A COMPARATIVE STUDY BETWEEN PARAMETRIC AND NON-PARAMETRIC REGRESSION (GAUSSIAN PROCESS) MODELS

Shubha Sankar Banerjee

Roll No.: [redacted]

Reg. No.: [redacted]-[redacted]-[redacted]-[redacted]

Under the supervision of Dr. Durba Bhattacharya

Department of Statistics

St. Xavier’s College (Autonomous), Kolkata

DECLARATION

“I affirm that I have identified all my sources and that no part of my dissertation paper uses unacknowledged materials.”

CONTENTS

ABSTRACT

In statistical analysis, it is often our objective to study the relationship between response and predictor variables, so as to draw some knowledge about future outcomes. This is the study of Regression. The basic problem of regression is identifying the form of the dependence of the response on predictor, it may be known or in some cases, unknown. Based on this difference in the nature of the form of dependence we will be dwelling in the topics of OLS regression models and the Gaussian Process Regression Model. This project deals with the objective of understanding the concepts of Gaussian Process Regression Model, which is a type of Non-parametric Regression. We will focus on the comparison of OLS and GPR model using different data sets and also implement the two methods on a real life data set to establish their efficacy.

INTRODUCTION

Statistical Prediction is the method of using the data from an informative experiment E to make some statement about the outcome of a future experiment F. The prediction statements are commonly of inference types whose purpose is to give some indication of the likely outcome of F, or in some cases to suggest some subset of possible outcomes in which the actual outcomes of F are likely to fall.

In the vast arena of predictive modelling and problems, one of the most implemented modelling techniques is that of the Theory of Regression. In statistical modelling, regression analysis is a set of statistical process for estimating the relationships between a dependent variable (often called the “outcome variable”) and one or more independent variables (often called “predictors”, “covariates”, or “features”). Regression analysis is widely used for prediction, forecasting or to infer causal relationships between independent and dependent variables. This technique has vast usage in time series modelling. **For example**, the relationship between amount of rainfall and the annual production of rice can be best explained and in fact further studied through regression.

In statistics, based on nature of study, we can use some of the many different types of regression. Some of the different types of regression are: *Ordinary Least Squares regression (Linear or polynomial), Quantile regression, Gaussian process regression, Ridge Regression, Lasso Regression, Elastic Net Regression, Principal Components Regression (PCR), Partial Least Squares (PLS) regression, Poisson Regression, Negative Binomial regression, Cox Regression* and many more.

We may have models which fall into the category of **parametric** regression model, due to our knowledge of the function that describes the relationship between response and explanatory variables. In many situations, the form of the relationship may not be known. Such types of regression fall into the category of **non-parametric** regression models. Thus Non-parametric

Regression (NPR) is a category of regression analysis where the shape of the functional relationships between the response and the explanatory variables are not predetermined but can be adjusted to capture unusual or unexpected features of the data. NPR requires larger sample sizes than parametric regression models because the data must supply the model structure as well as the model estimates.

METHODOLOGY (*in brief*)

Let us consider some **noisy observation data** which is provided to us as dependent variable at certain values of some independent variable (say) **x**. Our problem, which in this case pertains to prediction, is to find the best estimate of the dependent variable as a new value $\mathbf{x}^*$**.**

Case I: We "expect" the dependence of the response variable on the predictor to be linear. We suggest the functional form of this dependence to be f(x). We further make some assumptions based on our input data and thereon use **OLS** method to fit a straight line (which is the case of a linear regression). Further, we may also suspect that f(x) is of some other form (quadratic, cubic, or even non-polynomial), and use the principles of model selection to choose among various possibilities with the aim of trying to explain the dependence in the most efficient manner possible.

Case II: Suppose f(x) does not have a specific form. In that case we cannot apply the OLS regression model to fit the data. In this case, here in our project, we will be applying a form of Non-parametric regression or specifically the **Gaussian Process Regression (GPR)**. In essence, we this process is not limited by a functional form. Instead of calculating the probability distribution of parameters of a specific function, GPR calculates the probability distribution over all admissible functions that fits the data.

In this project we will thus be interested in two forms of regression based on whether the form of dependence of the dependent variable on the predictor is known or not. If known, as in Case I, we will be applying an OLS regression model which is a **parametric** method and if unknown, as in Case II, we will be interested in a **non-parametric** approach particularly the Gaussian Process Regression. We will be comparing the two approaches based on certain points.

OLS REGRESSION

Let us consider that we are provided with observations on 2 variables (say) **x** and **y**. Let us consider variable y to be the dependent variable and variable x to be the independent variable or the predictor. The problem is to **find the value of y for a given value of x**. To solve such a problem, we need a mathematical relationship between y and x which would characterise the dependence of y on x. Some examples of such mathematical functions are:

I. $y = a + bx + \varepsilon$

II. $y = a + bx^2 + \varepsilon$

III. $y = a + b^2x + \varepsilon$

For simplicity, we choose (**I**) as our model. The terms "a" and "b" represent the parameters of the model. Parameters in statistical analysis are elements that characterises the model or the dependence. Thus to fit our assumed model of a linear dependence of **y** on **x**, we need to determined the values of "a" and "b" from the observed data. The term "ε" denotes the error committed in assuming this model i.e. the error committed in fitting a straight line based on observed values of **y** and **x**. Without loss of generality, let us assume that we are provided with n given pairs of values of **x** and **y**, the **i**$^{\textbf{th}}$ pair being denoted by $\mathbf{(x_i,y_i)}$.

MODEL AND ASSUMPTIONS:

Broadly there are 4 assumptions associated with linear regression model:

A) **The model must be "linear in parameters":** The term "linear in parameters" imply that no parameter exists as an exponent or is multiplied or divided with another parameter. Models (I) and (II) are linear in parameters, with model (II) being "non-linear in variables". The assumption is not satisfied for model (III) as it is not linear in the parameter "b".

B) **The variable x and the error variable is uncorrelated:** The distribution of the error terms are independent of the predictor variable x.

C) **Errors or residuals are independently and identically distributed Normal variables and are homoscedastic:** Homoscedastic assumptions imply that the error terms have equal variance. This assumption also includes another underlying assumption that the mean of the residuals must be zero. Thus, if ε_i denote the residual for the ith observation of x, $\boldsymbol{\varepsilon_i \sim N(0,\sigma^2)}$ **and** $\boldsymbol{\varepsilon_i}$**'s are i.i.d. random variables with** $\boldsymbol{Cov(\varepsilon_i, \varepsilon_j) = 0 \ \forall\, i \neq j}$.

