ROB313 Assignment 2

Alvin Pane [1004281118]

March 9, 2020

**Objectives**

The objective of this assignment is to study use cases of generalized linear models (GLM) in standard and kernel forms. The application of both forms will allow us to compare the results between the two. Implementing the GLMs will necessitate knowledge about basis functions and their construction. It will be important to understand how they can be constructed to create accurate predictions on regression and classification sets. This assignment uses the same datasets as in Assignment 1, meaning that we will have further opportunity to learn about the differences with GLM results as compared to the previous k-NN and linear regression model results.

**Code Structure**

The code was designed in a way to optimize the experience for the person running the code. The main strategy was to make the code modular. This was achieved by defining different functions, each responsible for handling a small task. Where applicable, general functions were written such that they could be called multiple times with different inputs. This was in the interest of space efficiency. The code uses print statements frequently to present data to the user in a clear manner. The main section calls functions depending on what question is to be answered. There are four variables: Q2, Q3, Q4, and Q5, all initialized to False. To run a question, simply set the variable equal to True.

- Q2 uses 2 functions, testGLM and validateGLM to formulate predictions on the test set of Mauna Loa, compute the test RMSE, and plot the predicitions against the actual values.
- Q3 uses only one function, kernelizedGLM, which constructs a kernelized form of the GLM from Q2 from a dual perspective.
- Q4 uses the functions Q4test and Q4valid. Q4 valid determines the optimal values for $\theta$ and $\lambda$. Q4test then computes the test root mean square error, and test ratio.
- Q5 uses 2 functions, orthogonalmatching, and helper. Helper computes a set of candidates, functions, and weights. Orthogonalmatching then computes the predictions and test error, given input of number of iterations and shape parameter

**Q1**

Derivation of closed form expression for the GLM weights using a least-squares loss and general Tikhonov regularization.

$$\arg\min_{w \in \mathbb{R}^m} \left( \sum_{i=1}^{N} \left( y^{(i)} - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(x^{(i)}) \right)^2 + \sum_{i=0}^{M} \sum_{j=0}^{M} \Gamma_{ij} w_i w_j \right.$$

In matrix form:

$$\operatorname*{argmin}_{w \in \mathbb{R}^m} \| \underline{y} - \underline{\underline{X}}\,\underline{w} \|_2^2 + \underline{w}^T \underline{\underline{\Gamma}}\,\underline{w} \,,$$

where $\underline{\underline{\Gamma}}$ is a symmetric positive semi-definite matrix

$$\Rightarrow \nabla_{\underline{w}} \left[ (\underline{y} - \underline{\underline{X}}\,\underline{w})^T (\underline{y} - \underline{\underline{X}}\,\underline{w}) + \underline{w}^T \underline{\underline{\Gamma}}\,\underline{w} \right] = 0$$

$$2\,\underline{\underline{X}}^T \underline{\underline{X}}\,\underline{w} - 2\,\underline{\underline{X}}^T \underline{y} + 2\,\underline{\underline{\Gamma}}\,\underline{w} = 0$$

$$\Rightarrow (\underline{\underline{X}}^T \underline{\underline{X}} + \underline{\underline{\Gamma}})\,\underline{w} = \underline{\underline{X}}^T \underline{y}$$

$$\boxed{\underline{w} = (\underline{\underline{X}}^T \underline{\underline{X}} + \underline{\underline{\Gamma}})^{-1} \underline{\underline{X}}^T \underline{y}} \quad \text{as desired.}$$

**Q2**

For the Mauna Loa dataset, a GLM was implemented to make predictions. The optimal regularization parameter, $\lambda$, was selected by iterating over [0, 20] and selecting the $\lambda$ which minimized the RMSE error. This was used to determine the predictions on the test set, and the corresponding test RMSE.

The basis function used to build the model is:

$$\phi(x) = [\ 1,\ x,\ x^2,\ x^3,\ x^4,\ x\sin(wx),\ x\cos(wx)\ ]$$

*w* was determined to be 111.15 through looking at the period of oscillation of the Mauna Loa training set. In this way, the basis function is designed such that it fits the trends of the dataset. The dataset appears as a linearly increasing sinusoid, with small trends that can be approximated with polynomial terms. Originally, the cubic and quartic terms were not included, but after including them it was found that the test RMSE could be significantly reduced.

