MATLAB Problem 3

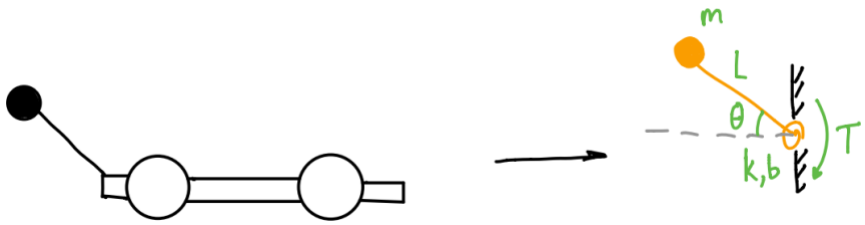
A quick note before you start – you will answer all questions for this problem within this Word document. When finished, please save the document as a .pdf, which you will upload directly to Gradescope. **For this and future MATLAB problems, we will also ask you to upload your code to CCLE.** This can be done through the same CCLE "assignment" from which you downloaded the problem set files. You should turn in all 3 of the .m files that you use in this problem.

In the field of autonomous robotics, there has recently been a trend toward "biomimetic design," or studying and replicating evolved characteristics of nature's most agile movers (click the links that follow to check out some sweet videos). Wall-climbing robots use gecko-like micro-spines to scale sheer surfaces with ease. Insect-like robots fly to a vertical surface, perch, climb, and then take off again. Cheetah-like robots, built with actuators that mimic biological muscle and tendon, offer unprecedented maneuverability and dexterity in a small package. Biomimetic design is motivated by a recognition that nature has millions of years of experience producing biological machines that adeptly navigate their environment, and that we would do well to draw from nature's expertise as we seek to do the same.

In this problem, we're going to focus on an adaptation that is wide-spread in the animal kingdom, and almost completely absent in the majority of today's robots: the use of a dynamic tail! Some fun studies out of UC Berkeley have explored how lizards use their tails for mid-air adjustments while jumping, and applied the same ideas to a wheeled robot. We are going to consider a simplified version of this fascinating problem, and explore how changing the system's dynamics affects overall behavior.

3a: Build a mathematical model and analytically solve the resulting differential equation.

The simplified robotic tail system can be modeled according to the diagram below.



Here, we have abstracted away the wheeled robot and represented it as a rigid wall, with a tail connected to it via a hinge joint. For our purposes, we've also chosen the robotic tail to be a *passive* dynamic system, meaning that its behavior is governed by 0th, 1st, and 2nd order elements (spring, damper, and inertia, respectively). The tail has been reduced to a point mass m at a distance L from the center of the tail's rotation, such that the tail's inertia is $J = mL^2$. The rotational spring-damper has spring constant k and damping constant b. All perturbations to the robot are represented by an external torque T acting on the system. For this problem, we can ignore the effects of gravity on the mass.

Your first task is to find the differential equation that models this system when there is no external torque acting on the tail ($T = 0$). For this first pass, let's say $m = 0.4\ kg$, $L = 0.5\ meters$, $b = 0.3\ \frac{Nms}{rad}$, and $k = 0.2\ \frac{Nm}{rad}$. Initial conditions are $\theta(0) = -1\ rad$ and $\dot{\theta}(0) = 0\ \frac{rad}{s}$. **Find a mathematical model that characterizes this system, and classify the equation as completely as possible. Once you have your model, solve the initial value problem analytically, using the methods we learned in class. Do all of this by hand, then take a picture and insert it into this document. Make sure that your work is readable! Hint:** it may make your math easier to multiply the equation through by a factor of 10 before solving.

2nd order, linear, homogenous

Homogenous:

$$-k(\theta) - b(\theta)' + mL^2(\theta)'' - T = 0$$

$$mL^2(\theta)'' + b(\theta)' + k(\theta) - T = 0$$

$$(0.4kg)(0.5m)^2\theta'' + 0.3\frac{Nms}{rad}\theta' + 0.2\frac{Nm}{rad}\theta = 0$$

$$0.1\theta'' + 0.3\theta' + 0.2\theta = 0 \quad (\times 10)$$

$$1kgm^2\theta'' + 3\frac{Nms}{rad}\theta' + 2\frac{Nm}{rad}\theta = 0$$

2, 3, 1

$\theta = -2, -1$

$\theta = y = c_1e^{-2t} + c_2e^{-t}$

$\theta(0) = -1\,rad$

$-1 = c_1 + c_2$

$\theta' = -2c_1e^{-2t} - c_2e^{-t}$

$\theta'(0) = 0\,rad/s$

$0 = -2c_1 - c_2$

$2c_1 = -c_2$

$c_1 = -\frac{1}{2}c_2$ $\qquad c_1 = -\frac{1}{2}(-2)$

$c_2 = -1 - c_1$ $\qquad c_1 = 1$

$c_2 = -1 + \frac{1}{2}c_2$

$\frac{1}{2}c_2 = -1$

$c_2 = -2$

$\theta = e^{-2t} - 2e^{-t}$

homogenous

Based on the roots of the characteristic equation, what can we say about this system's response?