D) **The observations are random samples drawn from a population:** This assumption implies that the sample taken from the population for the linear regression model must be drawn randomly. The number of observations taken in the samples for making the linear regression should be greater than the number of parameters to be determined. The **x's should be non-stochastic** i.e. the values of x should be fixed.

Since we have assumed (I) as our model, we consider the model function **y = a + bx**, which describes a line with **slope b** and the **y-intercept a**. In reality, such an exact relationship may not hold for the largely unobserved population of values of independent and dependent variables; we thus call the unobserved deviations in our model from the actual observations as **errors** or **residuals.** We have already considered n pairs of data $\boldsymbol{\{(x_i, y_i), i=1,2,...,n\}}$. We thus define the underlying relationship between y_i and x_i involving this error term ε_i by

$$\boldsymbol{Y_i = a + bx_i.}$$

The goal is to find estimated values $\boldsymbol{\hat{a}}$ and $\boldsymbol{\hat{b}}$ for the parameters **a** and **b** which would provide the "best" fit in some sense for the given data points. Here, the "best" fit will be understood as n **least-squares** approach. The difference $\boldsymbol{y_i\text{-}Y_i}$ is thus the error committed in assuming the linear relationship between **x** and **y**. Since the line is used for estimation purposes, it is reasonable to require that **a** and **b** should be such as the error estimates $\hat{\varepsilon}_i$ will are as small as possible. Thus we determine the estimates of model parameters **a** and **b** by the method of **ordinary least squares (OLS).**

For any given pair of observations, $\hat{\boldsymbol{\varepsilon}}_{\boldsymbol{i}} = \boldsymbol{y}_{\boldsymbol{i}} - \hat{\boldsymbol{a}} - \hat{\boldsymbol{b}}\boldsymbol{x}_{\boldsymbol{i}}$.

In other words $\hat{\boldsymbol{a}}$ **and** $\hat{\boldsymbol{b}}$ are the solutions of the normal equations derived by minimising the square of errors i.e. $\sum_{i=1}^{n} \hat{\boldsymbol{\varepsilon}}_{\boldsymbol{i}}^{2} = \sum_{i=1}^{n} (\boldsymbol{y}_{\boldsymbol{i}} - \hat{\boldsymbol{a}} - \hat{\boldsymbol{b}}\boldsymbol{x}_{\boldsymbol{i}})^{2}$.

Thus the OLS estimates of the model parameters are:

$$\hat{\boldsymbol{a}} = \overline{\boldsymbol{y}} - \hat{\boldsymbol{b}}\overline{\boldsymbol{x}}$$

$$\hat{\boldsymbol{b}} = \frac{\sum_{i=1}^{n}(\boldsymbol{x}_{\boldsymbol{i}} - \overline{\boldsymbol{x}})(\boldsymbol{y}_{\boldsymbol{i}} - \overline{\boldsymbol{y}})}{\sum_{i=1}^{n}(\boldsymbol{x}_{\boldsymbol{i}} - \overline{\boldsymbol{x}})^{2}} = \frac{\boldsymbol{s}_{\boldsymbol{xy}}}{\boldsymbol{s}_{\boldsymbol{x}}^{2}} = \boldsymbol{r}_{\boldsymbol{xy}} \frac{\boldsymbol{s}_{\boldsymbol{y}}}{\boldsymbol{s}_{\boldsymbol{x}}}$$

Here,

- $\overline{\boldsymbol{y}}$ and $\overline{\boldsymbol{x}}$**:** average of $\mathbf{y_i}$ and $\mathbf{x_I}$ respectively
- $\boldsymbol{r}_{\boldsymbol{xy}}$**:** the sample correlation coefficient between **x** and **y**
- $\boldsymbol{s}_{\boldsymbol{y}}$ and $\boldsymbol{s}_{\boldsymbol{x}}$**:** the uncorrelated sample standard deviations of $\boldsymbol{x}$ and $\boldsymbol{y}$
- $\boldsymbol{s}_{\boldsymbol{xy}}$ and $\boldsymbol{s}_{\boldsymbol{x}}^{2}$**:** the sample covariance and variance respectively

Substituting the values of estimates of **a** and **b**, we have the desired prediction formula as:

$$\boldsymbol{y} = \overline{\boldsymbol{y}} + \boldsymbol{r}_{\boldsymbol{xy}} \frac{\boldsymbol{s}_{\boldsymbol{y}}}{\boldsymbol{s}_{\boldsymbol{x}}} (\boldsymbol{x}_{\boldsymbol{i}} - \overline{\boldsymbol{x}})$$

The above equation is called the **regression line of y on x**.

AN ILLUSTRATION:

To understand the concept of OLS linear regression, we consider the following problem:

The following table gives data on the GDP (Gross Domestic Product) deflator for domestic goods and the GDP deflator for imports for Singapore for the period 1968-1982.

Year	GDP deflator for domestic goods (Y)	GDP deflator for imports (X)
1968	1000	1000
1969	1023	1042
1970	1040	1092
1971	1087	1105
1972	1146	1110
1973	1285	1257
1974	1485	1749
1975	1521	1770
1976	1543	1889
1977	1567	1974
1978	1592	2015
1979	1714	2260
1980	1841	2621
1981	1959	2777
1982	2033	2735

Our problem is to find a relationship between the response variable **y** and the predictor **x**. We try to fir a straight line through these given observations. Let us say that the relationship between **y** and **x** is linear and of the form $\mathbf{y = a + b\ x + \varepsilon}$, where **a** and **b** are model parameters which are to be estimated by the method of least squares. In this case, we are provided with 15 pairs of values of **y** and **x**, the i^{th} pair being denoted by $\mathbf{(x_i,y_i)}$, i=1,2,....,15.

Thus the model in terms of the observation data: $\boldsymbol{y_i = a + bx_i + \varepsilon_i}$**.** $\boldsymbol{\varepsilon_i}$**'s** are error terms that are independently and identically distributed normal variables with mean zero and variance σ^2. The estimated values of the response variable are given as: $\boldsymbol{Y_i = \hat{a} + \hat{b}x_i}$**,** where $\boldsymbol{\hat{a}}$ and $\boldsymbol{\hat{b}}$ are the least square estimates of the model parameters.

