# Gaussian Processes to Predict Gene Expression in Differentiating Induced Pluripotent Stem Cells

Nirmal Krishnan
Advanced Topics in Genomics Data Analysis

December 21, 2017

## 1 Introduction

Induced pluripotent stem cells (IPSCs) are a type of stem cell that can be generated from sampled adult cells. These cells are pluripotent because they have the potential to differentiate into any of the three germ layers: "endoderm (interior stomach lining, gastrointestinal tract, the lungs), mesoderm (muscle, bone, blood, urogenital), or ectoderm (epidermal tissues and nervous system)" [1]. In medicine, these cells are incredibly promising because they can be differentiated to any cell in the body, allowing regeneration of tissue for patients who have been injured in an accident or diseased. Generating these cells works as follows:

1. Cells are isolated from a patient.
2. Isolated cells are treated with the products of "reprogramming factors."
3. Cells are cultured for 3-4 weeks. After this period, the cells are pluripotent stem cells.
4. The culture conditions are changed to simulate cells into differentiating into desired cell type.

For differentiation of IPSc into cardiomyocyte, the cell type discussed in this paper, step 4 takes approximately 16 days to complete. Although the "reprogramming factors", a set of genes that can induce the iPSC state, have been identified, little is known about how genes and their expressions evolve in the differentiation process over time [2].

This paper aims to better understand and model the change in expression of genes over the differentiation process. Prediction of gene expression in ordered samples typically involves using baseline statistical methods like linear regression; this project, however, aims to apply newer methods, specifically Gaussian processes (GP), to this task of prediction and compare them with the baseline. Gaussian processes have quickly gained acclaim in the scientific community for their extreme flexibility and predictive accuracy. This study aims to answer the question of whether GPs are appropriate for prediction in the context of gene expression in differentiating iPSCs or whether these newer methods overfit this data compared to the traditional baseline methods.

# 2 Data

The data for this project is gene expression (RNA seq) drawn from IPSCs over their 16 day differentiation process into cardiomyocyte cells. 10 cell lines were differentiated, for a total of 154 samples [1] and 15,794 total genes used in this study. These results come from the Gilad lab at the University of Chicago.

# 3 Methods

## 3.1 Data Pre-processing

The design matrix for gene expression undergoes several rounds of preprocessing before being used in model training. First, the data is quantile normalized across genes and samples. This is done primarily because gene expression is incredibly noisy, so outliers need to be scaled appropriately.

Next, surrogate variable analysis (SVA) is run on the quantile normalized matrix and 21 latent factors unrelated to time are formed [3]. Using the 21 latent factors, a linear model is trained to predict expression for each gene using all samples. The predicted values for each gene is then subtracted from the quantile normalized matrix, resulting in a matrix that has effectively "regressed out" the covariates discovered from SVA.

[1] 6 of the samples were removed due to quality concern reasons

### 3.2 Gaussian Processes Background

Gaussian processes are stochastic processes such that ever point in a finite continuous space is associated with a normally distributed random variable. Gaussian processes have many special properties such as the collection of any set of points being a multivariate Gaussian distribution [4]. Viewed as a machine-learning algorithm, for prediction, GPs are not just point estimates, but also offer ranges of uncertainty around the predicted mean.

GPs are fully parametrized by a mean and kernel function, the latter of which measures the similarity among different points in the input space. A key fact of the Gaussian process is that they can be completely defined by the kernel function, meaning a zero mean Gaussian process with a fitted kernel completely defines the process's behavior [5]. Thus the choice of kernel function and the prior on its parameters completely affect the results of the trained GP, since the mean function is set to zero in most practical applications. Consequently, the choice of kernel is an extremely import decision and three different styles were used and compared in this study.

### 3.3 Kernel Selection

Squared Exponential:

$$k(x_i, x_j) = exp(\frac{1}{2}d(x_i/l, x_j/l)^2$$

This kernel, also known as the radial basis function (RBF) kernel, is stationary and infinitely differentiable and can consequently generalize smooth curves extremely well. It is parameterized by a length scale parameter where $l > 0$. A standard non-informative prior of $l = 1$ was used in this paper.

Matern:

$$k(x_i, x_j) = \frac{1}{\Gamma(\nu)2^{\nu-1}}(\sqrt{2\nu}d(x_i/l, x_j/l))^\nu K_\nu(\sqrt{2\nu}d(x_i/l, x_j/l))$$

This kernel is a generalization of the RBF kernel-it is also stationary and infinitely differentiable; however, it has an extra parameter $\nu$ that can control the smoothness of the resulting function. A standard non-informative prior of $l = 1$ and $\nu = 1.5$ was used in this paper.