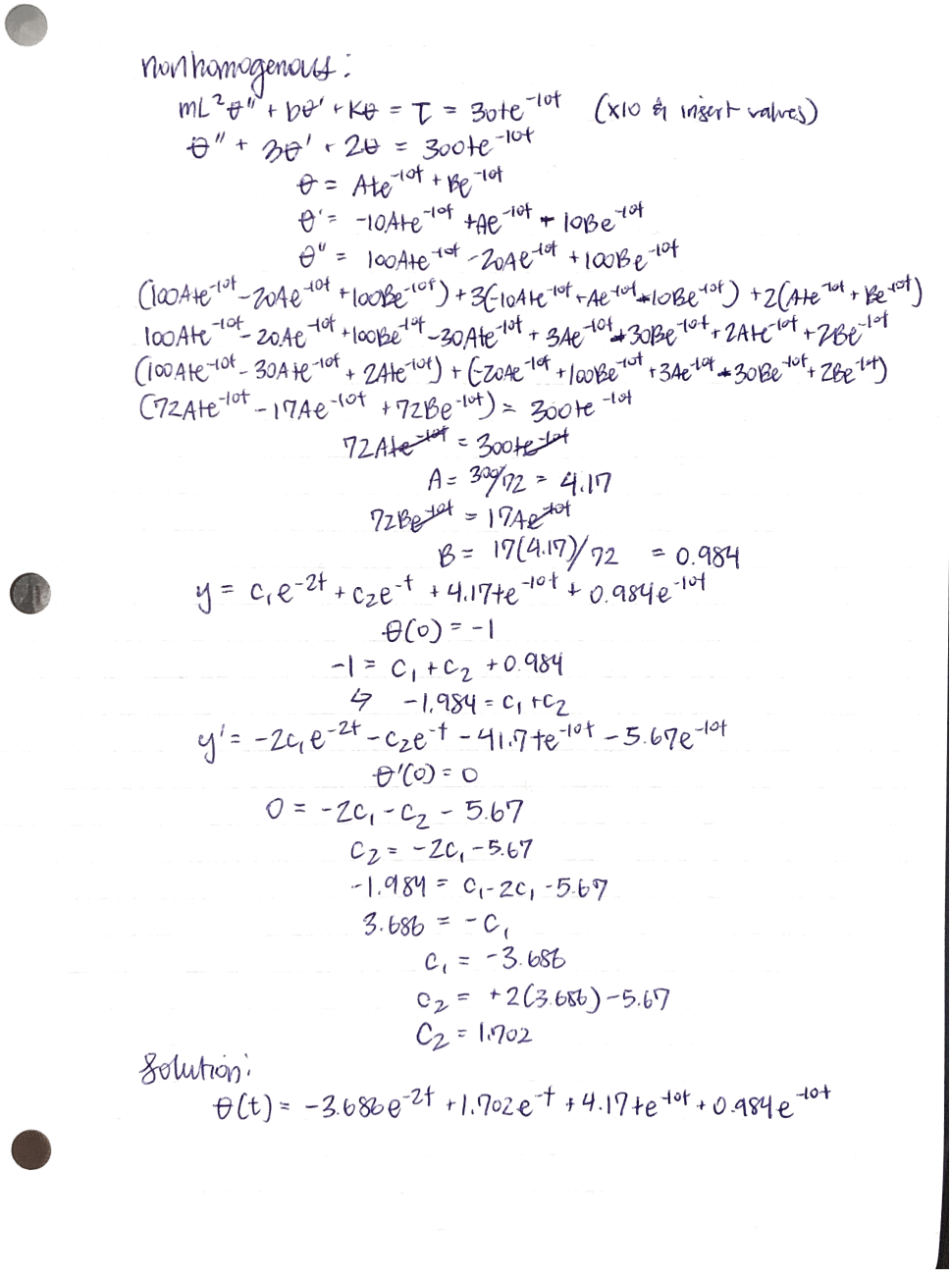
The system will not reach a phase angle of 0 since the roots are negative and we do not consider negative time.

What will this system do as $t \to \infty$?

The system approaches 0 as t approaches infinity

Now that you've solved the homogeneous system model, let's see what happens when we add an external torque $T = 30te^{-10t}$. **Hint**: don't forget that you may have already multiplied your equation by 10, and would need to adjust this accordingly. **Solve the resulting non-homogenous equation analytically, using the same initial conditions as the homogeneous equation. Do this by hand, then take a picture and insert it into this document. Feel free to convert to decimal approximations, where convenient.**

2nd order, non homogenous, linear



What will this system do as $t \to \infty$? What does this tell us about the system's response to this particular external torque?

The system also approaches 0 as it approaches positive infinity. The system would be stretched out to its max to have and be linear with the rest of the system

3b: Use ODE45 to solve the homogeneous and non-homogeneous cases.

Download Homework4_MatlabProbs.m and roboTailODEfun.m. As always, without changing any of the file names for these files, put both into a single folder. Navigate to that folder in Matlab's "current folder" pane.

Open both files in MATLAB. In the section of Homework4_MatlabProbs.m entitled "Problem 3b", your task is to use ODE45 to solve your second-order differential equation describing the tail's behavior. To do this, we're going to make use of a technique that we haven't learned in class quite yet, which involves representing our 2nd order differential equation as a system of 1st order equations. For instance, consider the equation:

$$y''(t) + p(t)y'(t) + q(t)y(t) = g(t)$$

If we let $x_1 = y(t)$ and $x_2 = y'(t)$, then we readily see that $x_1' = x_2$ and $x_2' = y''(t)$. As such, we can rewrite our 2nd order equation as the following system of two 1st order equations:

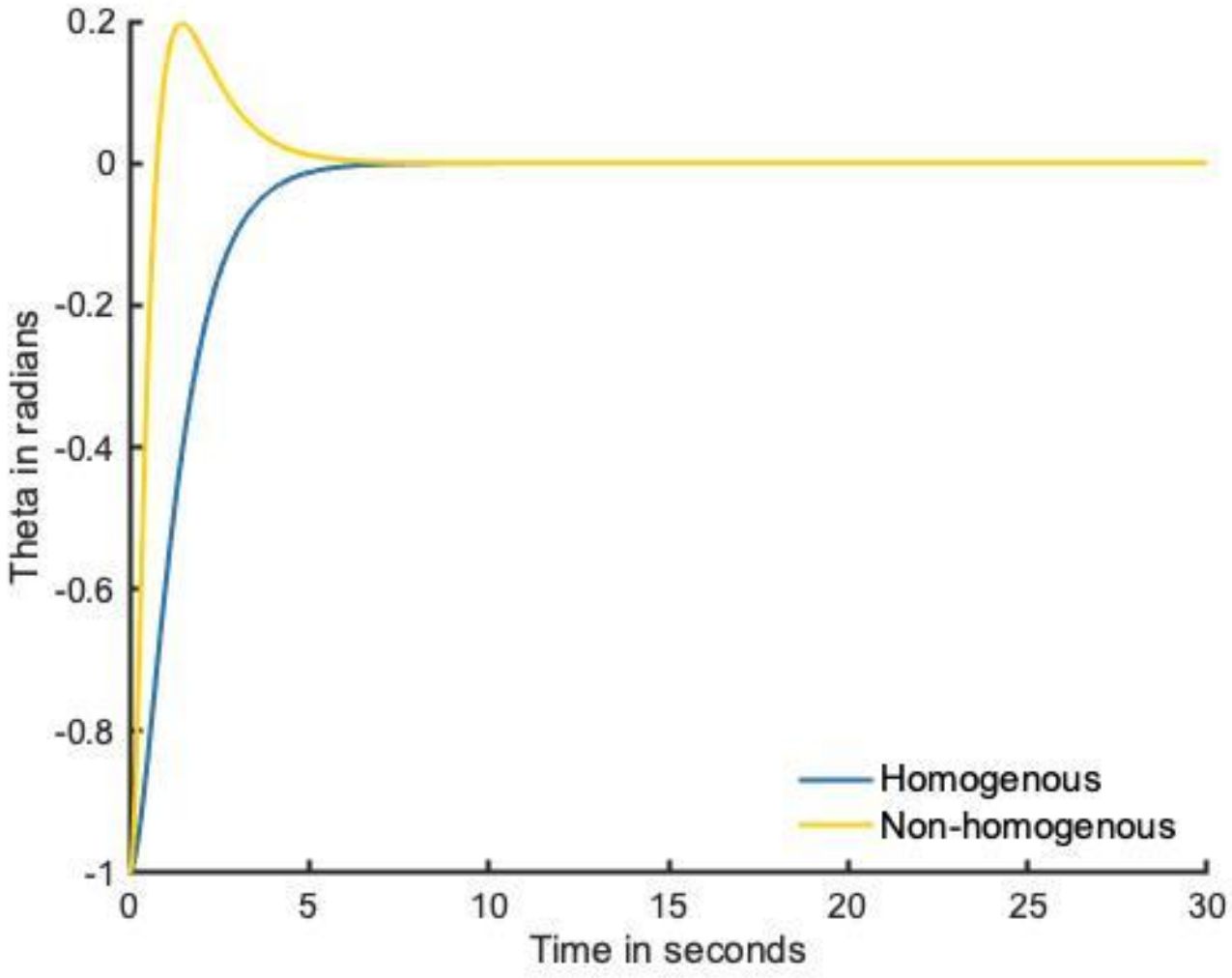
$$x_1' = x_2$$

$$x_2' = g(t) - p(t)x_2 - q(t)x_1$$

It turns out that this is an extremely helpful formulation, because ODE45 is built to handle systems of 1st order equations. Check out this example from MATLAB documentation before you continue: https://www.mathworks.com/help/matlab/math/solve-nonstiff-odes.html

This is not only useful for numerical methods; there are some great tools for analysis of linear 2nd and higher order ODEs that require reduction of the equation to a system of 1st order ODEs. We'll learn more about this later in the course.

For now, fill in the missing code in Problem 3b of Homework4_MatlabProbs.m and roboTailODEfun.m to find numerical solutions to the differential equation, for both the homogeneous and non-homogeneous cases of external torque: $T = 0$, and $T = 30te^{-10t}$. You'll want to solve these one-at-a-time, and store the results in two different sets of "y" and "t" variables. Fill in code to plot the resultant $\theta(t)$ trajectories for the two cases in Figure 1. Note that your "y" output will now be an nx2 array, where the first column contains the variable x_1 ($\theta(t)$ in your case) and the second column contains the variable x_2 ($\dot{\theta}(t)$ in your case). This means that you need to plot only the first column for each case. Be sure to follow the formatting instructions as described in the comments in the code. **Save the resultant figure as a .png or .jpg (don't just screenshot it), and insert the image into this document.**