The performance of the model is summarized below:

**Optimal Lambda:** 0
**Test RMSE:** 0.04180296665397129

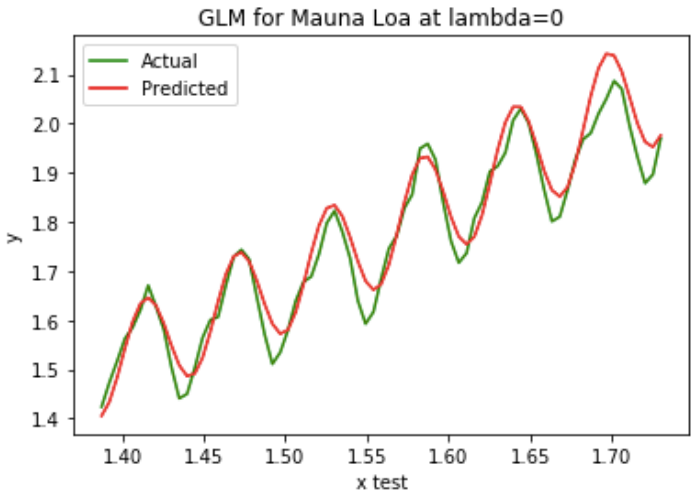


**Figure 1**: GLM Predictions, Mauna Loa Dataset

The optimal lambda was found to be 0, which means that it was favorable to disable regularization in this case. The designed basis function performed exceptionally well, with a test RMSE about 4.2%. The predictions are plotted against the actual values in Figure 1.

**Q3**

A kernelized form of the previous GLM was implemented from a dual perspective. The predictions on the Mauna Loa test set are shown in Figure 2. The results are summarized below. Since the question specified to choose a positive regularization parameter, $\lambda$ was chosen to be 1, rather than 0.

**Test RMSE:** 0.06086115811994485

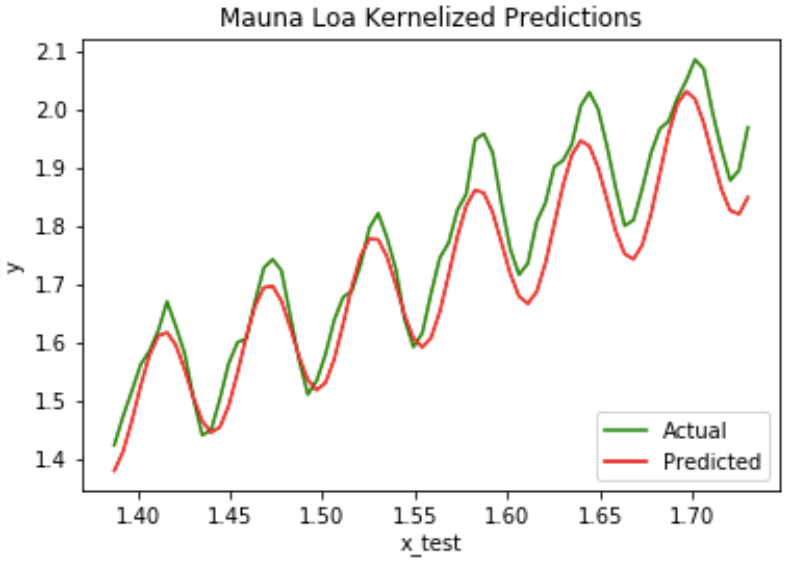


**Figure 2:** Mauna Loa Kernelized Predictions, $\lambda = 1$

We expected the kernelized GLM results to be identical to the results in Q2, however since we had to change $\lambda$, this was not the case. Regardless of the change, the predictions in Figure 2 are

still very close those in Figure 1, and the test RMSE is only off by about 2%. Thus, we can conclude that the kernelized version of the GLM gives results that are equivalent to those of the base GLM, just expressed in a different form.