Exponential-Sine-Squared:

$$k(x_i, x_j) = exp(-2(sin(\pi/p \times d(x_i, x_j))/l)^2)$$

This kernel enables the modeling of periodic function because of its use of the sine function. It is parameterized by a length scale parameter where $l > 0$ and a periodicity parameter $p > 0$. A standard non-informative prior of $l = 1$ and $p = 1$ was used in this paper.

### 3.4 Model training

Since training takes an extensive period of time, initially, the top 100 genes with the most variance in the original design matrix are selected to be used for evaluation.

The posterior distributions for the parameters listed in the previous section as well as the beta coefficients for linear regression (referred to as $\Theta$ below) were trained using the procedure described below:

**Algorithm 1** Model Training and Prediction Procedure

```
for model do
    for gene do
        for cell line do
            for i in time step do
                x ← time[0:15], except time step i
                y ← gene expression[0:15], except time step i
                Θ ← fit(x, y)
                Predict using Θ on time step i and save results
```

We then record the absolute prediction error for each model separately and compare the results using the Wilcoxon rank sums test.

We used the sklearn package available in python to train the parameters for both the linear regression and GPs. The linear regression beta coefficients are trained by minimizing the sum of squares of error. The GP's parameters are trained using the variational inference posterior approximation method [6].

### 3.5 Gene clustering

One issue with gene expression is that the results are extremely noisy and prone to outliers. Quantile normalization can prune the effects of outliers; however, it does not effectively control the noise associated with non-outlier genes. In order to control for this effect, genes with the top 1000 variance

were clustered (k=10, 100) and the models were re-trained on the cluster means. This clustering is done using a standard k-mean clustering algorithm. In summary, models were trained on three data encodings: genes with the top 100 variance, gene mean clusters with k=10, and gene mean clusters with k=100.

# 4 Results

Figure 1:

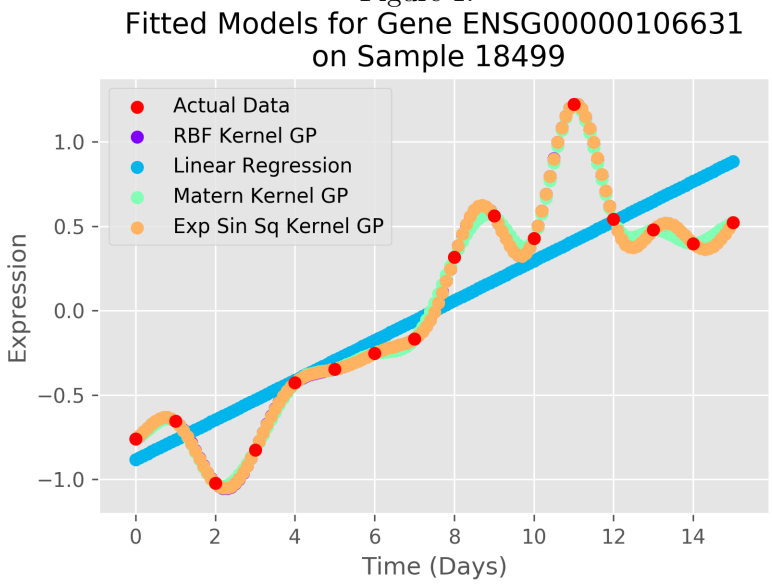
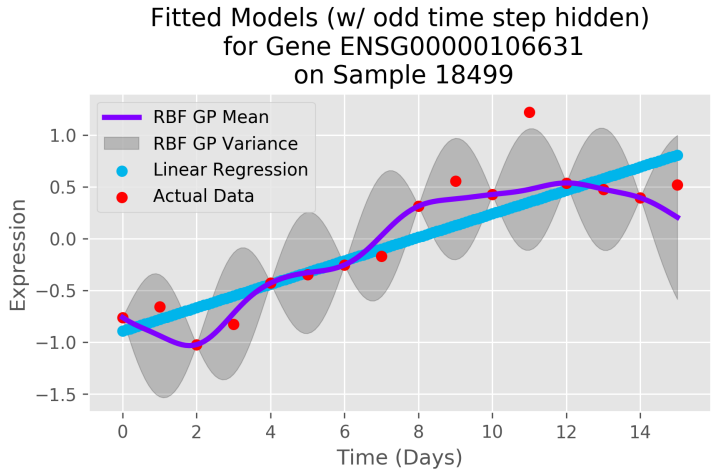


