Homework 1

A051010 林奕汝

In this homework, I use Python to implement these three approach. The first one is most-likelihood approach. It only needs to minimize the mean-square-error. Second approach is Ridge Regression avoiding the model generated is overfitting. Last approach is Bayesian, calculating the result from S inverse.

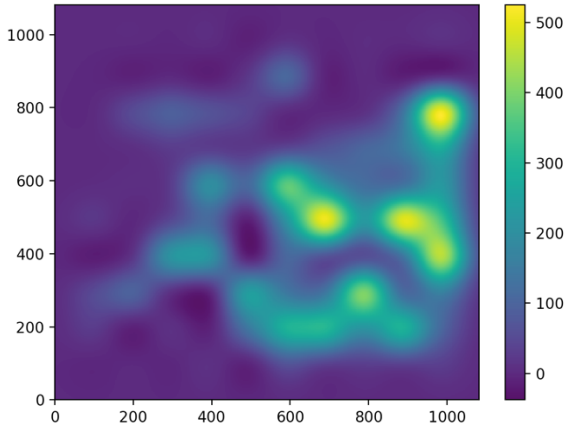
- **Design**

Because we need to tune the hyperparameters, the main.py includes the parameters settings. After updating settings from argv, it puts the model setting in approach.py, and split the shuffled known data into train/test case according to parameter “data_size”. Finally, it chooses the approach method to get ML/MAP weights.

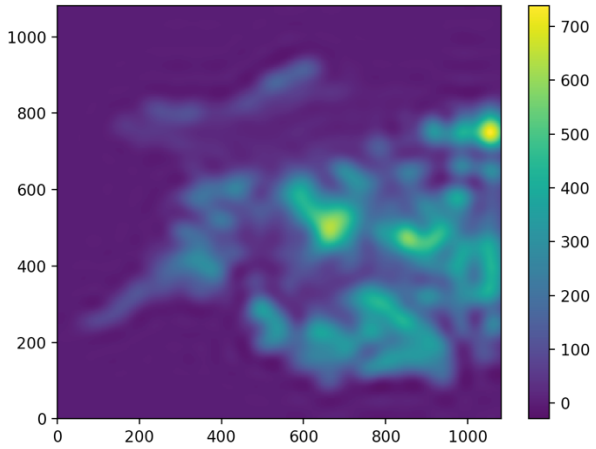
Both of ML and MAP take Stochastic (sequential) gradient descent to approach weights, it uses parameter “batch_size” to decide how many items to use at once. Going through all the batches, it counts as one iteration. Such iterations run for “iter” times. After getting the weights result, I put the test case into the model to get the MSE of it.

- **Difference and Performance**

Case_1: Change the basis_size and sigma’s ratio (100:50 to 1024:30) on ML

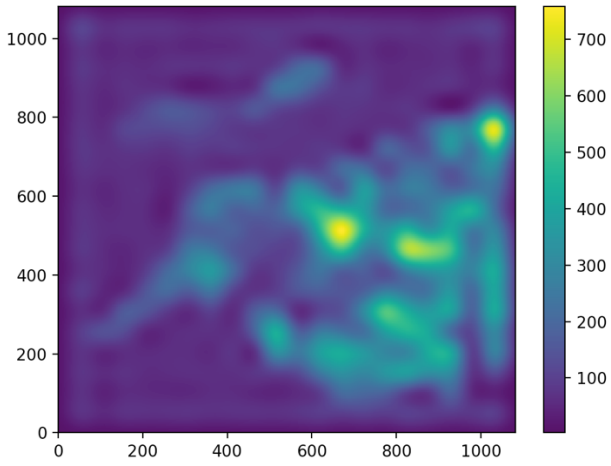


100:50 ML’s MSE= 2094.9319

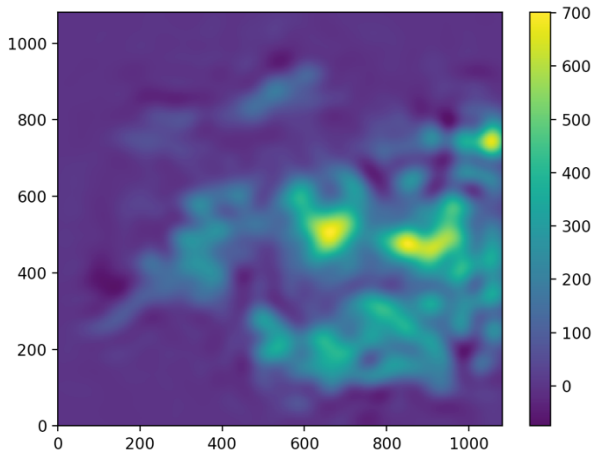


1024:30 Bayesian's MSE = 248.0606

```
settings = {"map_size":1081,"dim":2,"data_size":10000,
    "batch_size":16,"iter":10,"eta":0.6,"k_folder":0,
    "basis_size":400,"sigma":30, "lamb":0.005,
    "m0":0,"alpha":2,"beta":25,"graph_scale":2,
    "x_train":"data/X_train.csv","t_train":"data/T_train.csv"}
```



