QUIZ 5 – MATLAB

Directions:

1. This is an open book quiz, meaning that you can use ***any*** sources of information you wish.
2. "Teamwork"/discussions are permitted during this take home quiz; however, cheating will not be accepted, and each submission will be run through plagiarism detection methods.
3. The submission of identical solutions will result in a mark of zero for any involved parties.
4. Only your owl submission will be marked.

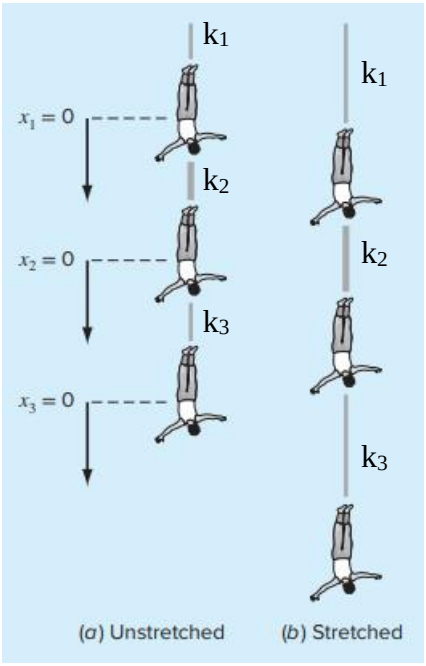
Submission Directions:

5. Code all your solutions in MATLAB. Save your solution as an '.m' file.
6. Submit all your '.m' files on owl. You will not get any marks if a '.m' file is missing. Ensure you include all '.m' files that are necessary to run your script/function.
7. Ensure your upload to OWL is successful and that all files are functional. After the deadline has passed missing or corrupt files will not be accommodated for.

Note: While performing problems you can use the help function built into MATLAB to help you with syntax, and the proper use of commands.

Problem 1 [30 marks]

Three bungee jumpers are connected in series. In **a)** they are held in place with the cords unstretched. In **b)** they have been released and are at equilibrium (no motion) causing the bungee cords to be stretched by the weight of the jumpers.



Assuming each bungee cord can be modelled as a linear spring the following equations are true:

$$(k_1 + k_2)x_1 - k_2x_2 = m_1g$$
$$-k_2x_1 + (k_2 + k_3)x_2 - k_3x_3 = m_2g$$
$$-k_3x_2 + k_3x_3 = m_3g$$

Jumper	Mass (kg)	Spring Constant (N/m)	Unstretched Cord Length (m)
Top (1)	60	50	20
Middle (2)	70	100	20
Bottom (3)	80	50	20

It is also known that the bungee cords experience a change in stiffness with respect to the ambient temperature. To account for this, a correction factor is applied to the spring constant of each bungee cord, where $k_{true} = k_{given} * [correctionFactor]$. A series of correction factors and their corresponding temperatures are provided in array format in the provided file `givenData.m`.

a) We first want to work with the correction factor data. Using the data provided in `givenData.m` fit a polynomial function to the given correction factor vs. temperature data. This could be done using the built in `polyfit` function. You may choose any order of polynomial you think is appropriate. **[4 marks]**

b) We now want to analyze the temperature-correction factor polynomial curve fit. Using the provided `goldmin.m` function find the smallest correction factor described by the polynomial over the temperature range -20 ≤ T ≤ 50. You may select appropriate values for the inputs to the function. **[4 marks]**

Note: you will need to define your polynomial fit as an anonymous function. Using the `polyfit` function, you can store your resulting polynomial coefficients in a variable, and then expand those coefficients into the resulting polynomial in the anonymous function definition. For example, if you used a 3rd order polynomial fit and stored the coefficients in the variable 'p', the corresponding anonymous function call would be:

```
f =@(x) p(1)*x.^3+p(2)*x.^2+p(3)*x+p(4);
```

c) Plot your resulting polynomial fit and mark on the plot the location of the smallest correction factor using an appropriate symbol (a circle would work well). **[2 marks]**

d) Now, we want to solve for the position {x} of each individual jumper once the bungee is stretched for three temperatures; i) the temperature where the lowest correction factor is found, ii) -10ºC, and iii) 40 ºC. Be sure to report the location of each bungee jumper from the origin of the first bungee cord. **[15 marks]**

Note: you will need to multiply your [K] matrix by the corresponding correction factor for each temperature. This could be done in a loop for each desired temperature, or simply could be done three times. Then, {x} can be solved for each case using any method of your choice.

Note: Also, remember that the `polyval` function can be used to get the value of a function for varying T values. Use the help `polyval` command in the command window for syntax definitions if needed.

Note: In order to present the results with respect to the origin of the first bungee cord, the position of each jumper in the unstretched state (as shown above in figure a) needs to be added to the {x} values calculated, since they represent only the change in length at equilibrium. These initial positions can be determined using the positions in figure a and the unstretched lengths of each cord provided in the table above.

e) Finally, have your script output an error message reporting the temperature if the position of the last bungee jumper (#3) exceeds 100 m from the origin. **[5 marks]**

Hint: If you have done everything correctly, one of the temperatures you will check should result in this error message!