Provide your results, discussion, and Matlab code for the following questions in a single PDF.

1. In this problem, you will design an MPC controller for the active suspension system presented in class. The goal is to use this application example as an opportunity to design a well-functioning MPC controller with limited guidance on the specific approach, controller design, or tuning. Unlike previous homework assignments there is not a single "correct" answer. Instead, use what you have learned throughout the class so far and your engineering intuition to guide your controller design. The objective is to be mindful and systematic in the control design so that you can justify your design choices. The entire control design process is broken down using the following steps and code is provided on eLearning to simplify the closed-loop simulation plotting/visualization.

   When designing your MPC controller, the control objective is to satisfy the actuator and state constraints, if possible, while minimizing the acceleration of the unsprung mass (for ride comfort) and minimizing the use of the actuator (since applying force in the active suspension consumes energy). The exact trade-off between these objectives is up to you. To help design your controller, several road profiles (that define the disturbances) are provided on eLearning. Several of these are very simplified profiles that can be used in preliminary testing and controller development. One profile represents a more realistic road with a more random profile. While you can use these profiles however you like to help in with the controller design, your final assignment should at least include closed-loop MPC results for the `roadBumpHole` and `IRI_737b` road profiles.

   (a) In the provided Matlab code from eLearning, read through and run the first three sections of code that 1) define the model parameters and continuous-time state-space model, 2) define the state, input, and disturbance constraints, and 3) conduct an open-loop simulation without any force from the actuator in the active suspension. Note that this third section is where you can select from several choices of the road profiles provided on eLearning. Plot the open-loop simulation data (and run the animation, if you like). Observe the open-loop behavior of the system for different road profiles to develop some intuition about how the system behaves. You will use these open-loop simulations to learn about the dynamic behavior of the system to guide the design of your MPC controller.

   (b) Many of our MPC controller formulations have used an LQR-type cost function with quadratic penalties on the states and inputs. The next section in the provided code, provides an opportunity to design an LQR controller for the system to determine how the Q and R matrices can be used to affect the closed-loop system behavior. In this optional step, you can design/tune your Q and R matrices using LQR to use as a starting point for the cost function for your MPC controller. Typically, LQR simulations are much faster than MPC simulations (due to the computation time of the MPC controller) and thus you can often save time by using LQR to identify appropriate Q and R matrices.

   (c) Since we assume the rate of change of the road height is an unknown bounded disturbance, it makes sense to think about a robust MPC formulation similar to the ones from Homework #3. The next two sections in the code are also optional and can be used to analyze the effects of these bounded disturbances on the system. You can use forward reachability analysis (similar to the first approach in Homework #3) and/or compute a robust positive invariant set (similar to the second approach in Homework #3) to analyze the effects of these disturbances. Based on your results you could decide whether a formal robust MPC formulation is applicable and if state and input constraint satisfaction could be guaranteed given the bounds on the disturbance.

(d) The next section in the code is for your MPC control design. Two of the controller design parameters are the discrete update rate of the MPC controller and the prediction horizon length. Note that the discrete update rate of the controller should be an integer multiple of the simulation time step. In addition to these two parameters, you now have complete control over which optimization variables you use and how you define the objective function and constraints for the controller. The main part of this homework assignment is to design this MPC controller and provide explanations for your design choices.

(e) The last section of the code simulates the closed-loop system under MPC control and plots the results. Feel free to modify this section (and any of the previous sections) however you like. Specifically, you will at least need to complete/update some of this code based on the controller inputs and outputs that you chose in the previous section. Note that the code is written to create an variable called `d_MPC` which can be used as an input to your controller to provide different levels of information about disturbance. If `dFlag` is set to 0, then there is no knowledge about the disturbance and `d_MPC` is set to all zeros. If `dFlag` is set to 1, then it is assumed that the current disturbance is perfectly measured and assumed constant over the prediction horizon. If `dFlag` is set to 2, then it is assumed that the exact disturbance profile is known for the entire prediction horizon. You can use these three cases to develop and test your controller under different assumptions about the disturbance.

In summary, this assignment is intended to give you a large amount of flexibility in how you approach the MPC control design for the active suspension system similar to the flexibility you have in your class project and may have as a control engineer in the future. Remember that this assignment is less about getting the "correct" answer and more about your thought process and rational throughout the control design. Therefore, your report should focus on 1) presenting your design decisions, 2) explaining/justifying these design decisions, 3) using graphs/figures to support these explanations, and 4) demonstrating your controller's performance on the for the `roadBumpHole` and `IRI_737b` road profiles. These controller performance demonstration should at least include the cases where no knowledge of the disturbance is assumed (`dFlag` = 0) and where perfect preview of the disturbance is assumed (`dFlag` = 2). Use these results to briefly discuss the benefits of perfect disturbance preview.