To obtain the above mentioned estimates, we need to minimize the sum of squares of the error terms i.e. $\boldsymbol{Q = \sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} = \sum_{i=1}^{n}(y_i - \hat{a} - \hat{b}x_i)^2}$**.**

The desired values are obtained by solving the simultaneous equations, called the **normal equations** given by:

$$\boldsymbol{\frac{\partial Q}{\partial a} = 0}$$

$$\boldsymbol{\frac{\partial Q}{\partial b} = 0}$$

Solving the above two equations we get:

$\boldsymbol{\hat{a}}$ **= 516.1** and $\boldsymbol{\hat{b}}$ **= 0.534.**

Thus, our fitted regression line for the given data is given by:

$$\boldsymbol{Y = 516.1 + 0.534X}$$

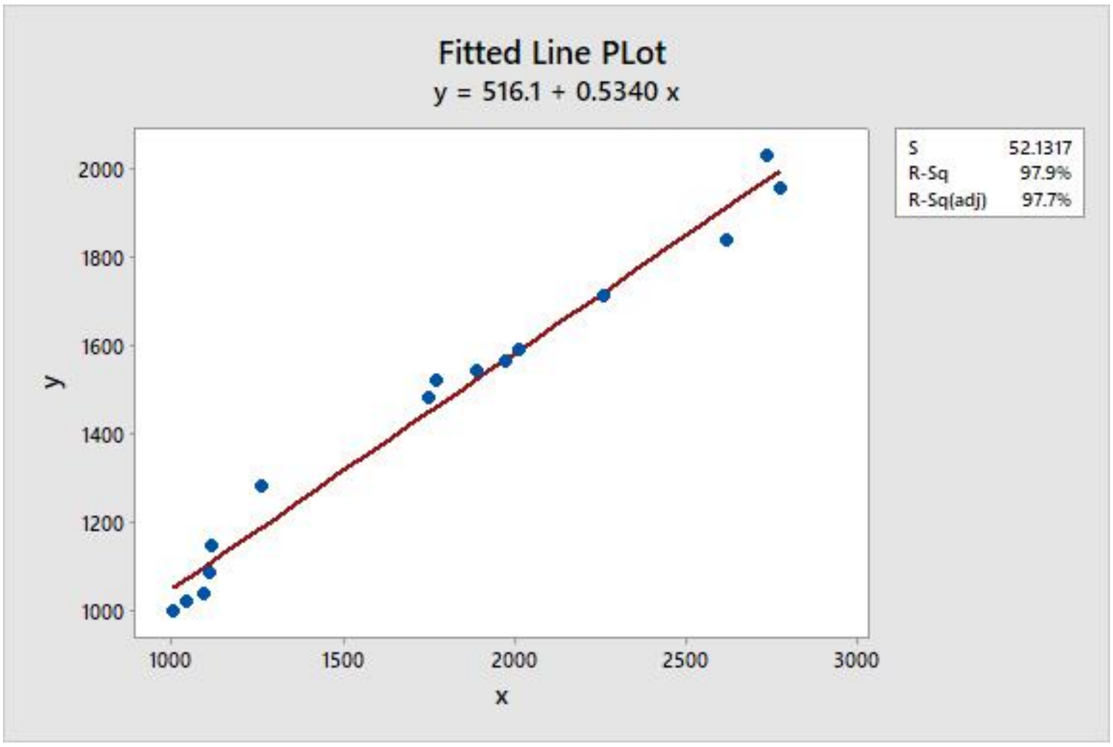


Figure: The corresponding fitted regression line of **y** on **x**.

CONCLUSION:

- We obtain the value of R^2 (coefficient of determination) as 97.9%. It implies that 97.9% of the variability in the observed values of the response variable **y** is explained by our assumed linear regression model. Thus we can say that the linear regression is a good fit.
- The correlation coefficient (**r**) between the variables **x** and **y** i.e. $\mathbf{r_{xy}}$ = 0.989. Since the value of **r** is very high, we can conclude that the variables **x** and **y** are highly positively correlated to each other.
- Suppose we are provided with a value the GDP deflator for imports i.e. a value on **x** for some other year, say **x** = 3000, and we are asked to estimate the value of GDP deflator for domestic goods i.e. **y**, we can simply plug in the given values in our fitted linear regression model. In this case, if **Y** is the estimated value of **y** for **x** = 3000, **Y** can be calculated as follows: **Y=516.1+ (0.534*3000) = 2118.1**. Thus the estimated value of the response for the given value of predictor is 2118.1.

In this previous section, we discussed about the OLS Linear Regression Theory and how to use this technique to obtain a mathematical form of dependence of a response variable on an independent variable. The similar approach can be extended over to polynomial regression as well as in cases of dependence of the response on more than one predictor.

However, it may happen that the basis of our applying this OLS technique i.e. the underlying assumptions is challenged.

What if the structure of the dependence i.e. f(x) is not a systematic one (linear, quadratic etc)? It may also be the case that the distribution of f(x) does not make it suitable for the OLS regression models. Also, the underlying covariance function may not be so simple.

In any of the above mention cases, it will be fruitless to apply OLS regression theory because the basic assumptions of the model are not satisfied.

To cope up with this challenge, we may use another type of regression which does not involve any known form of dependence, or in other words none of the predictors take predetermined forms with the response but are constructed according to the information derived from the data. Such a type of regression is called **Non-parametric Regression (NPR).**

There are different kinds of NPR models such as:

- Gaussian Process Regression (GPR) Models
- Kernel Regression Models
- Multiplicative Regression Models
- Local Regression Models etc.

In this project, we shall be only dealing with GPR or Gaussian Process Regression.

GAUSSIAN PROCESS REGRESSION

INTRODUCTION:

The simple linear regression model where the output is a linear combination of the inputs has been studied and used exclusively. Its main virtues are simplicity of implementation and interpretability. Its main drawback is that it only allows a limited flexibility: if the relationship between input and output cannot reasonably be approximated by a linear function, the model will give poor predictions.

These limitations can be tackled using Gaussian Process Regression (GPR). GPR has been proven to be a powerful and quite effective method for non-linear regression problems due to many exclusive properties. GPR method directly captures the model uncertainty or the uncertainty in predictions. Another very important advantage of this method is that we are able to add prior knowledge and specifications about the shape of the model by selecting different kernel functions; the method thus has the ability to capture a wide variety of behaviour through simple parameterizations and many others.