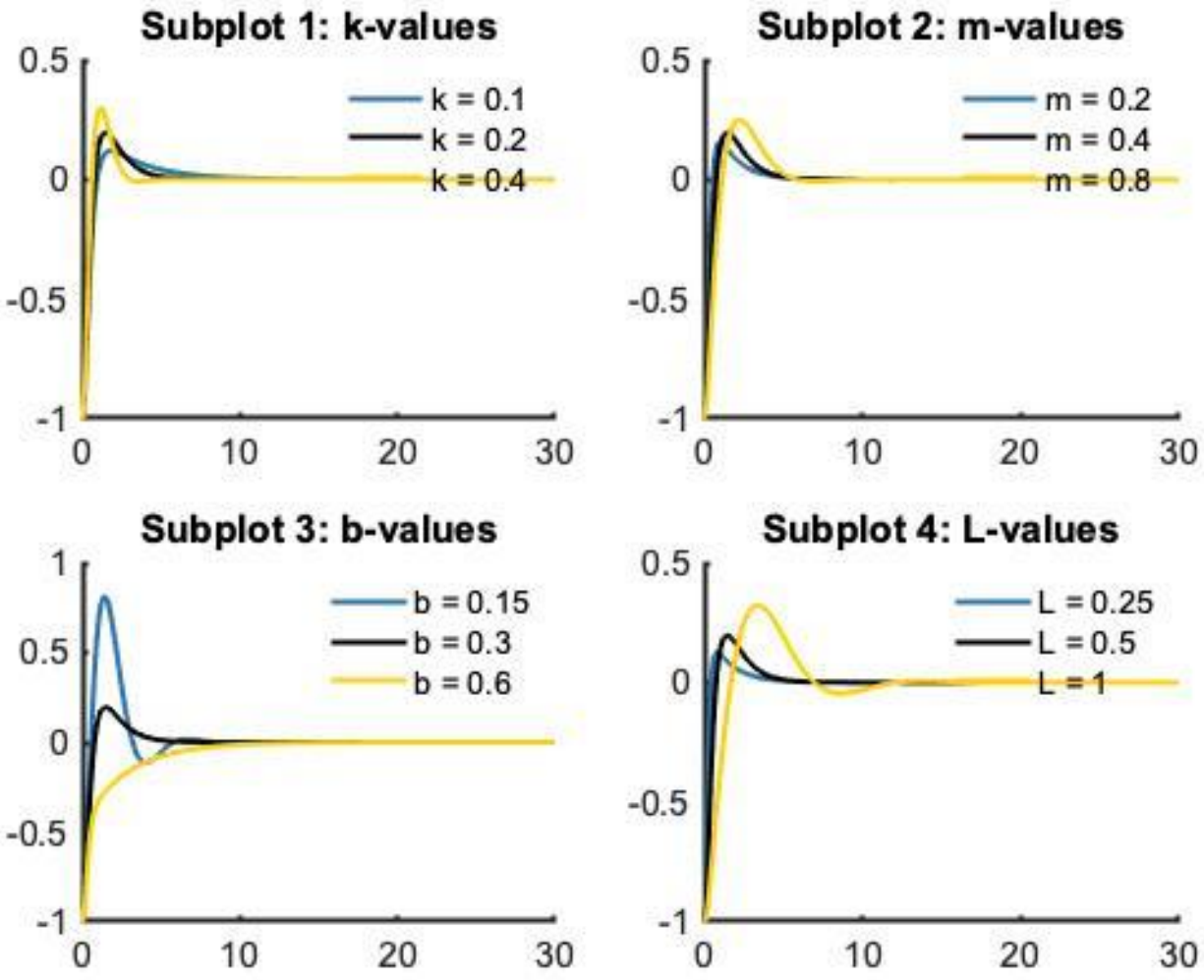
What do the two curves in this plot represent? Do these seem consistent with your analytical solutions? Hint: you might even consider using this plot to check that your analytical solutions are correct…

The two curves represent the homogenous and non-homogenous methods we solved analytically earlier as plotted functions. The non-homogenous includes the external torque and the homogenous does not. These are consistent with my analytical as verified by desmos.

3c. Explore how model parameters affect system behavior.

Next, we're going to explore the relative impact of changing each parameter on the overall system behavior. To do this, we'll use ODE45 to find a numerical solution to the non-homogeneous equation ($T = 30te^{-10t}$), and change the parameters one at a time while keeping everything else constant. For each parameter, we want to simulate system behavior at the original parameter value, at half the original value, and at twice the original value. We then want to plot these three solutions on top of each other, so we can see how they compare. We'll do this for each of the four key parameters: k, b, m, and L. Putting each of these in a subplot will make it easy to see them all at the same time. To get you started, I've done the first one for you.

Complete the code to add the remaining three subplots for b, m, and L, exactly as described above, and drop the resultant figure here.



Which of these parameters seems to have the greatest impact on system behavior? Which has the second-largest effect?

B has the greatest impact on the system's behavior as evident by the clear variation in amplitude between the different values. L has the second largest variation in system behavior.

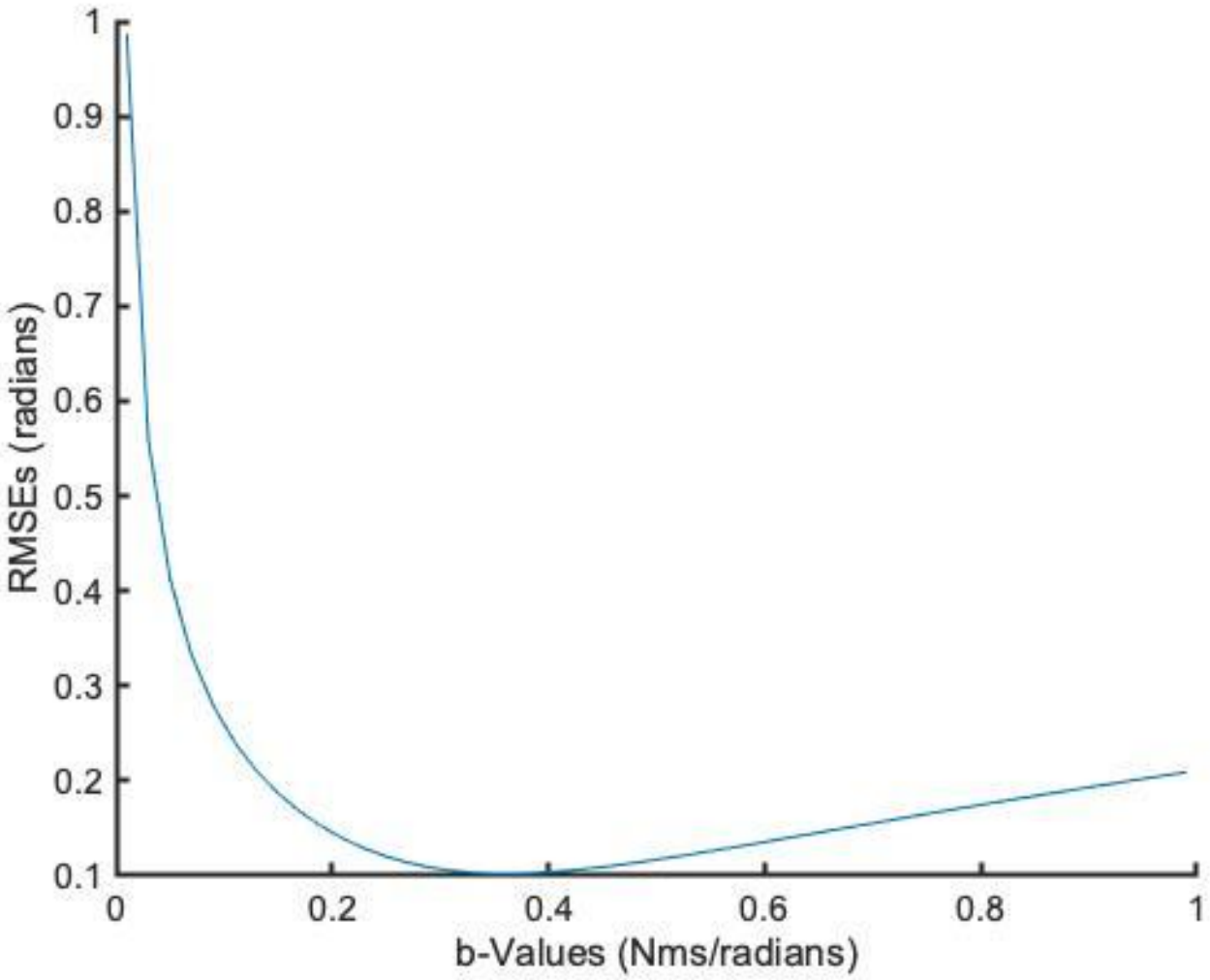
Let's play with the damping a bit, and see if we can use it to minimize the impact of the external torque on our tail's stability. In other words, we want to get close to $\theta = 0$ as quickly as possible, and stay as close as possible without oscillating. One way to quantify this is our old friend RMSE. We can compare our results to 0 by simply taking the RMS of our resultant theta vector.