lambda = 0.005 , MAP's MSE = 8366.3847



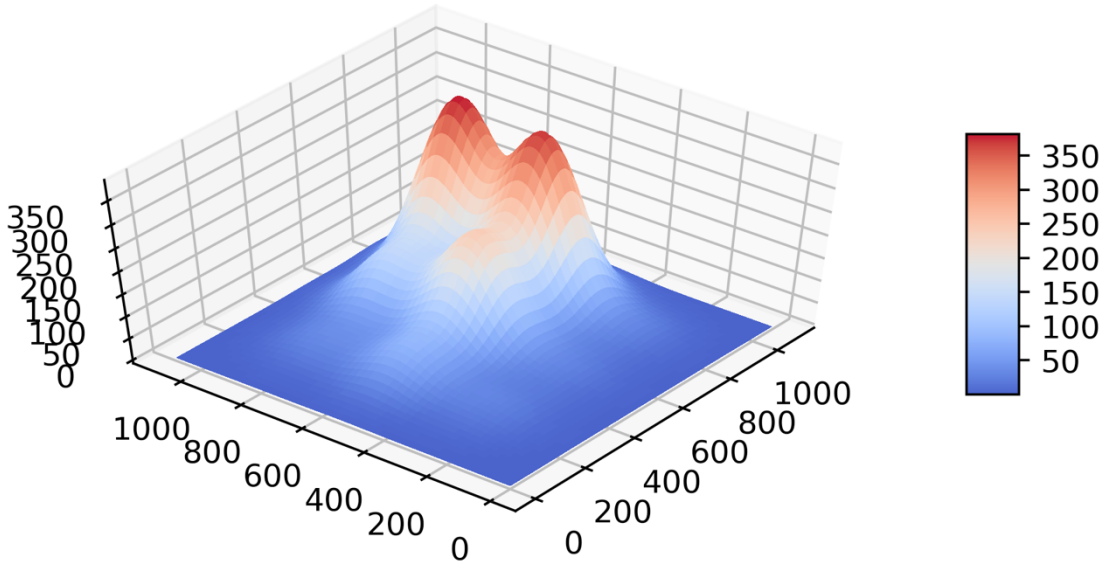
Bayesian's MSE = 4223.0587

Performance

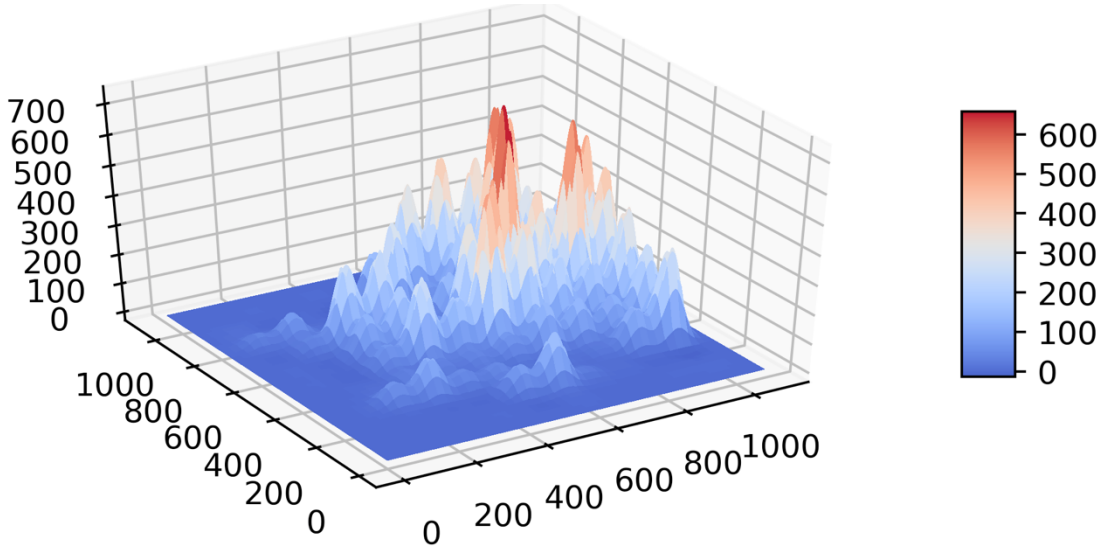
Approach	ML	MAP	Bayesian
MSE	907.943	1004.4955	248.0606

- **Underfitting and Overfitting**

Using Bayesian approach to get linear regression, the fitting could be number of basis function. Too many basis functions may cause overfitting, but too less would be underfitting. For example,



Underfitting(10 basis functions) Bayesian's MSE = 87980.2947121



Overfitting (1024 basis functions) Bayesian's MSE = 4223.0587

- **Cross validation for Hyperparameters Selecting**

  There're a few hyperparameters for different use:
  - MAP approach: λ (avoid for overfitting)
  - Bayesian approach: α, β, m_0
  - Gaussian basis function: number of basis function, μ (mean value), σ (variance)
  - Stochastic (sequential) gradient descent:
    η (eta: learning rate in each gradient), number of iterations, batch size

Here, I choose σ (variance) in basis function to do the analysis. Assume there's 400 basis functions in this model. With 5-folder's cross validation (2000 for train/test in each time) and Bayesian approach, the average mean-square-errors are shown below:

python3 main.py --approach Test --k_folder 5 --sigma *n* --data_size 10000

Sigma	20	21	22	23	24	25
MSE	1917.7065	1829.1147	1739.9916	1539.3145	1382.8066	1409.2965

Sigma	26	**27**	28	29	24	25
MSE	1374.7555	**1204.3463**	1324.2857	1539.3145	1382.8066	1409.2965

Reference:
http://aimotion.blogspot.tw/2011/10/machine-learning-with-python-linear.html
https://www.tutorialspoint.com/python/python_command_line_arguments.htm