GPR is a kernel based non-parametric method which relies on appropriate selection of kernel and the hyperparameters involved. Our assumptions are prior judgements about the function of our interest are encapsulated in Kernels which define the closeness and similarity between two data points. Kernels are characterised by a set of elements called hyperparameters. After an appropriate kernel has been selected, their unknown hyperparameters need to be estimated from the data provided, which is also called the training data. Hyperparameters are generally estimated using maximum marginal likelihood functions, or more specifically by optimising the marginal likelihood function. However, marginal likelihood functions are not usually convex with respect to the hyperparameters, which means optimisation of the marginal likelihood function may yield local optima instead of global optima. A common approach to tackle this issue is to use multiple starting points randomly selected from a specific prior distribution and after convergence choose the optimised values with the largest marginal likelihood as the estimates. It is importance to know if the GPR under consideration is sensitive to the choice of prior distribution which is generally selected based on expert opinions and experiences. Studies regarding prior selection can help using Gaussian Process as a modelling tool.

Historical background: Prediction with Gaussian processes is not a very recent topic, specially in the field of time series analysis; the basic theory goes back at least as far as the work of Wiener [1949] and Kolmogorov [1941]. Kauritzen [1981] discussed relevant work by Danish astronomer T. N. Thiele dating from 1880. GP prediction is well known in geostatistics field (Matheron, 1973) where it is known as *kriging*, and meteorology (Thompson, 1956; Daley, 1991). Ripley [1981] and Cressie [1993] provide useful overviews of Gaussian process prediction in spatial statistics. Gradually it was realised that Gaussian process prediction could be used in a general regression context. O’Hagan [1978] presented the general theory, and applied it to a number of one-dimensiona; regression problems. Williams and Rasmussen [1996] described Gaussian process regression in machine learning context, and described optimisation of the parameters in the covariance function (Rasmussen, 1996).

GAUSSIAN PROCESS REGRESSION MODEL:

A **Gaussian Process** is a collection of random variables, which are characterised by consistent Gaussian distribution. Mathematically, for any set S, a Gaussian process (GP) on S is a set of random variables $(f_x, x \in S)$ such that, for any $n \in \mathbb{N}$ and $x_1, x_2, \ldots\ldots, x_n \in S$, $(f_{X_1}, f_{X_2}, \ldots., f_{X_n})$ is (multivariate) Gaussian. As a Gaussian distribution is specified by a mean vector and a covariance matrix, a GP is also fully characterised by a mean function and a covariance function which is also known as the *Kernel*.

Theorem of Gaussian Process: For any set S, any **mean function** $\boldsymbol{\mu}: \boldsymbol{S} \to \mathbb{R}$ and any **covariance function (or *kernel*)** $\boldsymbol{k}: \boldsymbol{S} \times \boldsymbol{S} \to \mathbb{R}$, $\exists$ a GP f(x) on S, $\ni$ $\boldsymbol{E[f(x)] = \mu}$, $\boldsymbol{Cov[f(x_s), f(x_t)] = k(x_s, x_t)}, \forall \mathbf{x}, ; \mathbf{x}_{\mathbf{s},}\mathbf{x}_{\mathbf{t}} \in \mathbf{S}$**.** It denotes that $\boldsymbol{f \sim GP(\mu, k)}$**.**

If we consider a regression problem: y = f(x) + ε, by Gaussian process method, the unknown function f is assumed to follow a $GP(\mu, k)$. We consider n pairs of observations $(x_1,y_1),\ldots,(x_n,y_n)$. From these data, we can apply GPR in the following model: $\underline{y} = f(\underline{X}) + \varepsilon$, where, $\underline{y} = (y_1, y_2, \ldots, y_n)^T$ denotes the response vector, $\underline{X} = (x_1, x_2, \ldots, x_n)^T$ are the inputs, and $\underline{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^T$ are independently and identically distributed Gaussian noise with mean 0 and variance $\sigma_n{}^2$. It implies that the set of functions, $f(\underline{X}) = (f_{X_1}, f_{X_2}, \ldots., f_{X_n})^T$ follow multivariate Gaussian distribution $(f_{X_1}, f_{X_2}, \ldots., f_{X_n})^T \sim N(\mu, K)$, where $\underline{\boldsymbol{\mu}} = [\boldsymbol{\mu}(\boldsymbol{x_1}), \ldots, \boldsymbol{\mu}(\boldsymbol{x_n})]^{\boldsymbol{T}}$ **is the mean vector** and **K is the n×n covariance matrix of which the (i,j)$^{\text{th}}$ element $\mathbf{K_{ij}=k(x_i,x_j)}$**.

Prediction with Noise-free Observations: We are usually not primarily interested in drawing random functions from the prior, but want to incorporate the knowledge that the training data provides about the function. Initially, we will consider the simple special case where the observations are noise free, that is we know $\{(x_i,f_i)|i=1,2,\ldots n\}$. The joint distribution of the training outputs, **f**, and the test outputs $\underline{f_*} = \left(f_{*_1}, f_{*_2}, \dots, f_{*_m}\right)^T$ at test locations $\underline{X_*} = (x_{n+1}, x_{n+2}, \dots, x_{n+m})^T$ according to the prior is:

$$\begin{pmatrix} \underline{f} \\ \underline{f_*} \end{pmatrix} \sim N(0, \begin{bmatrix} \mathrm{K(X,X)} & \mathrm{K(X,X_*)} \\ \mathrm{K(X_*,X)} & \mathrm{K(X_*,X_*)} \end{bmatrix}$$

If there are n training points and m test points, then K(X,X*) denotes the n×m matrix of the covariances evaluated at all pairs of training and test points, and similarly for the other entries K(X,X), K(X*,X*), K(X,X*) and K(X*,X). To get the posterior distribution over functions, we need to restrict this joint prior distribution to contain only those functions which agree with the observed data points. In probabilistic terms, this operation is simple, corresponding to *conditioning* the joint Gaussian prior distribution on the observations to give

$$\underline{f_*} | \left(X_*, X, \underline{f}\right) \sim N(K(X_*, X)K(X,X)^{-1}\underline{f}, K(X_*, X_*) - K(X_*, X)K(X,X)^{-1}K(X, X_*))$$

Function values f* (corresponding to test inputs X*) can be sampled from the joint posterior distribution by evaluating the mean and covariance matrix from the above conditional distribution and then generating samples.

