On Gaussian Processes for Regression

Jeffrey Alido
Department of Electrical and Computer Engineering
Boston University
`jalido@bu.edu`

Shashank Manjunath
Department of Electrical and Computer Engineering
Boston University
`manjuns@bu.edu`

Abstract

Gaussian processes emerged in machine learning as a powerful tool for regression and classification that provides interpretability through kernel choice and uncertainty quantification. By leveraging properties of multivariate normal distributions and Bayes's rule, we may infer a probability distribution over possible functions when fitting a dataset. This Bayesian framework allows flexibility through choosing a covariance function as a prior belief about the dataset, which can provide further insight into the trends of the training data. We implement a multi-dimensional Gaussian process regressor and evaluate its performance on the Boston Housing dataset, which is comparable to those in the top 10 of the Kaggle competition. Furthermore, we perform optimization on the hyperparameters through 5-fold cross-validation. We provide a MATLAB implementation of Gaussian Processes for Regression at https://github.com/shashankmanjunath/GaussianProcessRegression.

1 Gaussian Random Variables

A random variable is a function that maps from an event space to a measurable space. The event space represents a set of all possible outcomes that the random variable may take, and the measurable space is a probability measure between 0 and 1 (inclusive). We say that a random variable X is normally distributed if the event space has a Gaussian probability distribution, fully characterized by two parameters: a mean μ and variance σ^2:

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

For a one-dimensional Gaussian random variable, we refer to its distribution as a univariate Gaussian distribution. A set of Gaussian random variables may be characterized jointly as a multivariate Gaussian distribution, with joint probability distribution fully characterized by a mean vector and a covariance matrix:

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

Preprint. Under review.

where $\boldsymbol{\mu}$ is the mean vector, and Σ is the covariance matrix whose entries describe the covariance between each pair of random variables. Multivariate Gaussian random variables form the foundation for Gaussian processes, by characterizing the functions drawn from the Gaussian process.

2 Gaussian Process

A random process is essentially a collection of random variables jointly characterized as a set or vector of random variables with a multivariate joint probability distribution. A Gaussian process $f(\boldsymbol{x})$ is defined as a random process where each set of random variable in the random process is has a multivariate Gaussian distribution. $f(\boldsymbol{x})$ is fully characterized by a mean function $m(\boldsymbol{x})$ and covariance function, $K(\boldsymbol{x}, \boldsymbol{x}')$:

$$\begin{aligned} f(\boldsymbol{x}) &\sim \mathcal{GP}(m(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}')) \\ K(\boldsymbol{x}, \boldsymbol{x}') &= \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))] \end{aligned}$$

Typically, the mean function is assumed to be zero. The kernel for the covariance is chosen based on some prior belief about the dataset; more on kernels is discussed in 3.1. In a Gaussian process, we assume that any new observed points follow the same multivariate normal observed in our training data. GP's are non-parametric models, unlike models such as standard linear regressors or neural networks, since they do not have "weights" in the sense that a linear regressor or neural network has weights[1]. Rather, a GP is characterized by a mean and covariance function, from which predicted can be drawn[4]. While we have defined a Gaussian process, we now aim to create a scheme under which to perform regression.

3 Regression

Suppose we observe training data $\boldsymbol{x} \in \mathbb{R}^{n \times d}$, training labels $\boldsymbol{y} \in \mathbb{R}^n$, testing data $\boldsymbol{x}' \in \mathbb{R}^{m \times d}$ and choose kernel κ, and choose noise parameter $\sigma_n^2 \in \mathbb{R}$. Then the mean and covariance functions are given by

$$\begin{aligned} m(\boldsymbol{x}) &= \kappa(\boldsymbol{x}, \boldsymbol{x}')^\top (\kappa(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 I_n)^{-1} \boldsymbol{y} \\ K(\boldsymbol{x}, \boldsymbol{x}') &= \kappa(\boldsymbol{x}', \boldsymbol{x}') - \kappa(\boldsymbol{x}, \boldsymbol{x}')^\top (\kappa(\boldsymbol{x}, \boldsymbol{x}) + \sigma_n^2 I_n)^{-1} \kappa(\boldsymbol{x}, \boldsymbol{x}') \end{aligned}$$

where I_n indicates the identity matrix. Those interested in a thorough derivation of the results are encouraged to consult Chapter 2 of [5].

3.1 Kernels

Covariance Functions or kernels, denoted $\kappa(\boldsymbol{x}, \boldsymbol{x}')$, form the core of a Gaussian process. Kernels allow projection of input data into an alternate feature space, allowing easier separability of data in this new feature space. Gaussian processes leverage kernels to featurize input data. In particular, if we have a function $\Phi(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^n$, we can write the kernel defined by this function as:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle = \Phi(\boldsymbol{x})^\top \Phi(\boldsymbol{x}')$$

We illustrate specific kernels that we used below. These kernels can additionally be combined through addition and multiplication of the kernels together, though we do not explore this as part of this project. The kernel must be chosen prior to fitting the GP, and is typically chosen based on the specific modelling problem at hand.