Why might RMSE be an effective way to capture our objective for $\theta(t)$ as described in the previous sentence?

RMSE can effectively tell us how concentrated data is around the best fit line so we can minimize the amount of error to find the best b value to approach theta = 0.

To minimize RMSE, we'll need to run an optimization. As in MATLAB Problem 1, we'll use a brute force method. Write code to sweep through the values of b contained in the array bVals in Homework4_MatlabProbs.m, while keeping all other parameters at their original values ($m = 0.4\ kg$, $L = 0.5\ meters$, and $k = 0.2\frac{Nm}{rad}$). For each iteration, solve the differential equation numerically, and calculate RMSE of the resultant $\theta(t)$. Make sure you interpolate your theta vector to the testTime vector before finding

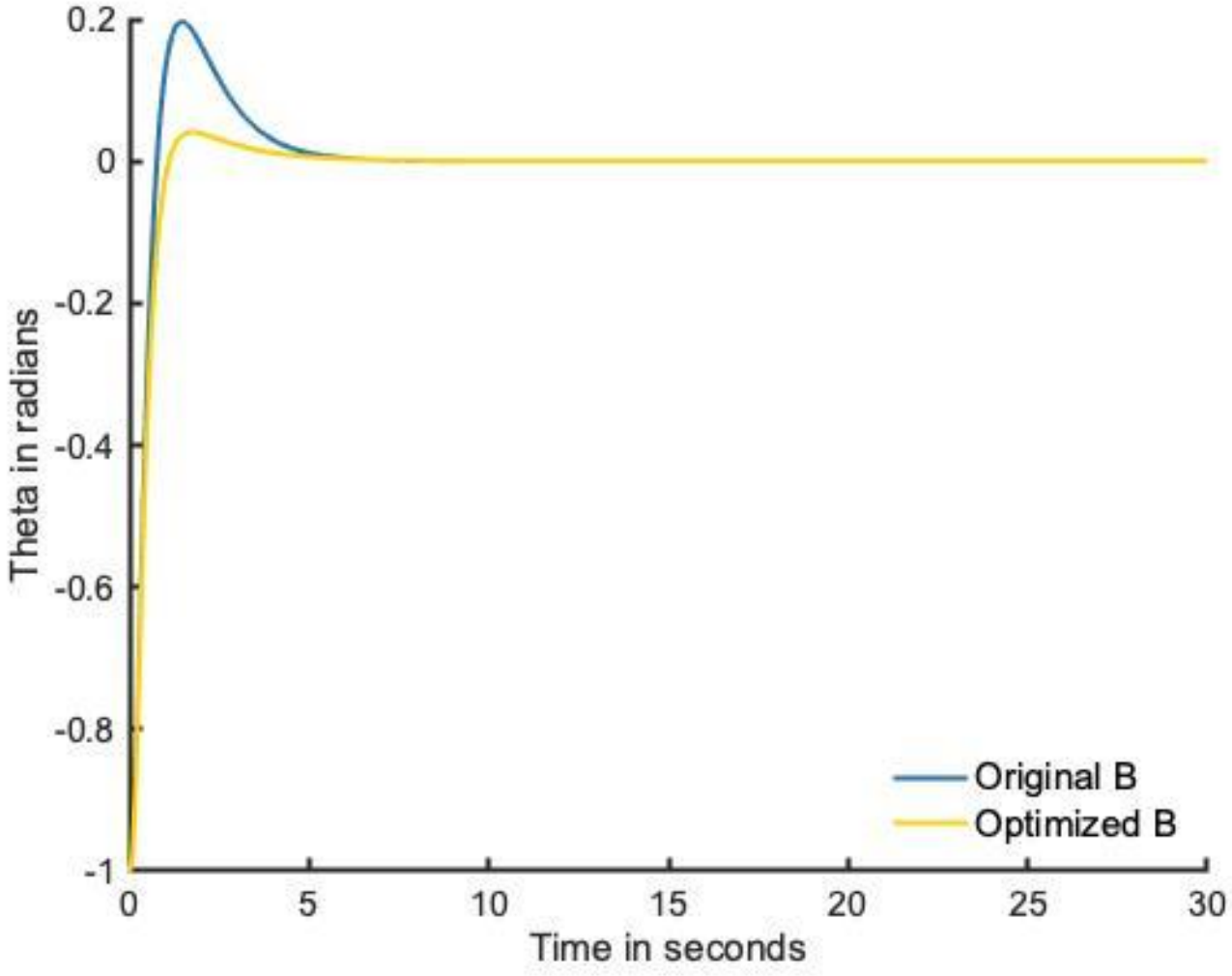
the RMSE (see MATLAB problem 1c). Store these RMSE values, and make a plot of b versus RMSE. **Insert the resultant figure here.**