Prediction with Noisy Observation: Generally, for more realistic modelling situations, we do not have access to function values themselves, but only noisy versions thereof y = f(x) + ε. Assuming additive independent identically distributed Gaussian noise ε with variance $\sigma_n{}^2$, the prior on the noisy observations become

$$cov\left(y_p, y_q\right) = k\left(x_p, x_q\right) + \sigma_n{}^2\delta_{pq} \qquad \text{or} \qquad Cov\left(\underline{y}\right) = K(X,X) + \sigma_n{}^2 I$$

Where δ_{pq} is a Kronecker delta which is “one” iff p = q and zero otherwise. It follows from the independence assumption of noise, that a **diagonal** matrix is added, in comparison to a noise-free case.

Now, to predict the function values $\underline{f_*} = \left(f_{*_1}, f_{*_2}, \dots, f_{*_m}\right)^T$ at the test locations $\underline{X_*} = (x_{n+1}, x_{n+2}, \dots, x_{n+m})^T$, the joint distribution of the observed target values and the function values at the test locations under the prior is given by,

$$\begin{pmatrix} \underline{y} \\ \underline{f_*} \end{pmatrix} \sim N(\begin{pmatrix} \mu(\underline{\mathrm{X}}) \\ \mu\left(\underline{\mathrm{X}_*}\right) \end{pmatrix}, \begin{bmatrix} \mathrm{K}(\mathrm{X},\mathrm{X}) + \sigma_n{}^2 I & \mathrm{K}(\mathrm{X},\mathrm{X}_*) \\ \mathrm{K}(\mathrm{X}_*,\mathrm{X}) & \mathrm{K}(\mathrm{X}_*,\mathrm{X}_*) \end{bmatrix}$$

Where $\mu(\underline{\mathrm{X}}) = \underline{\mu}$, $\mu\left(\underline{\mathrm{X}_*}\right) = [\mu(\mathrm{x}_{\mathrm{n+1}}), \dots, \mu(\mathrm{x}_{\mathrm{n+m}})]^{\mathrm{T}}$, K(X,X*) is a m×n matrix with the (i,j)th element $[K(X_*, X)]_{ij} = k(x_{n+i}, x_j)$ and $[K(X_*, X_*)]_{ij} = k(x_{n+i}, x_{n+j})$. Thus the predictive distribution is:

$$f_* | (X, y, X_*) \sim N(\hat{\mu}, \hat{\Sigma})$$

where $$\underline{\hat{\mu}} = K(X_*, X)^T (K(X,X) + \sigma_n{}^2 I)^{-1} (\underline{y} - \mu(\underline{X}))$$

and $$\hat{\Sigma} = K(X_*, X_*) - K(X_*, X)^T (K(X,X) + \sigma_n{}^2 I)^{-1} K(X_*, X)$$

In GPR method, the mean function $\underline{\mu(\mathrm{x})}$ is often assumed to be 0. The final form of predictive mean and variance can thus be given by:

$$\underline{\hat{\mu}} = K(X_*, X)^T (K(X,X) + \sigma_n{}^2 I)^{-1} \underline{y}$$

and $$\hat{\Sigma} = K(X_*, X_*) - K(X_*, X)^T (K(X,X) + \sigma_n{}^2 I)^{-1} K(X_*, X)$$

Kernel: We have seen that a covariance function is a crucial ingredient in a Gaussian process predictor, as it encodes our assumptions about the function which we wish to learn. From a slightly different viewpoint it is clear that in supervised learning the notion of similarity between the data points is crucial; it is a basic assumption that points with inputs **x** which are close are likely to have similar target values y, and thus training points that are near to a test point should be informative about the prediction at that point. Under Gaussian process view, it is the covariance function or the ***kernel*** that defines the nearness or similarity.

In non-parametric statistics, **kernel** is a weighting function used in estimation techniques to estimate density functions of random variables. In the most basic sense, kernels define the closeness or similarity between two data points (which we shall be denoting by **x** and **x'** respectively). The choice of Kernel thus has a profound impact on the performance of a GPR model.

Some commonly used kernels are listed below:

- **Powered Exponential Kernel:** It is a generalised version of the *square exponential kernel*. It is defined as: $\boldsymbol{k_{PE}(x,x') = \sigma_f{}^2.\exp(-\frac{(x-x')^p}{2l^p})}$, where p ∈ (0,2], $\boldsymbol{\sigma_f}$ is the output scale amplitude and is basically a *hyperparameter*, "l" denotes the length(/time) scale.
- **Radial basis Function (RBF) or Squared Exponential Kernel:** It is the most widely used kernel in GPR. The kernel is defined as: $\boldsymbol{k_{SE}(x,x') = \sigma_f{}^2.\exp(-\frac{(x-x')^2}{2l^2})}$, where the hyperparameters have same meaning as mentioned above.
- **Periodic Exponential-Sine Squared Kernel:** This kernel is used to model functions which exhibit periodic pattern. It is defined as: $\boldsymbol{k_{PER}(x,x') = \sigma_f{}^2.\exp(-\frac{2\sin(\pi\frac{(x-x')}{p})^2}{l^2})}$, where 'p' is the period of the function.

- **Rational Quadratic Kernel:** This kernel is equivalent to adding together many SE kernels with different length scales and is defined as $\boldsymbol{k_{PE}(x,x') = \sigma_f{}^2 . (1 + \frac{(x-x')^2}{2\alpha l^2})^{-\alpha}}$**,** where "α" determines the relative weighting of large-scale and small-scale variations. It is seen that when $\alpha \to \infty$, the RQ is identical to RBF.
- **Matern Kernel:** This kernel is commonly used to define the statistical covariance between measurements made at two points that are *d* units distant from each other. It is defined as $\boldsymbol{k_M = \sigma_f{}^2 \frac{2^{1-v}}{\Gamma(v)} \left(\sqrt{2v}\frac{d}{\rho}\right)^v K_v(\sqrt{2v}\frac{d}{\rho})}$**,** where **Γ** is the *gamma* function, K_v is the modified Bessel function of the second kind, "ρ" and "v" are non-negative parameters of covariance.

<u>Estimation of Hyperparameters</u>: In GPR method, hyperparameters involved in the kernel are needed to be estimated from the training data. A *hyperparameter* is a parameter whose value is set before the learning process begins. By contrast, the values of other parameters are derived via training.

