Problem Set 2 for ABE 598 Autonomous Decision Making in the Real World

Name: Jingwei Zhu

UIN: 675271843

Problem 1:

1. *Collect a dataset: simulate the wingrock dynamics system using wingrock_main with different initial conditions and for 5 seconds runtime in each condition.*

Four initial conditions are tested:

#1 (1.2;1), #2 (3;6), #3 (2;0), #4 (0;4)

2. *Write code to learn the weights of the wingrock model for $\dot{p}$ assuming they are unknown. Use the basis from the parameterized model provided. Compare learned weights with the original weights.*

$$\dot{\phi} = \mathrm{p}$$

$$\dot{\mathrm{p}} = 0.8 + 0.2314\phi + 0.6918\mathrm{p} - 0.6245|\phi|\mathrm{p} + 0.0095|\mathrm{p}|\mathrm{p} + 0.0214\phi^3$$

Answer:

In the matrix form,

$$\dot{\mathrm{p}} = [1, \phi, \mathrm{p}, |\phi|\mathrm{p}, |\mathrm{p}|\mathrm{p}, \phi^3] \cdot [\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6]^{\mathrm{T}}$$

Let $\mathrm{B} = [\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6]^{\mathrm{T}}$, which is the parameter vector we are trying to solve.

Put all the data collected in the five seconds together:

$$\mathrm{PDOT} = \begin{bmatrix} 1, \phi_1, \mathrm{p}_1, |\phi_1|\mathrm{p}_1, |\mathrm{p}_1|\mathrm{p}_1, \phi_1^3 \\ \vdots \\ 1, \phi_\mathrm{n}, \mathrm{p}_\mathrm{n}, |\phi_\mathrm{n}|\mathrm{p}_\mathrm{n}, |\mathrm{p}_\mathrm{n}|\mathrm{p}_\mathrm{n}, \phi_\mathrm{n}^3 \end{bmatrix} \cdot [\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6]^{\mathrm{T}}$$

where $\mathrm{PDOT} = [\dot{\mathrm{p}}_1, \dots, \dot{\mathrm{p}}_\mathrm{n}]^{\mathrm{T}}$

Using linear regression to solve for B:

	Data under initial conditions #1	Data under initial conditions #1,2	Data under initial conditions #1,2,3	Data under initial conditions #1,2,3,4	Original weights
β_1	1.1632	0.6730	0.6983	0.7771	0.8
β_2	0.3227	0.2888	0.2849	0.2610	0.2314
β_3	0.6098	0.6742	0.6821	0.7147	0.6918
β_4	-0.7709	-0.6616	-0.6646	-0.6666	-0.6245
β_5	-0.0703	0.0099	0.0099	0.0100	0.0095
β_6	0.0277	0.0226	0.0227	0.0227	0.0214

Using the data collected with four initial conditions, the parameter vector obtained by linear regression is close to the original weight vector.

3. *GP regression: Now we will learn a data driven GP model with RBF bases using Csato's online GP algorithm. Run the regression: use the onlineGP class to learn a GP representation of wingrock dynamics using the data collected in 1. The output (dependent variables) of the GP should be the second state derivative. The input to the GP (the independent variables) should be the control input and the entire state vector.*

The code and results are attached.

4. *Validate: create a different dataset on which learning was not performed, and compare the predictions of your learned model with that dataset. Compare in terms of the following metrics:*

a. Mean squared error between your predictions and the true values

b. What is the mean squared error for the initial conditions you had in the training data set, what is it for those that were far from those in the training data set (be careful, the wing rock system can be unstable for some initial conditions)

GP parameters:

bandwidth = 30; noise = 1; tol = 0.00001; max_points = 30
Figure 1 shows the comparison of GP predicted $\dot{p}$ and the actual $\dot{p}$ for 60 s for initial condition (1.2;1), which is one of the initial conditions used to train the model. The MSE is 4.2. It can be observed that the predicted $\dot{p}$ follows closely the trend of actual $\dot{p}$.

Figure 2 displays the comparison of GP predicted $\dot{p}$ and the actual $\dot{p}$ for 60 s for initial condition (6;3). This is not one of the four initial conditions for model training. The MSE is 5.5. At the beginning due to the fact that the initial state is not within the range of training datasets, the model prediction significantly deviates from the actual value.

