shape with $n = 12$ (10 design variables).

(b) Report the travel time between the two points. Don't forget to put $g = 9.81$, the acceleration of gravity, back in.

(c) Study the effect of increased problem dimensionality. Start with 4 points and double the dimension up to 128 (i.e., 4, 8, 16, 32, 64, 128). Plot and discuss the increase in computational expense with problem size. Example metrics include things like major iterations, functional calls, wall time, etc. Hint: When solving the higher-dimensional cases, it is more effective to start with the solution interpolated from a lower-dimensional case—this is called a warm start.

1.2 Constrained Truss Problem: Solve the ten bar truss problem defined in C.2.2 of the book. Code to analyze the truss is available in our resource repository: `https://github.com/mdobook/resources`

Report and discuss the following:

(a) the optimal mass and corresponding cross-sectional areas

(b) a convergence plot: x-axis should be some measure of computational time (e.g., major iterations, function calls) on a linear scale, the y-axis should be some measure of convergence. If your solve gives you access "first order optimality" that is ideal (we will learn what that means later), but other reasonable metrics can be used.

(c) the number of function calls required to converge (functions calls from the `truss` function)