According to the GP assumption, distribution of training outputs is given as:

$$\underline{y}|\underline{X},\underline{\theta}\sim N(0,\Sigma_\theta)$$

where, $\boldsymbol{\Sigma_\theta = K + \sigma_n{}^2 I}$ is the covariance matrix for the noisy targets $\underline{\mathbf{y}}$ and **K** is the covariance matrix for the noise-free latent **f**. $\underline{\boldsymbol{\theta}}$ is the set of unknown hyperparameters.

The **Negative log marginal Likelihood (NLML)** function is thus given by:

$$\gamma(\theta) = -\log p(\underline{\boldsymbol{y}}|\underline{X},\underline{\theta}) = \frac{1}{2}\underline{\boldsymbol{y}}^T{\Sigma_\theta}^{-1}\underline{\boldsymbol{y}} + \frac{1}{2}\log|\Sigma_\theta| + \frac{n}{2}log2\pi$$

Here, the three terms of the marginal likelihood have readily interpretable roles: $\frac{1}{2}\underline{\boldsymbol{y}}^T{\Sigma_\theta}^{-1}\underline{\boldsymbol{y}}$ is the only term involving the observed targets, and is called the **data-fit**. $\frac{1}{2}\log|\Sigma_\theta|$ is called the **complexity penalty** depending on the covariance function and the inputs. $\frac{n}{2}log2\pi$ is the **normalization constant**.

To set the hyperparameters by maximising the marginal likelihood, we seek the partial derivatives of the marginal likelihood w.r.t. the hyperparameters. Thus the required partial derivatives of NLML with respect to the hyperparameters are given by:

$$\frac{\partial\gamma(\theta)}{\partial\theta_j} = -\frac{\partial}{\partial\theta_j}\log p(\underline{\boldsymbol{y}}|\underline{X},\underline{\theta}) = -\frac{1}{2}\underline{\boldsymbol{y}}^T{\Sigma_\theta}^{-1}\frac{\partial\Sigma_\theta}{\partial\theta_j}{\Sigma_\theta}^{-1}\underline{\boldsymbol{y}} + \frac{1}{2}tr({\Sigma_\theta}^{-1}\frac{\partial\Sigma_\theta}{\partial\theta_j})$$

For many kernels, the likelihood function is not convex with respect to the hyperparameters, thus optimisation algorithm may converge to a local optima which does not yield results as good as in case for global optima.

Local optima may suffer sensitivity of initial hyperparameters.

A common strategy adopted for optimisation is the method of **Conjugate Gradient Method**. The optimisation is repeated using several initial values generated randomly from a simple prior distribution, which is often based expert opinions and experiences. Those values of hyperparameters are accepted for which the likelihood function yields the lowest value after convergence.

The **ALGORITHM** for hyperparameter estimation:

Given a prior distribution $p_0(\theta)$ and the number of iterations M,

1. Randomly choose an initial hyperparameter θ_0 from $p_0(\theta)$.
2. Numerically minimise $\gamma(\theta)$ using θ_0 as the starting value and obtain an estimate of the hyperparameter.
3. Repeat steps (1) and (2) for M times and select the estimate with the largest NLML as the optimal estimate.

Prediction is then made based on the optimal estimate of the hyperparameter and comparison is made on the basis of whether the estimate is adequate for the given problem and tried over for different priors.

Generally, the following methods are used to select initial hyperparameters:

- Select initial hyperparameters from Uniform (0,1).
- Put different prior distributions on initial hyperparameters, some choices of which are given below.

Prior Type	Sl. No	Form of the generic notation of the hyperparameters in a given kernel	Prior Distribution ($p_0(\theta)$)
Non-informative priors (Also called vague priors; selected with the assumption that they have slight or no influence in the inferences)	1.	$\boldsymbol{\theta_i}$	**Uniform(0,1)**
	2.	$\boldsymbol{log}(\boldsymbol{\theta_i})$	**Uniform(-1,1)**
	3.	$\boldsymbol{log}(\boldsymbol{\theta_i})$	**Uniform(-10,10)**
	4.	$\boldsymbol{\theta_i}$	***N*(0,1)**
	5.	$\boldsymbol{\frac{\pi}{\theta_i}}$	**Uniform(0,1)**
	6.	$\boldsymbol{log}(\boldsymbol{\frac{\pi}{\theta_i}})$	**Uniform(-5,5)**
Data-dominated priors (incorporated with some information inferred from training data)	7.	$\boldsymbol{\theta_i}$	**Uniform(0,Nyq)**
	8.	$\boldsymbol{\frac{1}{\theta_i}}$	***TN*(MaxI)**
	9.	$\boldsymbol{\frac{\pi}{\theta_i}}$	**Uniform**($\boldsymbol{\frac{\pi}{MaxI}}, \boldsymbol{\pi Nyq}$)

Where, **Nyq**: Nyquist frequency i.e. half the largest interval between input points if the data are not regularly sampled.

MaxI: Truncated Normal distribution with mean proportional to the maximal range of the inputs.

Priors 1, 2 and 3 are used for SE kernel. Priors 1, 5, 6, 7 and 9 are suitable for parameter period of PER/LP kernel. For SM (Spectral Mixture) kernel, there are two parameters from priors 5, 6, 7, 9 and prior 1, 8 respectively.

ILLUSTRATIONS OF COMPARISON BETWWEN GPR AND OLS REGRESSION MODELS

(*With the help of simulation of data*)

Let $\mathbf{x_i}$'s be predictor variables which are non-stochastic in nature, and $\mathbf{y_i}$'s be the response variables. We hence consider the training set of observations D:{(x_i,y_i); i=1(1)n}.

We consider the model: y = f(x) + ε , where $\varepsilon \sim N(0, {\sigma_n}^2), {\sigma_n}^2 > 0$.

We will be considering four sets of data and will be applying OLS and GPR regression models in each of those. In case of OLS model, the form of f(x) will be known, and correspondingly in case of GPR, we will be assuming that $f(x) \sim GP(m(x) = 0, k(x, x'))$. Now, since GPR model evolves with the data, it is expected that it will naturally be better fit for most observational data. However we will be explicitly testing this idea by sampling data from some **normal** and some **non-normal** setups including distributions which have some unique properties.

For all out tests, we will be taking a random sample of size 100 from Uniform(0,1) distribution as values of the predictor variable x.

NOTE: We will be comparing the two methods of regression modelling on the basis of the residual errors committed for choosing that specific modelling technique. In case of GPR regression model, we will be noting the **train error** (i.e. error of prediction) and for OLS regression model, we will be noting the **standard error** and make a relative comparison on the efficacy of the two modelling techniques based on the values of these two errors committed.