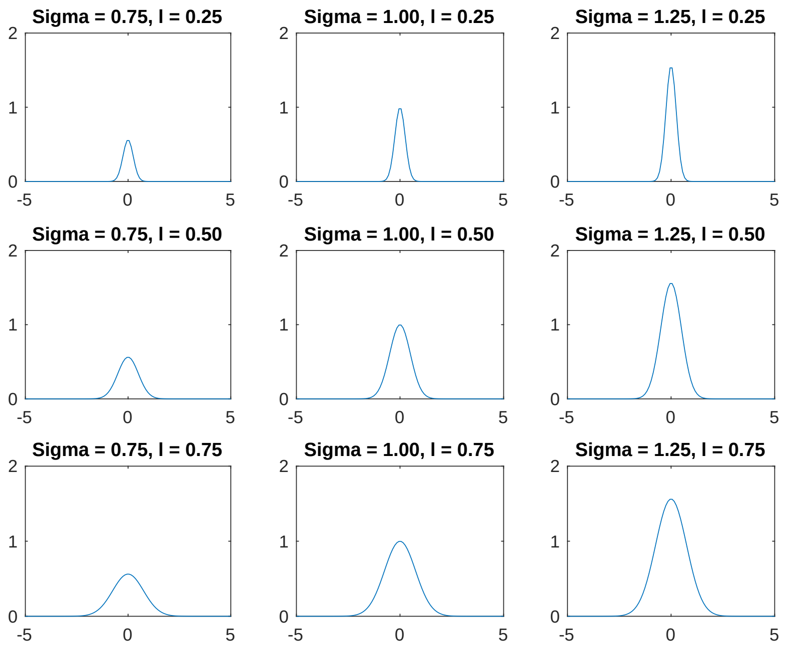
3.1.1 Radial Basis Function (RBF) Kernel/Square Exponential Kernel

The Radial Basis Function (RBF) Kernel, also known as the Squared Exponential Kernel, is given by:

$$\kappa_{RBF}(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\ell^2}\right)$$

This kernel is parametrized by two parameters, the lengthscale ℓ and the variance σ^2. The lengthscale determines the width of the kernel, and the variance scales the kernel[2]. We provide an images of the RBF Kernel with various lengthscales and variances in Figure 1

Figure 1: Graph of a square exponential kernel/RBF kernel for various lengthscales and variances



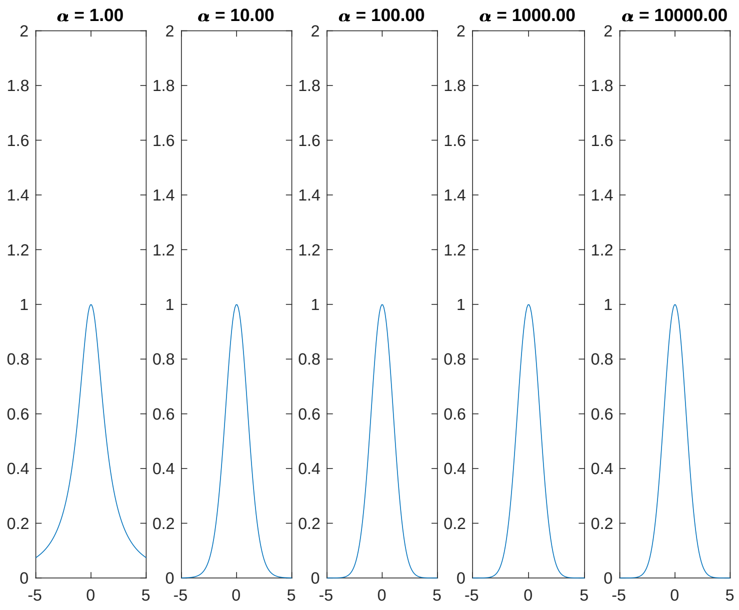
3.1.2 Rational Quadratic Kernel

The Rational Quadratic Kernel is another standard kernel is similar to the RBF kernel. It can be constructed from summing RBF kernels with varying lengthscales. The kernel is given by:

$$\kappa_{RQ}(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \left(1 + \frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\alpha\ell^2}\right)^{-\alpha}$$

This kernel is parametrized by three parameters, the lengthscale ℓ, the variance σ^2, and the lengthscale weighting parameter α[2]. We provide images of the Rational Quadratic Kernel with various α values in Figure 2

