Homework 4

July 6, 2020

1 Homework #4

Due July 15 @ 11:59pm

1.1 Submission requirements

Upload a **single PDF file** of your IJulia notebook for this entire assigment. Clearly denote which question each section of your PDF corresponds to.

1.2 Problem 1 – Tradeoffs and Regularization

Suppose we are planning a route for a new autonomous vehicle that will be used to map out the terrain in a certain area. The terrain has quite a few hills and valleys, so the vehicle will need to be able to speed up (to make it up a hill) or slow down (to keep from losing control going downhill). We would like to plan our speed changes based on the elevation analysis we have already performed. There are 300 discrete points where we will need to change speed. Speed changes are "one-dimensional"; a speed change of 1 means we have increased our speed by 1 mile per hour. A speed change of -1 means we have decreased our speed by 1 mile an hour. The desired sequence of speeds is given by the plot produced by the code snippet below.

Abrupt changes in speed are bad for the vehicle and could potentially cause a loss of control of the vehicle. We would like to find an alternative sequence of speed changes that gives us a tradeoff between matching the desired speeds in the graph below and a sequence of speed changes that gives us desirable properties. In particular, we will be interested in making sure our speed transitions are smooth (not too abrupt). If we denote our speed changes at each discrete time by $u_1, u_2, ..., u_{300}$, we can characterize smoothness by using the sum of the squared differences between each successive speed:

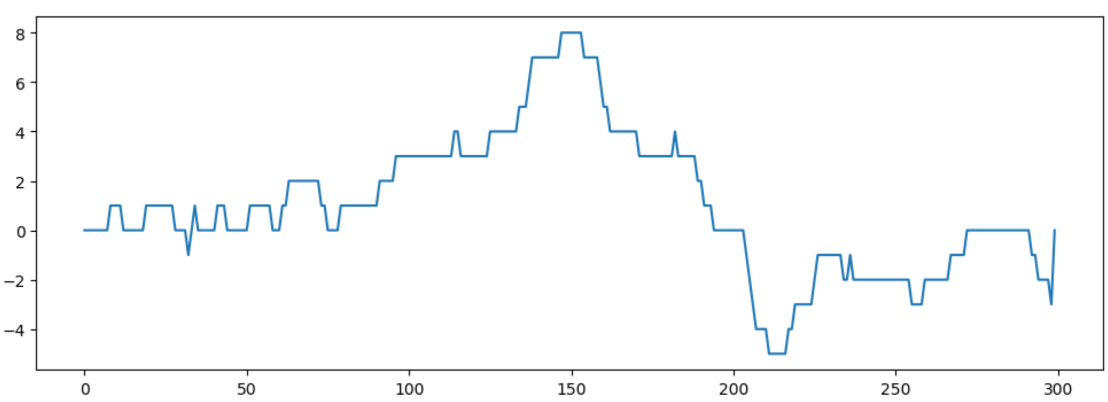
$$R(u) = (u_2 - u_1)^2 + (u_3 - u_2)^2 + \cdots + (u_{300} - u_{299})^2$$

Of course, the smaller we make $R(u)$ the smoother our transitions will be.

Find a set of optimal sequences of speed changes that explores the tradeoff between matching the desired seuqence given in the graph below and keeping the transitions smooth. Include a plot comparing the desired speeds to at least 4 different smoothed versions. Use regularization weights of 0.1, 1, 10, and at least one other weight that you choose. Also plot your solutions on a Pareto curve. To "fill in" the rest of the curve, either draw or describe in a few words what you think the Pareto curve would look like if you filled in all the points.

Use the code provided below to generate data for your model.

[10]:
```
using Random
# set a seed so we get the same output every time
seed = 393845
Random.seed!(seed)
# initialize the vector of speeds
val = 0; u = zeros(300); u[1] = val
# set a density that determines how often the speed changes
# low density corresponds to infrequent speed changes
dens = 0.1
# build speed vector for all times between now and time 299
for i in 2:299
    # if a uniform(0,1) variable is < density
    if rand() < dens
        # increase the speed by 1 mph
        val = val + 1
        u[i] = val
        # if a uniform(0,1) variable is >= 1 - density
    elseif rand() >= 1.0-dens
        # decrease the speed by 1 mph
        val = val - 1
        u[i] = val
    else# otherwise the speed stays the same
        u[i] = val
    end
end
# the final speed must be 0
u[300] = 0
# T = 300
T = length(u)
# plot the speeds (your figure should match the one in the assignment!)
using PyPlot
figure(figsize=(12,4))
plot(u,"-");
```



1.3 Problem 2 – Polynomial Regression

In this question, we will revisist the setting of Problem 1 but using a different approach. Suppose our task is the same: we want to match our speed changes to the desired sequence of speed changes given in the graph above. We also want to keep the transisions relatively "smooth." In Problem 1, we accomplished this by using regularization. Now, suppose instead we have been told that our vehicle can only change speed in such a way as to follow a polynomial curve where the speed is a function of time. We can program our vehicle to make speed changes according to a polynomial of degree 3. Perform polynomial regression on the data given in Problem 1 for a polynomial of degree 3.

(a) What are the coefficients of your polynomial?

(b) Graph the polynomial against the desired speed changes from Problem 1. Does it look like a good fit for your data?

(c) What is the total 2-norm error between the desired speed changes and the speed changes produced by your polynomial? In a broad sense, how does it compare to what you found in Problem 1? What conclusions can you draw by comparing these two techniques?

You can use any technique you like to solve this problem, but do not build an optimization model in JuMP!