The kernelized GLM performs worse in computational cost and memory requirement. Initializing the Gram Matrix iterates over two nested loops, each of length N, giving it $O(N^2)$ memory cost. The base GLM model is O(N*M), by virtue of storing the basis function for all training points in the phi-matrix. Computationally, the kernelized version is $O(N^3)$, while the standard GLM employs SVD which has computational cost $O(2N(M+1)^2 + 11N(M+1)^3)$. Here, N is much larger than M (the length of the dataset is much greater than 7), and so the computational cost is higher on the kernelized version. We can thus conclude that the standard GLM is more efficient overall for this use case.

Figure 3 and Figure 4 show plots of the kernel at x=0 and x=1, plotted over a range of z values, $z \in [-0.1,0.1]$. From these plots we can see that the kernel is not translationally invariant (stationary). Here we can conclude that the kernel depends on more than the difference between x and z because adding 1, a constant value, to both x and z produced drastically different plots.

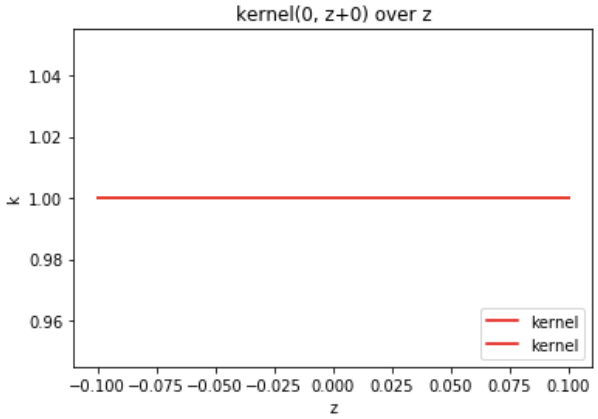


**Figure 3:** Kernel at x=0, over z

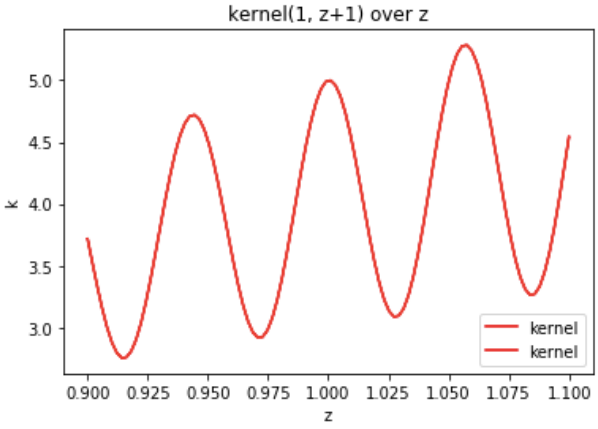


**Figure 4:** Kernel at x=1, over z

**Q4**

A GLM using a Gaussian RBF kernel was designed on the Mauna Loa, Rosenbrock, and Iris datasets. The results are summarized in Table 1 below. We see that for the classification dataset Iris, the optimal $\lambda$ was larger as compared to the optimal $\lambda$ selected for the regression sets Mauna Loa and Rosenbrock. The opposite was true for $\theta$, as a larger $\theta$ was favored for the regression sets. These parameters were selected by looping over all possible values and selecting the one which minimized RMSE for regression, or maximized Test Ratio for classification. I found this model to be quite heavy computationally, with many nested loops, it took the program a very long time to execute over the three datasets.

**Table 1**: Gaussian RBF Results

| Dataset | Optimal $\lambda$ | Optimal $\theta$ | Test Error (RMSE) |
|---|---|---|---|
| Mauna Loa | 0.001 | 1 | 0.150 |
| Rosenbrock | 0.001 | 2 | 0.148 |
| - | - | - | Test Ratio |
| Iris | 1 | 0.5 | 1 |

**Q5**

A greedy regression algorithm was implemented on the Rosenbrock dataset using a dictionary of Gaussian kernels centered at the training points. Orthogonal matching pursuit was used as a metric to select a different basis function at each iteration. The stopping criterion used was the minimum description length [MDL] shown below :

$$(N/2)*\log(l2\text{-loss}) + (k/2)*\log(N)$$

Where l2-loss is the least-squares training error. The number of iterations, k, used in this analysis was chosen to be 1000 for 3 different settings of shape parameter at 0.01, 0.1 and 1.0. The test error, and a plot of model complexity (sparsity) vs MDL value is shown below for each setting. Here the MDL value is considered to be a surrogate of the generalization error.