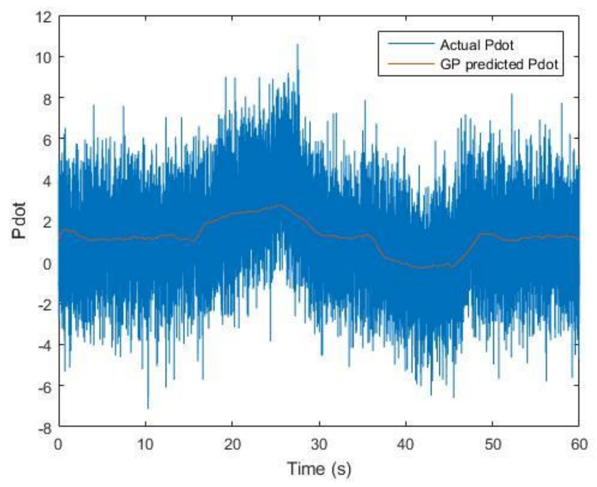


Figure 1. Comparison of Pdot prediction by GP regression with actual Pdot value for initial condition (1.2;1) which is one of the four initial conditions used to train the GP model.

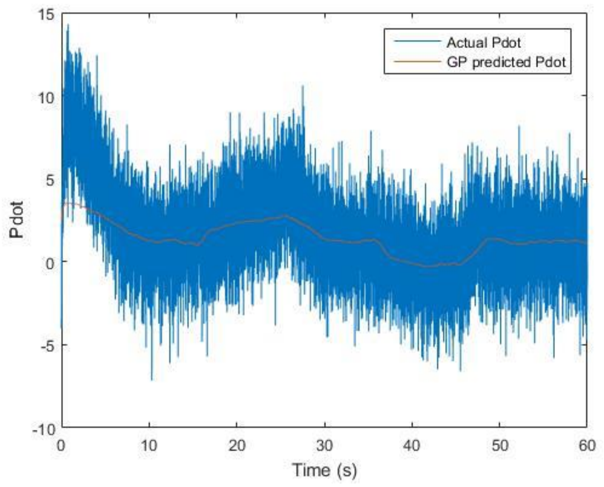


Figure 2. Comparison of Pdot prediction by GP regression with actual Pdot value for initial condition (6;3) which is not one of the four initial conditions used to train the GP model.

Problem 2: Clustering with K-means

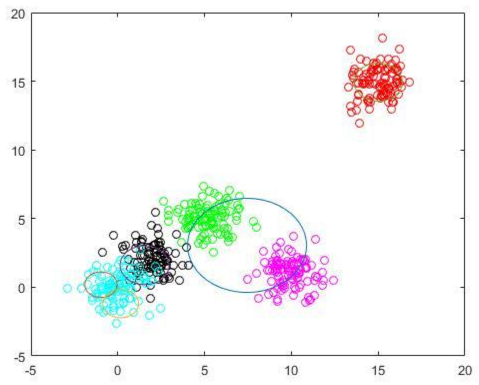
The file gauss_mix_data.mat contains data from 5 Gaussian distributions. Your goal is to implement the K-means algorithm to identify the 5 clusters. The file EM_Gaussian_mixture_model_setup.m shows how you can extract data from the gauss_mix_data.mat file and plots it in different clusters.

How long does your algorithm take to run?

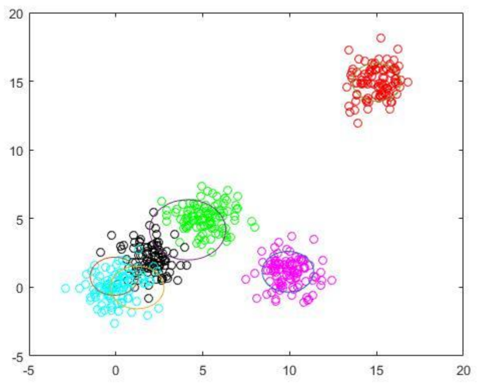
Does it work every time? Do at least 20 runs of your algorithm and report a histogram of showing the number of times your algorithm found the right clusters.

Answer:

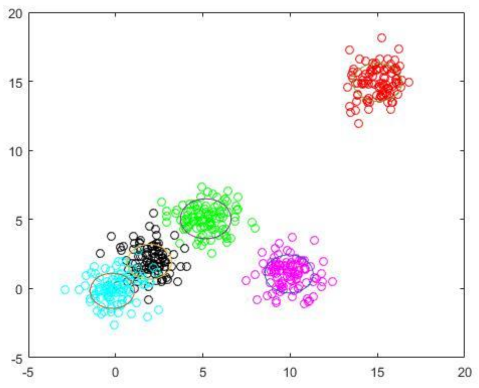
K-means clustering with initial prior mean X=[1.833885015, 0.86217332, -1.307688296, 0.342624467, 2.76943703], Y=[0.53766714, -2.258846861, 0.31876524, -0.433592022, 3.57839694] after 5, 10, 15, 20, 30, 100 iterations is shown as follows:



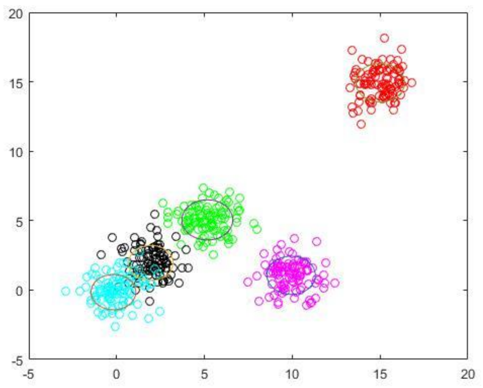
(a)



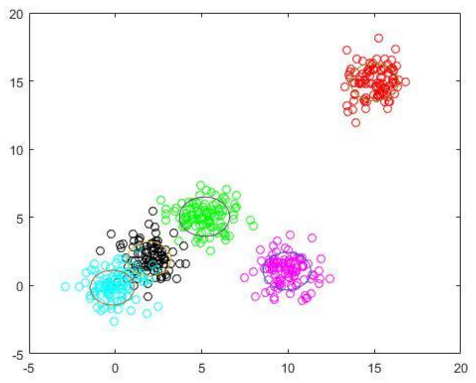
(b)



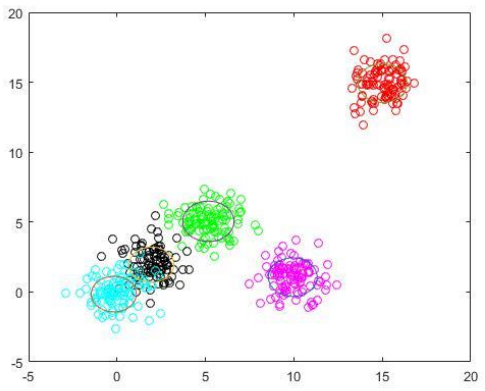
(c)



(d)



(e)



(f)

Figure 3. K-means clustering after (a) 5 iterations (b) 10 iterations (c) 15 iterations (d) 20 iterations (e) 30 iterations (f) 100 iterations.

It can be observed that the clustering converged after around 15 iterations. Right clusters have been found by the algorithm.

The algorithm found the right clusters every time. The algorithm was tested 30 times and it all found the right clusters when the prior means are initialized by
prior_mean=randn(2,5);
Even when the prior means are initialized by
prior_mean=randn(2,5)*1000;
the algorithm was tested 20 times and it always found the right clusters.

Problem 3: Neural Networks

Using the dataset from Problem 1 we will create a single hidden layer neural network for wingrock dynamics. Choose the number of hidden layers to be 10, use the sigmoidal activation function. Perform the following:

1. Write a script to program the neural network. Initialize the weight randomly

2. Implement backpropagation algorithm

3. Train on the same data set you are using for training in problem 1 and validate the output of the model using the same dataset you are using for validation on problem 1

4. Compare the performance of GP and SHN NN. How does changing the number of hidden layers affect the performance (reducing and increasing with increments of 2 from 4 to 16)?

5. Does the convergence depend on initial weights for the default case of 10 hidden layers?

Answer:

The shallow neural network initially has 10 hidden neurons.

After implementing the backpropagation algorithm to train the neural network (10000 iterations, learning rate = 0.2), the MSE (mean squared error) of the neural network prediction has been reduced to 3.95, as shown in Figure 4. A few spikes of MSE were observed during the iterations. They might be due to the fact that the learning rate is relatively large.

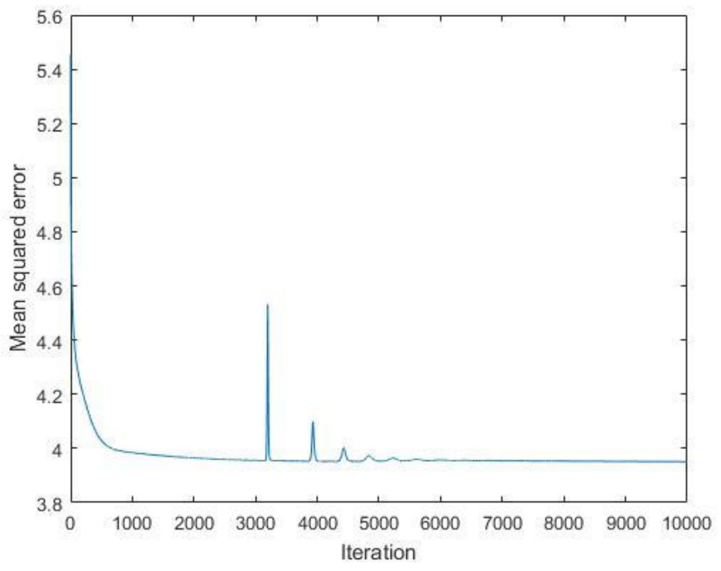


Figure 4. MSE of a single hidden layer neural network for wingrock dynamics.