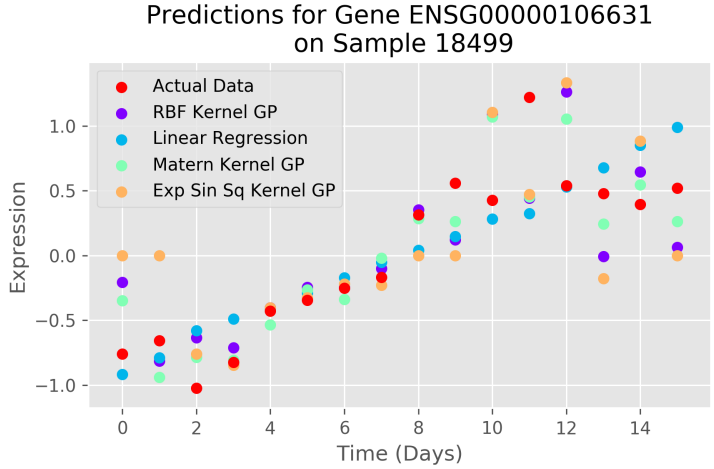

Figure 1 shows a demonstration of the flexibility of the Gaussian processes to fit to the data. Here, we have chosen a random gene and cell line and fit a GP across all time steps. We can see that all of the GP kernels successfully capture the non-linear progression of the expression trajectory while the linear regression struggles to maintain the same accuracy. Clearly, the GP has the flexibility to model nearly any progression of data; however, in the next figure (2), we now hide all of the odd time steps.

Figure 2:



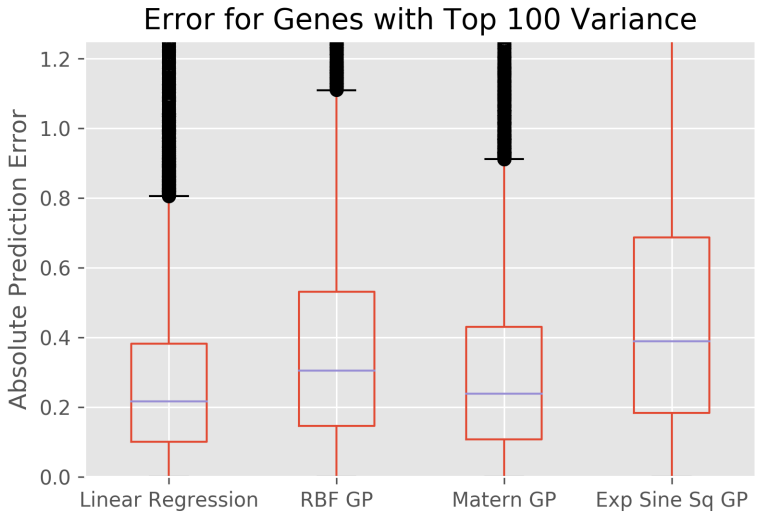

From figure 2, we can see the predictive difference between the GP (RBF used here for simplicity) and linear regression narrows significantly when data is hidden. This illustrates how GPs tend to overfit when given many time points, such as 8 in figure 2. Many modern GP methods like the spare Gaussian process approximation attempt to shrink the time space so that the GP will not overfit like it does in figure 2 [7].

Figure 3:



In figure 3, we show the results from Algorithm 1, the model training and prediction procedure, on a single gene and cell line. We can see that the results are even more varied here–on many data points the linear regression appears to outperform the GPs at predicting expression.

Figure 4:




In figure 4, we plot the absolute prediction error across our different models. We can see that the Matern kernel performs the best across the GP configurations. However, it appears to have a higher tail of error than the linear regression. This result is seen clearly in the Wilcoxon rank sums results below.

| Model 1 | Model 2 | Test Statistic | p-value |
|---|---|---|---|
| Linear Regression | RBF GP | -30.38 | 7.98E-203 |
| Linear Regression | Matern GP | -8.76 | 1.86E-18 |
| Linear Regression | Exp Sine Sq GP | -50.81 | 0 |

Table 1: Wilcoxon Rank Sums Results for Top 100 Variance Gene Predictions

As we can see, the linear regression performs better than all of the GP configurations, indicating that they are likely overfitting and that the simpler baseline better generalizes the data.