<u>CASE I</u>: We consider $\boldsymbol{f(x) \sim N(0.4 + 0.6x, 1)}$, thus considering a homoscedastic normal setup. We also consider error terms simultaneously and simulate 100 data points 4 times and name them y1_i (y denoting the response variable), i=1(1)4.

We plot y1_i against x using OLS and GPR regression models and they yield the following results:

Fit no.	Training data	Training error (GPR)	Standard Error (OLS)
1	(y1_1, x)	1.419	0.802
2	(y1_2, x)	1.559	0.714
3	(y1_3, x)	1.235	0.823
4	(y1_4, x)	1.206	0.883

The respective plots for GPR and OLS regression models based on the first set of training data i.e. (y1_1, x) are provided below:

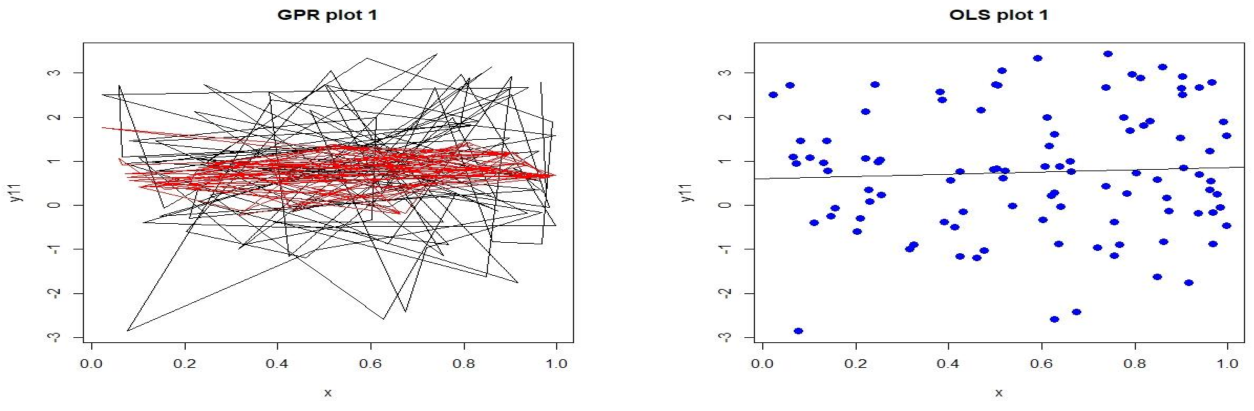


Figure: GPR and OLS regression plots for $\boldsymbol{f(x) \sim N(0.4 + 0.6x, 1)}$

CASE II: We consider $\boldsymbol{f(x) \sim N(0.4+0.6x, e^{x})}$, thus considering a heteroscedastic normal setup. We also consider error terms simultaneously and simulate 100 data points 4 times and name them $y2_i$ (y denoting the response variable), i=1(1)4.

We plot $y2_i$ against x using OLS and GPR regression models and they yield the following results:

Fit no.	Training data	Training error (GPR)	Standard Error (OLS)
1	($y2_1$, x)	1.872	0.789
2	($y2_2$, x)	1.859	0.898
3	($y2_3$, x)	1.079	0.924
4	($y2_4$, x)	1.152	0.797

The respective plots for GPR and OLS regression models based on the first set of training data i.e. ($y2_1$, x) are provided below:

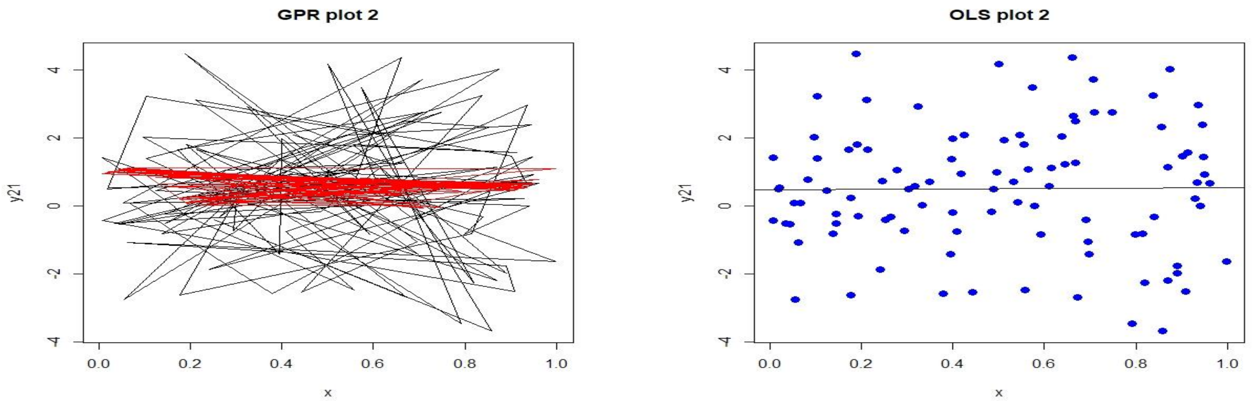


Figure: GPR and OLS regression plots for $\boldsymbol{f(x) \sim N(0.4+0.6x, e^{x})}$

<u>CASE II</u>: We consider $\boldsymbol{f}(\boldsymbol{x})\sim\boldsymbol{C}(\mathbf{0},\mathbf{1})$, thus considering a non-normal setup which characteristically possess no moments. We also consider error terms simultaneously and simulate 100 data points 4 times and name them $y3_i$ (y denoting the response variable), i=1(1)4.