The comparison of the predicted $\dot{p}$ with the actual $\dot{p}$ from the four initial conditions in the first 5 seconds is shown in Figure 5. The predicted value is following the trend of the actual value closely. The same network has been also implemented with TensorFlow with same learning rate (0.2) and iteration number (10000). Similar results have been obtained, as shown in Figure 6. The MSE after 10000 iterations was 4.17. The gradient descent optimizer in my algorithm is better since its computation of gradient is accurate while in TensorFlow the gradient is estimated. MSE of SHLNN is close to that of GP model.

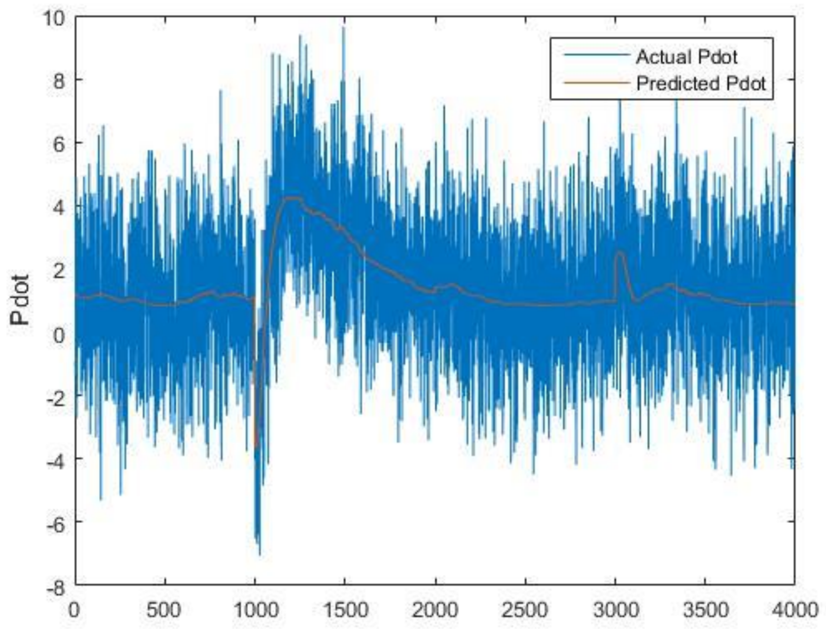


Figure 5. Comparison of Pdot prediction by SHLNN with the actual Pdot value for the four initial conditions used to train the model.

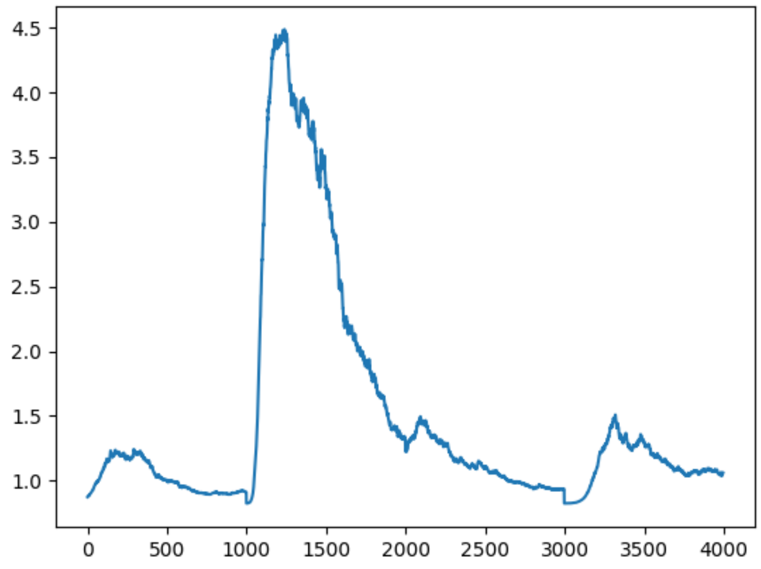


Figure 6. Pdot prediction by SHLNN (using TensorFlow) for the four initial conditions used to train the model.