*Shape Parameter = 0.01*

**Test RMSE**: [0.83817104]

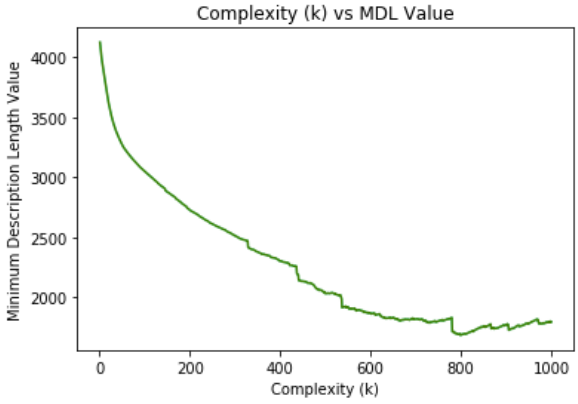


**Figure 5:** Complexity vs MDL Value, Shape Parameter = 0.01

*Shape Parameter = 0.1*

**Test RMSE**: [110.4417144]

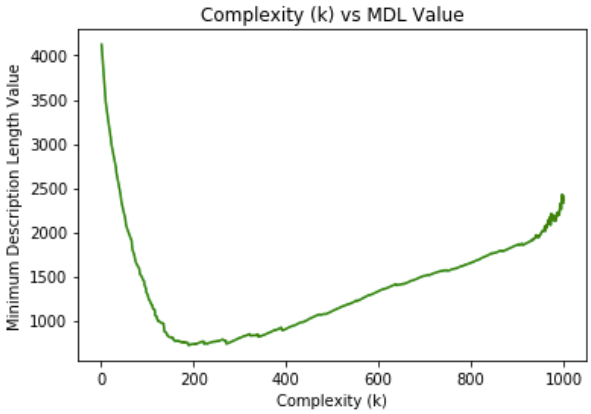


**Figure 6:** Complexity vs MDL Value, Shape Parameter = 0.1

*Shape Parameter = 1.0*

**Test RMSE**: [1589.91918489]

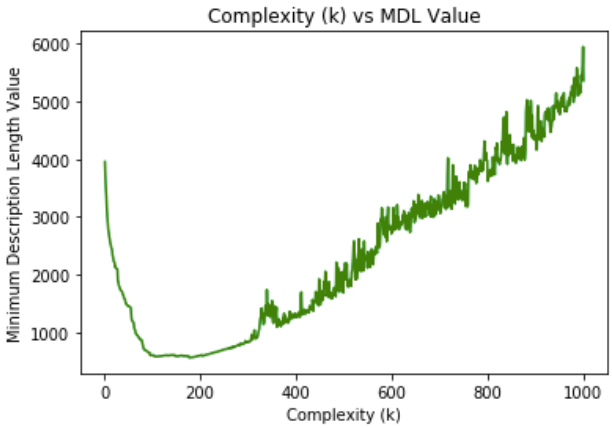


**Figure 7:** Complexity vs MDL Value, Shape Parameter = 1.0

Evidently, a shape parameter of 0.01 yielded the best results for test RMSE. From Figure 5, as k increases to 1000, the MDL value continues to decrease and we do not see any overfitting. As we then increase the shape parameter to 0.1 and 1.0, we see overfitting occur much faster and more aggressively. This also correlates with an RMSE that drastically increases with the shape parameter. The overfitting is shown by the upward trends at the end of the graphs in Figure 6 and Figure 7, which get steeper as the shape parameter increases. With a shape parameter of 1, we can also see that as the complexity increases, the MDL Value begins to oscillate. These results lead us to the overall conclusion that increasing the shape parameter also increases the amount of overfitting for this dataset.