Based on this plot, does there appear to be an optimum value of b?

There appears to be a minimum or optimum value at 0.35

It's time to see what your optimization produced. Write code to find the value of b that produces the minimum RMSE. Then, re-run your numerical simulation using that value of b, and plot the results on top of the non-homogeneous result from 3b. **Add your plot here.**



What value of b produced the minimum RMSE?

0.37 Nms/rad

What was the minimum RMSE? Make sure to include units.

0.1025 rad

How does this compare to the RMSE produced using the original value of b from 3a?

It is greater than the RMSE of the original value by about 0.0031 rad

3d. Adding in a non-linear spring.

We are now going to take advantage of MATLAB's computational power to solve a more challenging version of the same problem. In nature, it turns out that most spring-like materials behave non-linearly. This is especially true for muscles and tendon, which "harden" as they are stretched. One way to model this is to represent the spring constant k as a function of joint angle θ. For this problem, let's say that the physical behavior of the spring changes to $T_{spring} = k(\theta) * \theta$, where $k(\theta) = (\alpha\theta)^2$. **Write out the new differential equation that includes this updated spring model in the presence of external torque $T = 30te^{-10t}$, and classify the equation as completely as possible. You can do this by hand or using Word's built-in equation editor.**

Second order, non homogenous, nonlinear

$$T_{spring} = k(\theta)\cdot\theta = (a\theta)^2$$

$$T = 30te^{-10t}$$

$$mL^2(\theta'') + b(\theta') + k(\theta) = T$$

$$J = mL^2$$

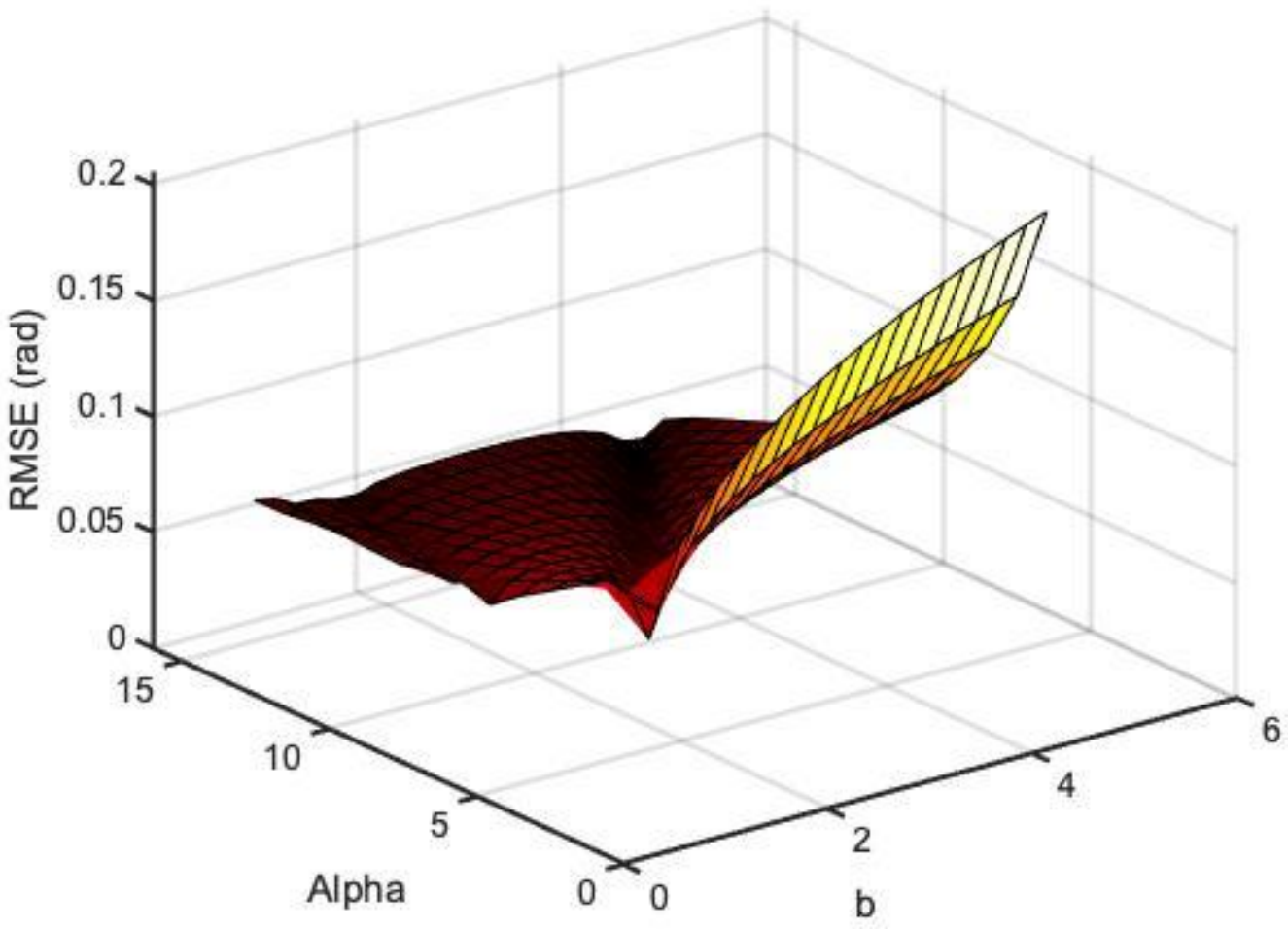
$$J(\theta'') + b(\theta') + a^2\theta^3 = T$$