Figure 2: Graph of a rational quadratic kernel for $\sigma = 1$, $\ell = 1$, and various α values



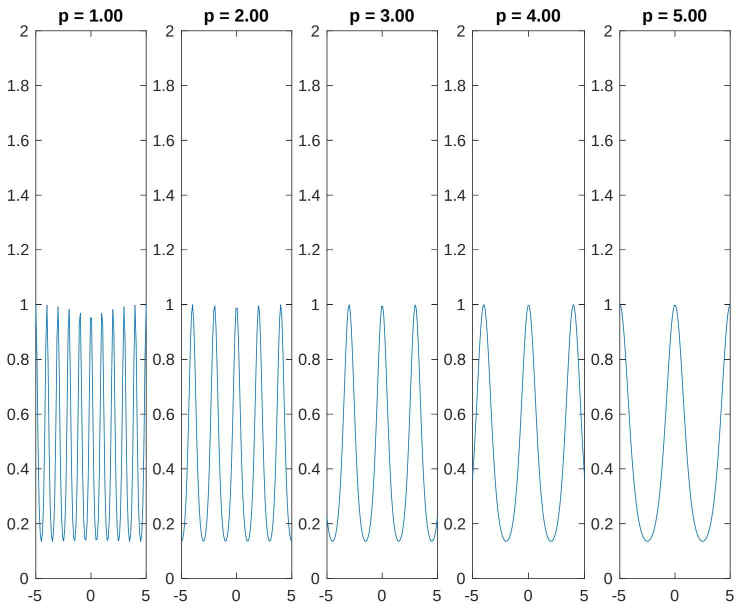
3.1.3 Periodic Kernel

The periodic kernel allows us to model periodic functions. It is given by:

$$\kappa_P(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{2\sin^2(\pi\|\boldsymbol{x} - \boldsymbol{x}'\|)}{p\ell^2}\right)$$

This kernel parametrized by two parameters, p which describes the period of the function, and ℓ which is the lengthscale[2]. We provide images of the Periodic Kernel with various p values in Figure 3.

Figure 3: Graph of a rational quadratic kernel for $\sigma = 1$, $\ell = 1$, and various p values



3.1.4 Other Kernels

We additionally test other kernels, including the Linear Kernel, Local Periodic Kernel, and the Polynomial Kernel. The Linear kernel has no hyperparameters, and is given by:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{x}, \boldsymbol{x}' \rangle$$

The polynomial kernel has hyperparameters c, which controls the zero crossing of the kernel, and d which controls the order of the polynomial. This kernel is given by:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^\top \boldsymbol{x}' +)^d$$

We additionally test one kernel which is a combination of two kernels already discusses, the Local Periodic Kernel. This kernel is the product of a periodic kernel and a square exponential kernel, and is given by:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\ell^2}\right) \exp(-\frac{2\sin^2(\pi\|\boldsymbol{x} - \boldsymbol{x}'\|)}{p\ell_p^2})$$

This kernel is parametrized by σ^2, the overall variance, ℓ_p, the lengthscale of the periodic function, p, the period of the periodic function, and ℓ, the lengthscale of the square exponential kernel.

4 A Simple Demo

Here we demonstrate GP regression on sinusoidal data fit using a square exponential kernel with lengthscale $\ell = 1.5$, and variance $\sigma^2 = 1.0$. The true function is represented by the black dashed line, with training points represented by the solid red dots. The orange yellow and blue curve represent samples from the distribution of possible functions, and the red dashed line represents the 95% confidence interval.

