Machine Learning

HW-1: Linear Regression

李有權 0210189

Development Environment

Language: Python2.7

CPU: Intel® Core™ i5-3320M CPU @ 2.60GHz

ML MAP Bayesian

Maximum Likelihood

To obtain the maximum likehood of the data given, we can obtain a closed-form solution from the dataset given to us.
The closed form solution is obtained by solving minimum value of:

$$\ln(\mathcal{L}) = -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_i (y_i - w^T x_i)^2$$

Which is obtained by taking the derivative with respect to w and equate it to 0 to find the minimum where we end up with the following equation:

$$w = (X^T X)^{-1} X^T y$$

Maximum A-Posteriori

MAP is done by getting the posterior of the data by multiplying the likelihood function with a prior. A prior is a belief on where the data will bound towards before training or applying the maximum likelihood function.
The closed form solution of MAP is obtained by solving for the minimum value of:

$$\sum_i (y_i - w^T x_i)^2 + \frac{\lambda}{2} w^T w$$

And by taking the derivative w.r.t. w and equate it to 0, we obtain:

$$w_{MAP} = (\lambda I + X^T X)^{-1} X^T y$$

Bayesian Treatment

From the text "Due to the choice of conjugate Gaussian prior distribution, the posterior will also be Gaussian". The main difference between Full Bayesian Treatment and MAP is that Full Bayesian gives information of all orders of the entire distribution [1]

It is also stated that in full bayesian, we assume a prior in σ^2 in addition to prior w [2]. Thus the result that we will obtain from full bayesian would be the same as MAP.

The procedure of full bayesian is the same as MAP and we will end up with the same equation:

$$w_{MAP} = (\lambda I + X^T X)^{-1} X^T y$$

Where w_{map} corresponds to m_N in the slides.

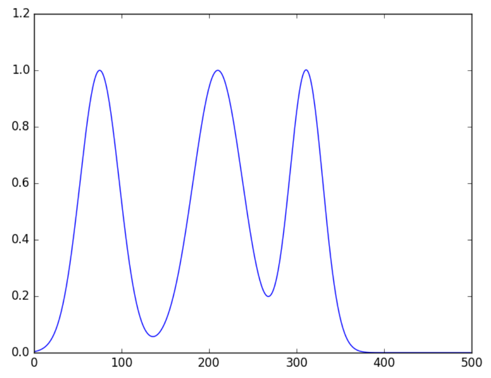
$$\begin{aligned}\mathbf{m}_N &= \mathbf{S}_N \left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}\right) \\ \mathbf{S}_N^{-1} &= \mathbf{S}_0^{-1} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}.\end{aligned}$$

Through full bayesian, we would obtain the following extra information:

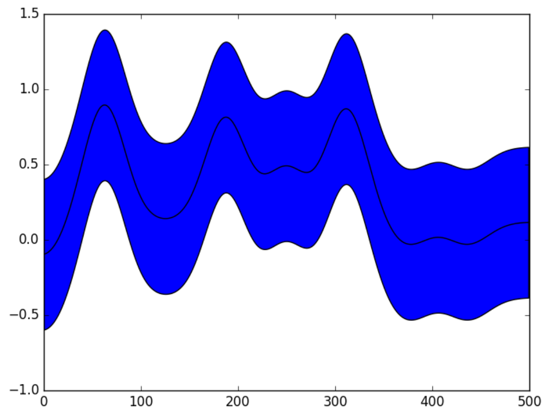
$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \phi(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\phi(\mathbf{x})$$

It can be seen that there's a new parameter β from the variance equation. From the text, we know that λ = α/β thus β is another unknown parameter that we'll have to figure out. As MAP and full bayesian shares the same result later, and it's hard for me to show the effect of the variance on a 3D plot, I'll show the effect of variance on my own 2D dataset below.