$$\boxed{\theta'' + \frac{b}{J}(\theta') + \frac{a^2\theta^3}{J} = \frac{T}{J}}$$

Next, we need to add this new equation to our code base. Create a new file called "roboTailODEfunVarK.m" to contain this new differential equation. I would recommend making a copy of roboTailODEfun.m, changing the filename and function name (be sure that the function name matches the new filename), and updating the code to calculate this new value for k. **Hint:** this new function should no longer have k as an input; however, α should now be an input.

Now that you've created this new function, we're going to use it to brute-force optimize our values of b and α, with a goal of minimizing RMSE. Add code to Homework4_MatlabProbs.m to carry out this optimization in both parameters simultaneously, over the pre-specified ranges. Keep all other parameters at their original values from problem 3a. Create a 3D surface plot of the resultant RMSEs versus b and α. Note that this optimization should take several minutes to run. **Hint:** this code will look very similar to the optimization code we used in MATLAB Problem 1.

Insert your 3D surface plot here.



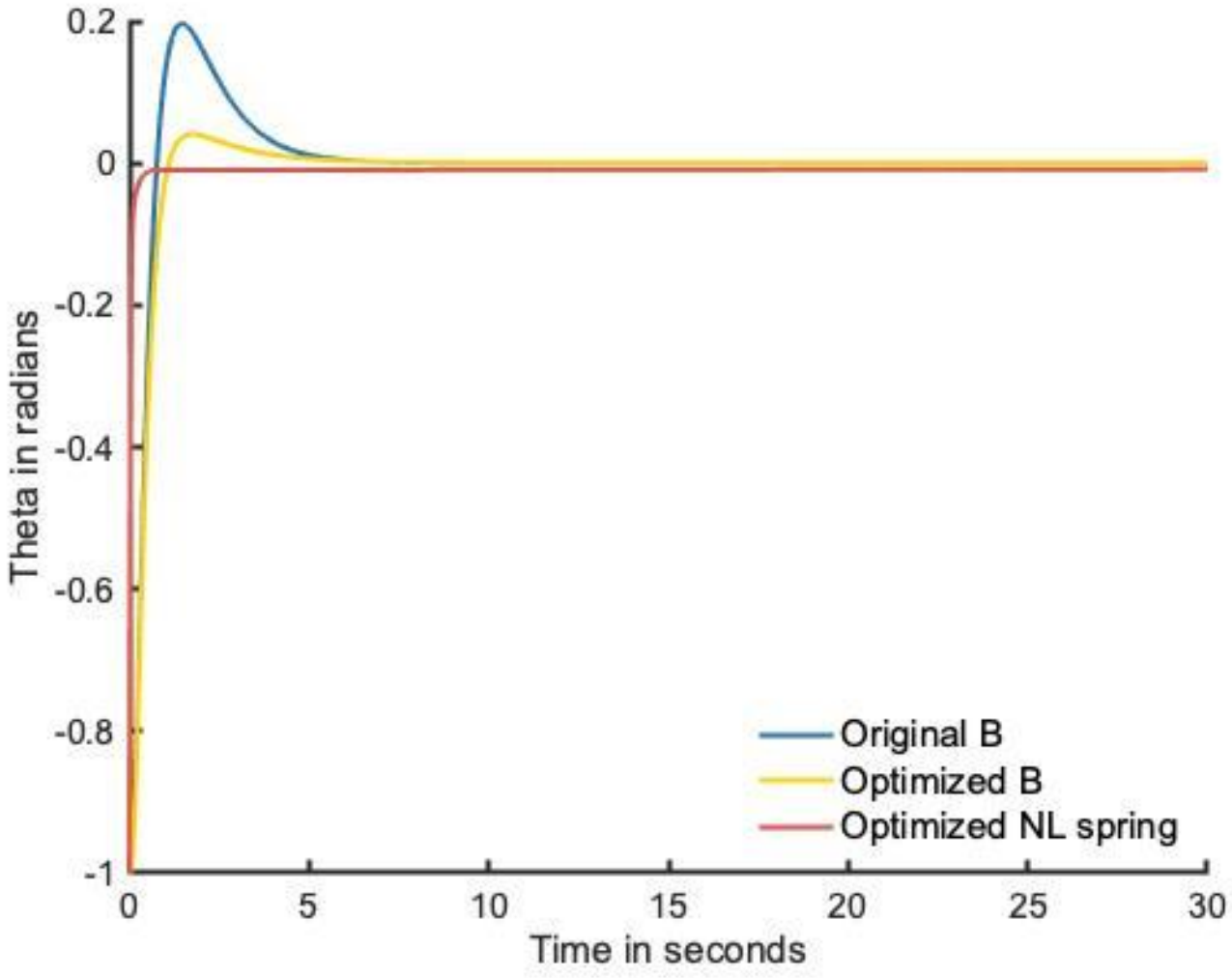
Comment on the shape of this surface. What does this tell you about the relationship between b, α, and RMSE?