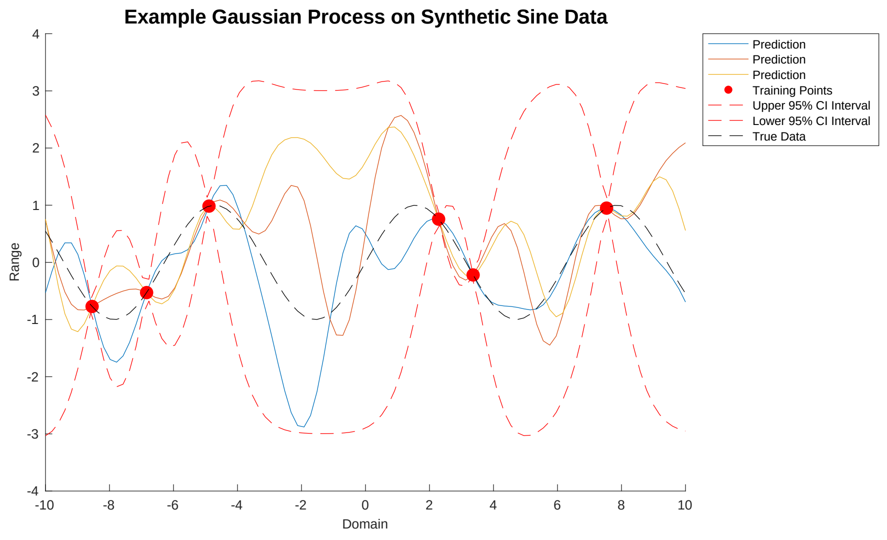


Figure 4: Visualization of a Gaussian process. The dashed black line is the synthetic function (a sinusoid), not provided to the GP. We provide the red training points, and fit the GP as described in Section 3. We then draw three example functions (blue, red, and yellow solid lines) from the GP, as well as calculate the bounds for the 95% confidence intervals (dashed red lines)

5 Boston Housing Dataset

The Boston Housing Dataset, originally published in 1978 contains 506 data points, each containing 13 features and 1 label for regression[3]. The dataset provides the median value of houses in Boston suburbs. This dataset is particularly suitable for Gaussian processes, as the dataset is quite small. Use of Gaussian processes allows us to accurately quantify our uncertainty for each The label is the median value of owner-occupied homes in $1000s, and all other features are used for model fitting. The features included in the dataset are given in the table in Table 1.

Table 1: Table of Boston Housing Dataset feature names and features

Feature Name	**Feature Description**
CRIM	Per capita crime rate by town
ZN	Proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	Proportion of non-retail business acres per town.
CHAS	Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX	Nitric oxides concentration (parts per 10 million)
RM	Average number of rooms per dwelling
AGE	Proportion of owner-occupied units built prior to 1940
DIS	Weighted distances to five Boston employment centres
RAD	Index of accessibility to radial highways
TAX	Full-value property-tax rate per $10,000
PTRATIO	Pupil-teacher ratio by town
B	$1000(Bk - 0.63)^2$ where Bk is the proportion of Black people by town
LSTAT	% lower status of the population
MEDV	Median value of owner-occupied homes in $1000's

For the application of Gaussian Processes, we use the regression task, i.e. fitting to the MEDV feature. Prior to fitting on the data, we normalize the data per feature. Specifically, for each feature in the dataset, we perform the following operation:

$$X_{\text{feat}} = \frac{X_{\text{feat}} - \mu(X_{\text{feat}})}{\sigma(X_{\text{feat}})}$$

where $\mu(X)$ is the mean value of that feature in the training set, and $\sigma(X)$ is the standard deviation of that feature in the training set. We additionally normalize the MEDV feature, and convert it back to non-normalized units before calculating our RMSE.

6 Results

We perform regression on the Boston Housing Dataset using our own GP implementation formulated as described in Section 3, MATLAB's built-in SVM Regressor function, and our own implementation of linear regression. 5-fold cross validation hyperparameter search was performed in training and the root mean squared error (RMSE) is used as our performance metric. More specifically, we calculate the RMSE using the following formula:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}}$$

where y_i is the true value, $\hat{y}_i$ is the predicted mean value of our GP, and N is the number of samples. Our results are shown in Table 2.

The smallest RMSE is the GP with a rational quadratic kernel, indicating the best performance. The rational quadratic kernel is equivalent to the sum of RBF kernels with different lengthscales, for increased robustness. The poor performance of periodic and locally periodic kernels reveal that there is likely no periodic trends in the data.

The results from the GP regression provide further information than SVM or linear regression through uncertainty quantification. This feature, along with incorporating prior beliefs through kernel choice gives the GP algorithm its interpretability.

Table 2: Table of Boston Housing Dataset feature names and features

	Gaussian Process Regressor	**SVM Regressor**	**Linear Regression**
Linear Kernel	**4.751**	4.935	**4.751**
Square Exp Kernel	**3.586**	9.014	4.751
Rational Quadratic Kernel	**3.560**	9.013	4.751
Periodic Kernel	8.944	9.017	**4.751**
Local Periodic Kernel	5.065	9.014	**4.751**
Polynomial Kernel	**4.213**	80.389	4.751

7 Conclusions

In this writeup, we provided a brief introduction to Gaussian Processes, discussed algorithms to fit GPs for regression, provided a simple demo, and tested out GPs with various kernels on the Boston Housing dataset, showing good algorithm performance. We additionally provide our implementation in MATLAB at https://github.com/shashankmanjunath/GaussianProcessRegression.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] David Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD Thesis, Computational and Biological Learning Laboratory, University of Cambridge, 2014.

[3] David Harrison and Daniel L. Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5(1):81–102, 1978.

[4] M. Kuss. *Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning*. PhD Thesis, Biologische Kybernetik, March 2006. Backup Publisher: Technische Universität Darmstadt, Darmstadt, Germany.

[5] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. OCLC: ocm61285753.