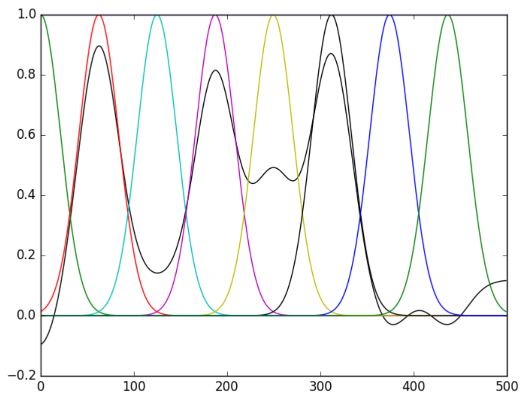
The figure below shows the ground truth of my data with is a joint gaussian distribution of 3 independent means.



The figure below is the estimated result from my trained model. Here, I assume β = 1 to keep things simple. The black line within is the estimated value by using the same method used for MAP. The purple area surrounding the actual value is the variance of computed from the equation above. It can be seen that the purple area is relatively small when the predicted model matches the ground truth and on the contrary, the area of the surrounding purple line is large when the predicted result doesn't fit the ground truth.

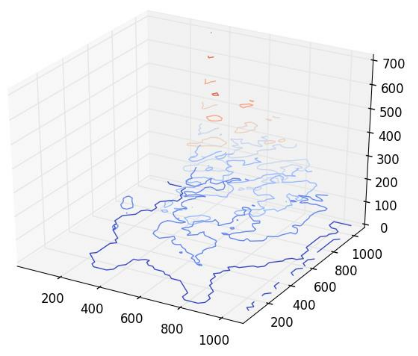


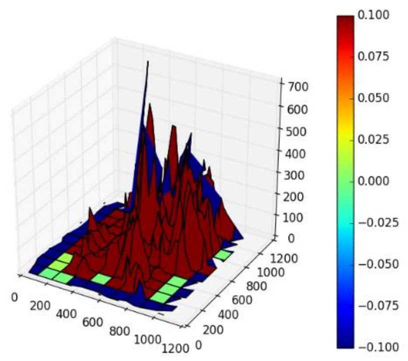
The diagram below shows the gaussian basis function I used and is cross validated for the model used for this example.



Basis Function

There are few basis function that we can consider for the design of our model. The two most commonly used basis function are polynomial and gaussian. I've experiment on both model in the design of by basis function and as expected, Gaussian basis function is better than polynomial basis function. The main reason is that the model that we are working with is to estimate the height map of the given 2D axis (x and y). The altitude of land is by nature close to gaussian distributions which closely models hills and terrains which are usually peaks. The ground truth dataset that we are about to work on is shown below:



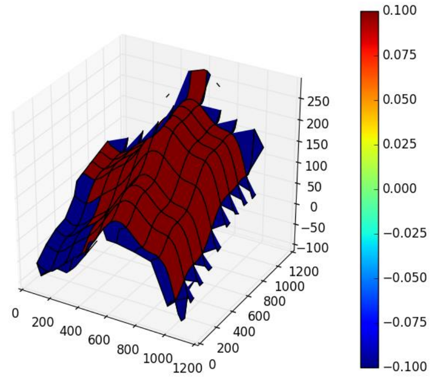


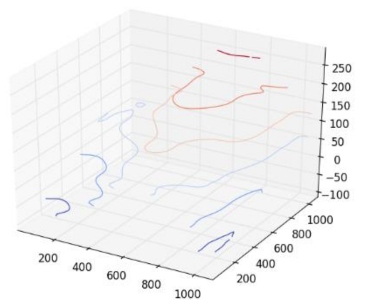
I'll base my discussion on the two basis function that I've experimented below.

Polynomial

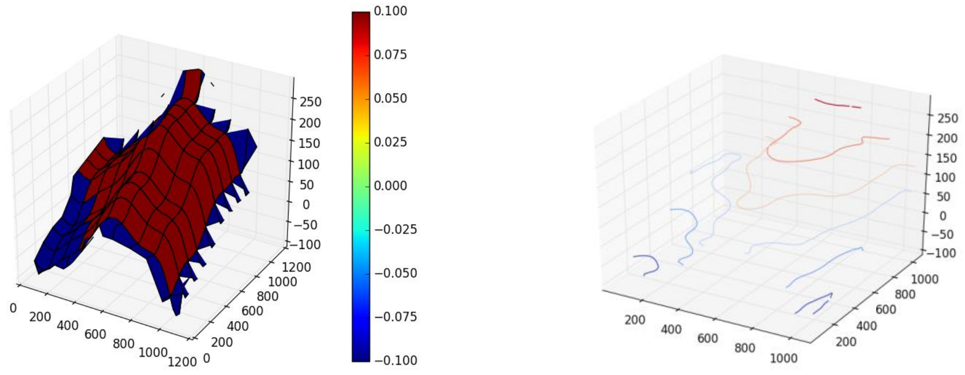
For the design of polynomial function, I consider X and Y independently in terms of order where terms like xy, x^2y, xy^2, etc. wouldn't appear (to keep things simple).