Figure 5:

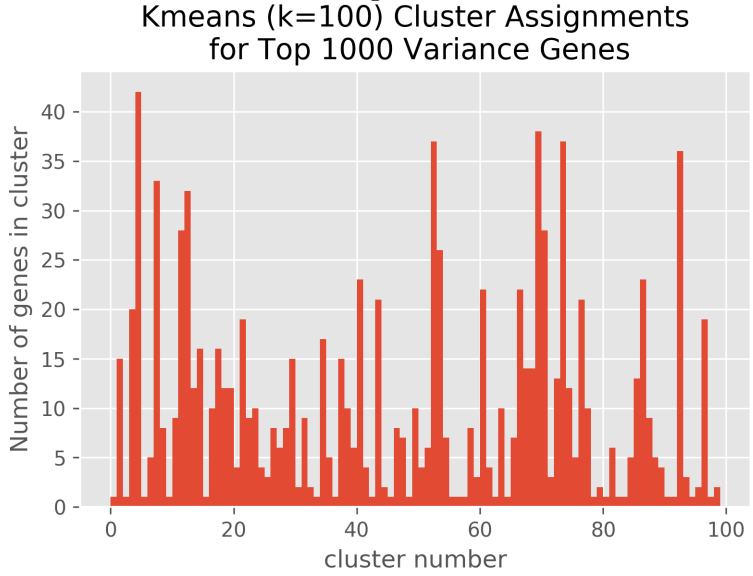



Figure 6:


Figures 5 and 6 show the kmeans clustering results for k=100 and k=10.

In order to ensure that our initial results for the top 100 variance genes are not due to observational noise, we re-ran the results on these two clustered configurations.

Figure 7:


| Model 1 | Model 2 | Test Statistic | p-value |
|---|---|---|---|
| Linear Regression | RBF GP | -35.17 | 4.44E-271 |
| Linear Regression | Matern GP | -9.86 | 6.00E-23 |
| Linear Regression | Exp Sine Sq GP | -49.55 | 0 |

Table 2: Wilcoxon Rank Sums Results for k=100 Cluster Means Predictions

We can see in this configuration as well, the same pattern holds- the Matern kernel performs the best of the GPs; however, the linear regression still better models the data.

Figure 8:


| Model 1 | Model 2 | Test Statistic | p-value |
|---|---|---|---|
| Linear Regression | RBF GP | -18.82 | 5.04E-79 |
| Linear Regression | Matern GP | -4.75 | 2.01E-6 |
| Linear Regression | Exp Sine Sq GP | -25.13 | 2.39E-139 |

Table 3: Wilcoxon Rank Sums Results for k=10 Cluster Means Predictions

We can see that the gap between the linear regression and matern kernel narrows here; however, even still the linear regression significantly outperforms all of the GPs.

Across all configurations of clustering, the linear regression consistently outperforms the Gaussian process models. It is certainly likely that the GP overfitting is a result of the high number of data points along the x-axis and a dimmensionality reduction across time points may prove effective at increasing performance. GPs obviously have the flexibility to model seemingly any dataset, given the results from figure 1; however, tuning a GP with the correct kernel and prior is no trivial task and in this paper we are unable to produce a configuration that works well on this dataset.

## 5 Conclusions

Gaussian processes are a powerful, flexible tool for modeling time series data; however, in IPSc gene differentiation, they heavily overfit the data, resulting in poor predictive accuracies compared to baseline methods like linear regression.

## References


[1] Wikipedia contributors. *Induced pluripotent stem cell — Wikipedia, The Free Encyclopedia*. [Online; accessed 21-December-2017]. 2017. URL: %5Curl%7Bhttps://en.wikipedia.org/w/index.php?title=Induced_pluripotent_stem_cell&oldid=813682262%7D.

[2] Ben-Nun I.F. "Generation of Induced Pluripotent Stem Cells from Mammalian Endangered Species". In: *Cell Reprogramming* 1330 (2015).

[3] Leek JT. "Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis". In: *PLoS Genet* (2007).

[4] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN: 026218253X.

[5] Wikipedia contributors. *Gaussian process — Wikipedia, The Free Encyclopedia*. [Online; accessed 21-December-2017]. 2017. URL: https://en.wikipedia.org/w/index.php?title=Gaussian_process&oldid=815269770.

[6] Wainwright M and Jordan M. "Graphical Models, Exponential Families, and Variational Inference". In: *Foundations and Trends in Machine Learning* (2008).

[7] Snelson E. "Local and global sparse Gaussian process approximations". In: *Arxiv* (2007).


## 6 Course Project Statements

Only a small portion of this assignment, the data pre-processing step, was done for my work at the Battle Lab. All other work was done specifically for this assignment.