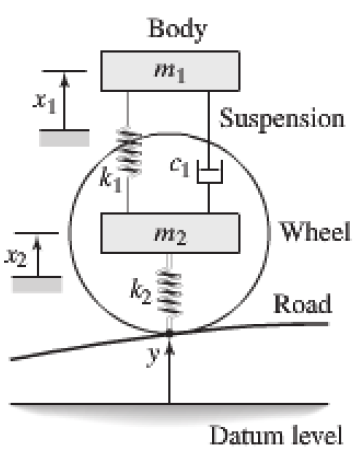
1. (40 points)

On exam 3, we created a program to model the vertical position of a car that is rolling along at constant speed and hits a linear ramp to a new level of y=6 inches. We modeled the behavior of the car using only a single wheel (i.e., the quarter car model) and treated the tire as a spring between the road and the hub of the wheel and the suspension as a spring and dashpot in parallel between the wheel hub and the car body.

To estimate the spring constants, we could guess that a static compression of the spring of the suspension should be between say 3” to 6” and for the tire between 1.5” to 0.75”. Given a car with a mass of 0.25mcar=450kg, use minimize from scipy.optimize to calculate the best combination of k1, c1 and k2 for our car model to minimize the sum of squared errors between the vertical position of the car and the contour of the road.



Necessary steps:

1. Calculate the minimum and maximum values for k1 and k2 based on the static compression values.
2. Calculate the sum of squared errors SSE as Σ(ycar(t)-yroad(t))2 with penalties if k1 or k2 violate the limits from 1.
3. Use minimize(SSE, x0, method=’Nelder-Mead’), where

x0=[self.model.mink1, self.model.c1, self.model.mink2]

1. You will want to update the values for k1, c1, and k2 on your GUI to show the optimized values and update the plot.

A program stem has been provided on canvas with the #$MISSING CODE$ marker to help you find where to focus your effort.

EXTRA CREDIT OPPORTUNITY: This simplified optimization of the suspension focuses solely on following the road contour as closely as possible. In reality, this would likely make for a bumpy ride since we are not imposing any limits on accelerations acting on the passengers.

**For 20 bonus points (added to your exam total)** both plot the acceleration experienced by the car passengers and limit the acceleration to something reasonable (say ~1.0g) in your suspension optimization.