The result of using polynomial is shown below for ML and MAP method (Bayesian would be the same as MAP):





Maximum Likelihood



Maximum A Posteriori

I ended up with order 10 after cross validation and I obtained the MSE of:
ML: 3884
MAP: 3883 (lambda = 0.2)
It can be seen that MAP beats ML by just 1 in terms of MSE which implies that adding a prior does help us in our prediction.

Gaussian

There are several methods of applying Gaussian Basis function or Radial Basis Function (RBF) since there are 2 hyperparameter that can be tuned (μ and σ^2). First, we'll have to decide where to place the means of our basis function which is critical in the design of our **Design Matrix**.

The methods that are proposed by others are listed below:
1. Evenly spaced centers [3]
2. EM Algorithm [3]
3. K-Means Clustering [4]

Since we're not allowed to use any toolbox, I'll forgo method 2 and 3 since they require some time for me to complete and it will be part of our assignment in the future. Thus I'll go with method 1 which is evenly spaced centers. By "evenly spaced", we will need to find the gap between the center point of each Gaussian Kernel, so we'll set this as our hyperparameter to be obtain during **cross-**

validation. The next hyperparameter that comes along is σ^2 which will also be determined through our **cross-validation** process.

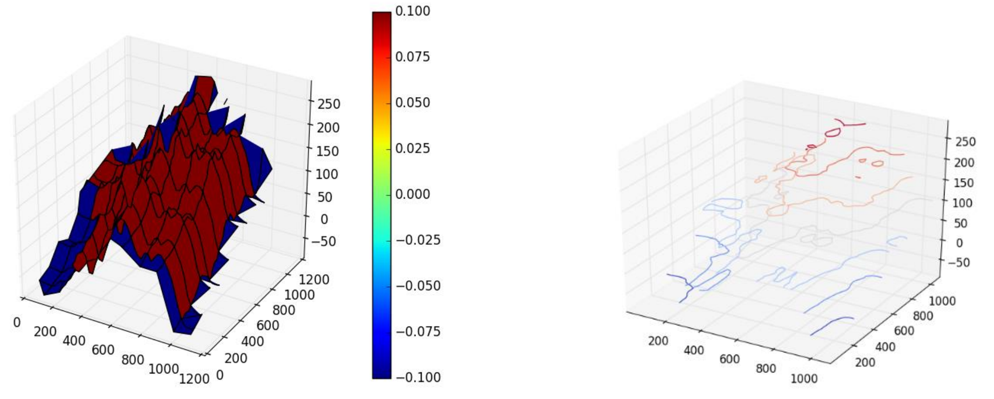
Instead of having the Gaussian basis function as in the slides:

$$\phi_n(x) = \exp\left(-\frac{(x_1-\mu_i)^2}{2s_{1n}^2} - \frac{(x_2-\mu_j)^2}{2s_{2n}^2}\right),\ for\ 1 \le i \le O_1, 1 \le j \le O_2$$