For b, RMSE increase as the value of b increases. For a, the RMSE increases as the value of a decreases. There is a diagonal valley in between a and b values where RMSE is low. This linear relationship is where the lowest RMSE can be found.

Based on this plot, are you convinced that we've found the true minimum value of RMSE?

No, we have to calculate the actual minimum but through inspection the minimum value should fall in the valley.

Let's look at our optimized results. Add code to extract the minimum RMSE, as well as the optimized values of b and α. Re-run your roboTailODEfunVarK.m model using these optimized values. Plot the resultant $\theta(t)$ on top of the non-homogeneous result from 3b and the optimized result from 3c. **Add your plot here.**



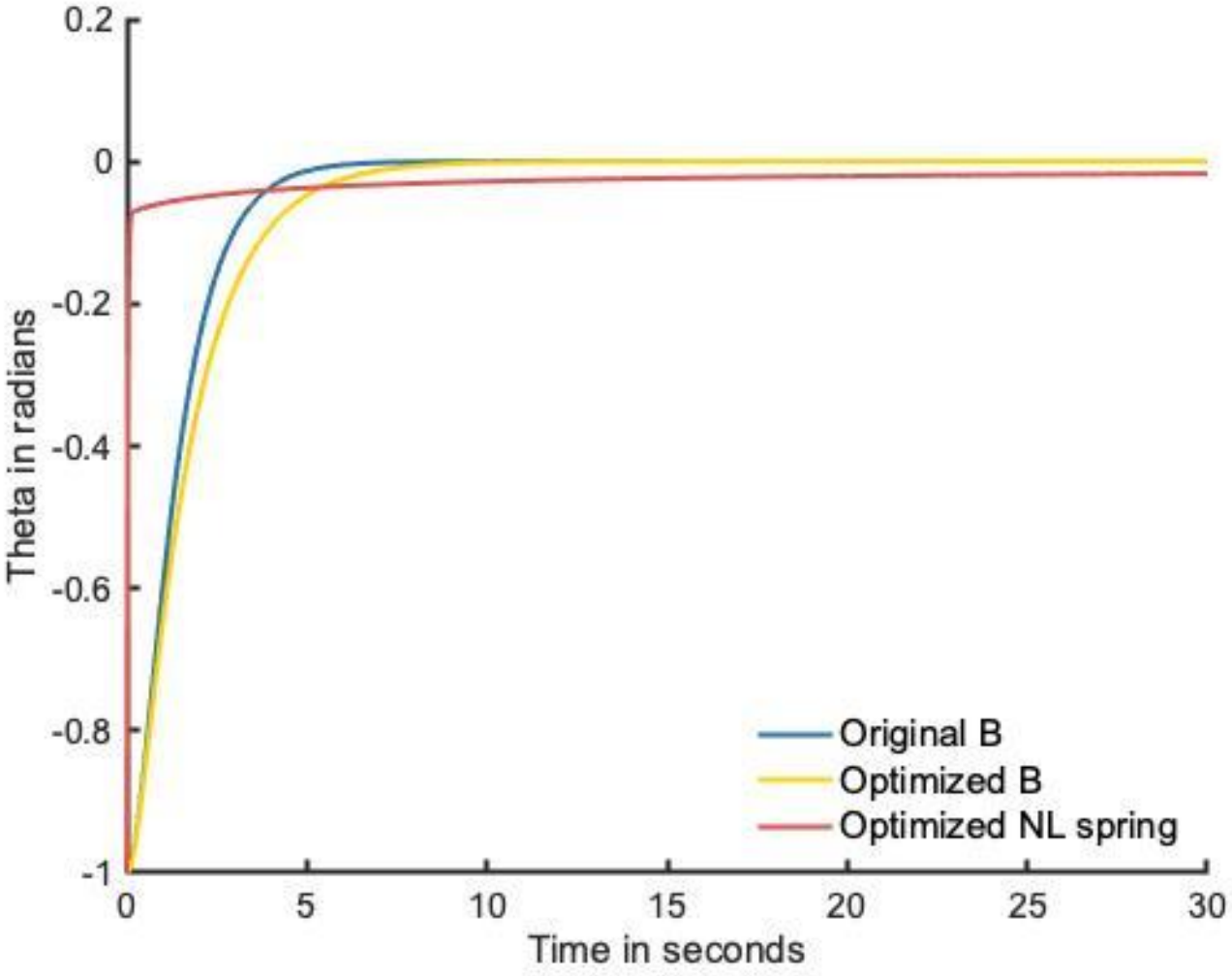
What is the RMSE for the optimized parameters? How does this compare to the system's performance when we only optimized b?

RMSE = 0.0321 rad for the optimized parameters. The RMSE is significantly lower than the two values calculated before indicating that it is a more accurate method.

What are the optimized values of b and α?

A = 16 Nm/rad b = 4.8 Nms/rad

To wrap things up, we're going to see how our optimized systems fare in the homogeneous case ($T = 0$). Remember, each of these systems was optimized for a particular external torque; by simulating the system response to a different input, we gain a sense for how specific our optimization was to the particular torque trajectory used for optimization. Re-run your numerical simulations using the optimized parameter values from 3c and 3d, but with $T = 0$ as your input torque (rather than $T = 30te^{-10t}$). Plot the results on top of each other, using the same formatting that you used for Figure 6. **Drop the resultant figure here.**



How well do the optimized systems seem to perform when no external torque is applied? What does this tell us about system robustness?

The systems perform better overall without torque as evidenced by the lack of a super high peak. The original b value seems to have the best performance, it converges to 0 faster than its optimized. Although the optimized NL seems to converge the fastest, it does not reach 0 like the other two in the given timeframe. The original b is a good balance of convergence and reaching 0. In terms of robustness, the original and optimized b value are pretty good. The optimized NL spring does not have good robustness due to its failure of reaching 0.