MSE of network with 4,6,8,10,12,14,16 hidden neurons with learning rate of 0.2 and 1000 iterations:

Number of hidden neurons	4	6	8	10	12	14	16
MSE	4.04	4.08	4.07	3.98	3.99	3.99	3.99

MSE stopped reducing after the hidden neuron number reached 10. Generally speaking, the more hidden neurons there are, the smaller MSE can be achieved. However, larger number of hidden neurons may also result in over-fitting.

In Figure 7 after removing the noise, the actual $\dot{p}$ from initial condition (1.2;1) is again compared with the predicted $\dot{p}$ for the whole 60 s. It can be observed that for most of the time, the predicted value agrees well with the actual one. However, between 35 s and 50 s, the two have large disagreement. This is probably because the features of the inputs in that period are not well represented by the dataset from four initial conditions in 5 s.

Similarly in Figure 8, the actual $\dot{p}$ after removing the noise from initial condition (6;3) is compared with the predicted $\dot{p}$ for the whole 60 s. Big deviation is observed at the beginning due to the fact that the initial state is not within the range of training datasets.

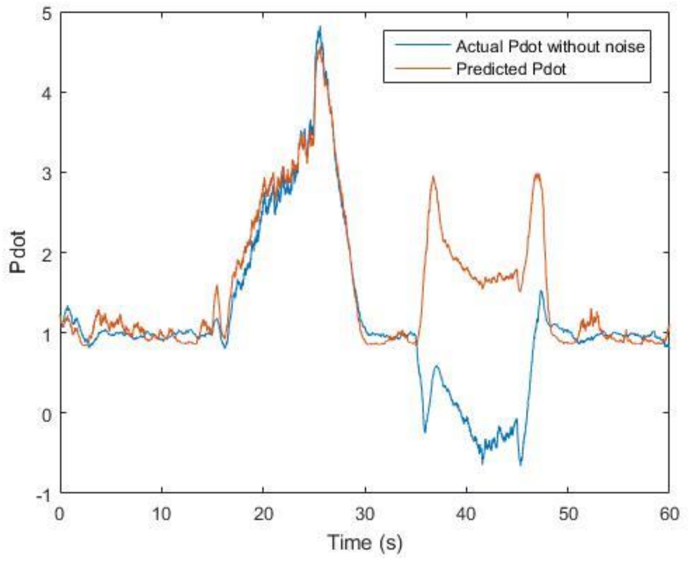


Figure 7. Comparison of SHLNN predicted Pdot with the actual value for initial conditions (1.2;1) (in the dataset used to train the network) for 60 s.

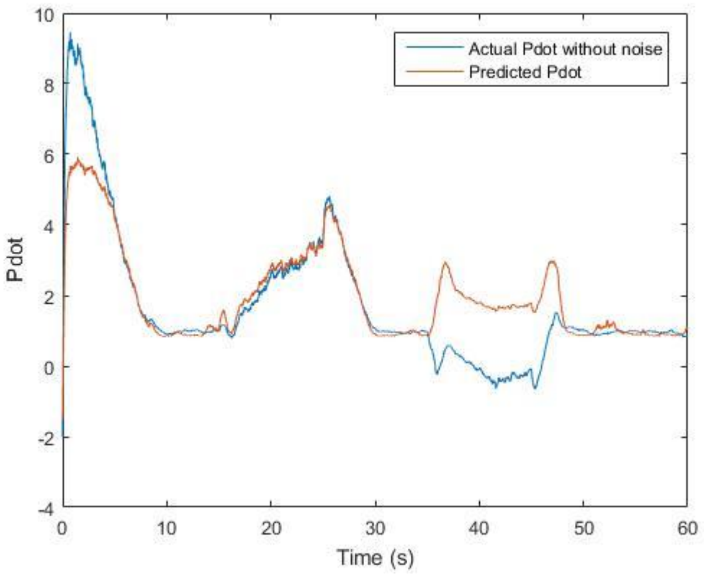


Figure 8. Comparison of SHLNN predicted Pdot with the actual value for initial conditions (6;3) (not in the dataset used to train the network) for 60 s.

The initial weights do not seem to affect the convergence of SHLNN in the current case.

Bonus:

Use Tensor Flow or your favorite DNN package to classify MNIST dataset as described here: https://www.tensorflow.org/tutorials/deep_cnn

A convolutional neural network has been implemented to classify MNIST dataset. After 2000 iterations with 50 samples in each batch of data, the training accuracy is 0.976.