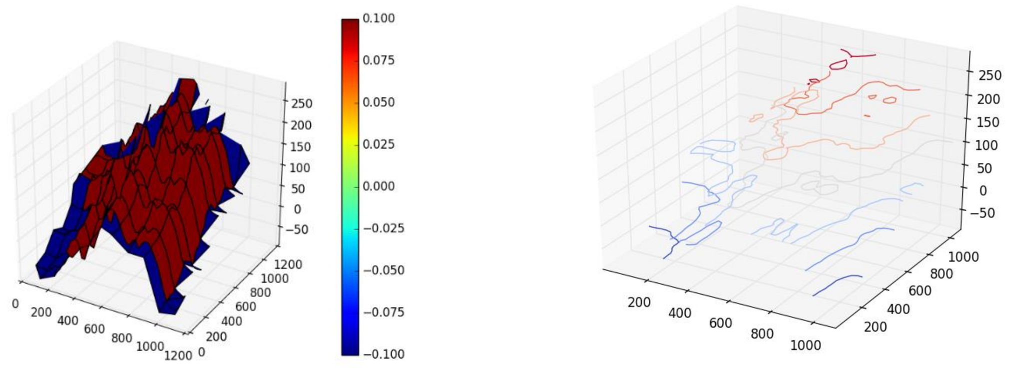
I consider x_1 and x_2 independently where each is represented as:

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

The results are shown below:



Maximum Likelihood



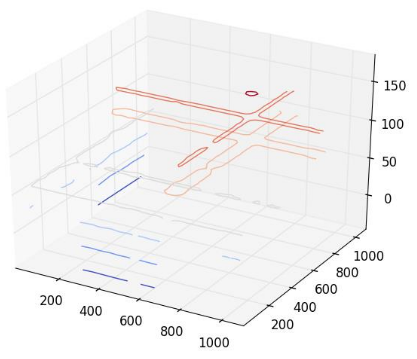
Maximum A Posteriori

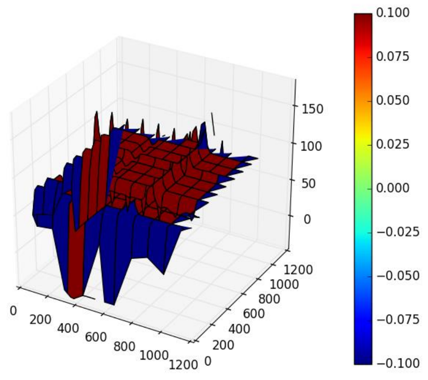
I ended up with 38 basis function for each dimension (x and y) and a variance of 250 after cross validation and I obtained MSE of:

ML: 3694
MAP: 3693 (lambda = 0.7)
It can be seen that MAP beats ML by just 1 again! We can also observe that using Gaussian as a basis funciton is superior compared to using polynomial.

Underfitting

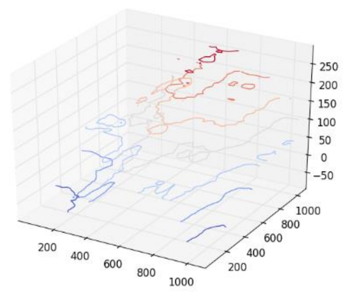
The example below shows a scenario of underfitting. I've used only 3 guassian basis function to create an underfitting case. It can be seen that the result that we got is far worse compared to the ground truth.

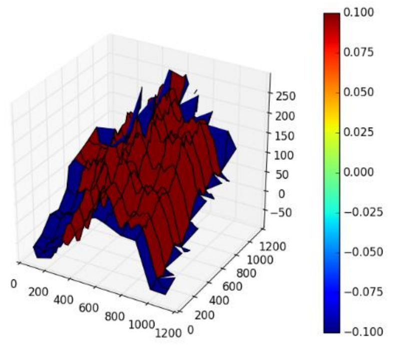




Overfitting

I've used 200 gaussian basis to create the effect of overfitting. It can be clearly seen in the second diagram where there're small "spikes" all over that is caused by overfitting.





Discussion

In all my work, I’ve segmented my training dataset into:

1. Training data (60%)
2. Cross Validation Data (40%)

The hyperparameter that has to be cross-validated are:

1. Number of basis functions
2. lambda (regularization parameter)
3. Variance for basis function (Gaussian)

References

[1]http://stats.stackexchange.com/questions/194033/relation-between-bayesian-estimation-and-maximum-a-posteriori-estimation
[2]https://www.quora.com/What-is-the-difference-between-linear-regression-using-MLE-Bayesian-linear-regression-and-linear-regression-using-MAP
[3]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.9552&rep=rep1&type=pdf
[4]http://stats.stackexchange.com/questions/117556/understanding-gaussian-basis-function-parameters-to-be-used-in-linear-regression
Lecture Notes