We plot $y3_i$ against x using OLS and GPR regression models and they yield the following results:

Fit no.	Training data	Training error (GPR)	Standard Error (OLS)
1	($y3_1$, x)	0.923	6.301
2	($y3_2$, x)	0.821	7.689
3	($y3_3$, x)	0.756	32.07
4	($y3_4$, x)	0.743	8.104

The respective plots for GPR and OLS regression models based on the first set of training data i.e. ($y3_1$, x) are provided below:

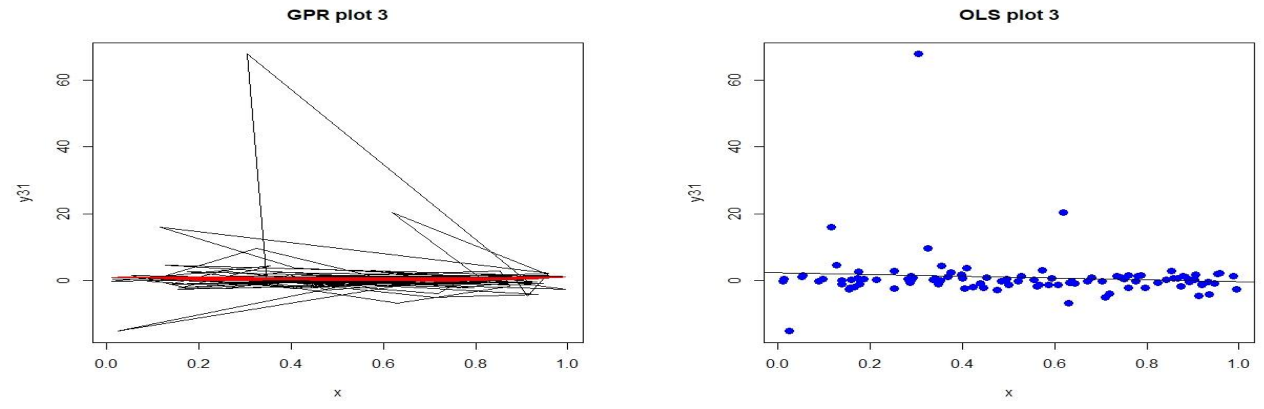


Figure: GPR and OLS regression plots for $\boldsymbol{f}(\boldsymbol{x})\sim\boldsymbol{C}(\mathbf{0},\mathbf{1})$

CASE IV: We consider $\boldsymbol{f(x)} \sim \boldsymbol{\chi^2_{16}}$, thus considering a non-normal setup. We also consider error terms simultaneously and simulate 100 data points 4 times and name them y4$_i$ (y denoting the response variable), i=1(1)4.

We plot y4$_i$ against x using OLS and GPR regression models and they yield the following results:

Fit no.	Training data	Training error (GPR)	Standard Error (OLS)
1	(y4$_1$, x)	0.808	4.997
2	(y4$_2$, x)	0.854	5.736
3	(y4$_3$, x)	0.884	6.029
4	(y4$_4$, x)	0.792	5.135

The respective plots for GPR and OLS regression models based on the first set of training data i.e. (y4$_1$, x) are provided below:

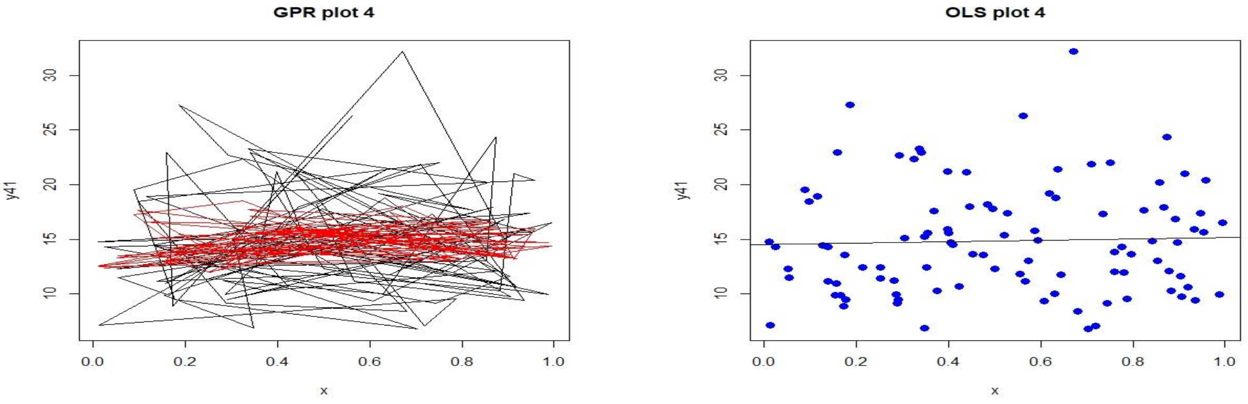


Figure: GPR and OLS regression plots for $\boldsymbol{f(x)} \sim \boldsymbol{\chi^2_{16}}$

After fitting OLS and GPR models in each of the above setups, we observe the following:

- The train errors (for GPR) are greater than the standard errors in cases I and II. We notice that those setups feature f(x) which follows normal distribution (with equal or unequal variances). We can thus argue, based on our observations, that OLS regression technique is more efficient in explain the observed response variable than the fit obtained by GPR method, based on the values of their prediction errors.
- In case III, we have considered a Cauchy distribution, which is characterised by the non-existence of its moments, which indicates *a priori* the experiment that OLS regression models may not yield proper fits to the data. This hypothesis is tested true as we obtain large standard error values for each experimental data taken. On the other hand, in GPR modelling, we obtain train errors which are significantly lower than the standard errors obtained in case of OLS models, clearly indicating that the fitting the data derived from a Cauchy distribution by applying GPR models is more efficient than that of fitting with OLS models.
- In case IV, $f(x) \sim \boldsymbol{\chi}^2_{16}$. In this case we see that the train errors are lower than the standard errors. But the supremacy in efficiency of fitting the data by GPR over fitting by OLS technique is not as pronounced as that in case of Cauchy distribution in case III.
- **Overall, we can conclude solely based on our experimental observations that OLS regression gives better fits than GPR models in case of normal distributions. GPR models, on the other hand, give better fits in case on non-normal distributions.**

REAL LIFE ILLUSTRATION

To further understand the concepts of GPR modelling and how it is different from the traditional OLS method of regression, we will be applying both the methods on a real life dataset.

Here we have considered the dataset of “Gross Domestic Product (GDP) of India from 1960 to 2019”. The data is extracted from “World Bank accounts data, and OCED National Accounts data files”.

GDP at purchaser’s prices is the sum of gross value added by all resident producers in the economy plus any product taxes and minus any subsidies not included in the value of the products. It is calculated without making deductions for depreciation of fabricated assets or for depletion and degradation of natural resources. Data are in current US dollars ($). Dollar figures are converted from the Indian currency using single year official exchange rates. Data is adjusted for inflation.

The data is basically a time series data of 60 observations, yearly from 1960 to 2019.

It is a suitable example to illustrate the concept of kernel selection and hyperparameter optimization using the marginalization technique of log marginal likelihood. We will be modelling the GDP figures of India as a function of time (in years) using GPR and OLS regression models.

To continue, we need to discuss the characteristics of a typical time series data. The different components of a time series data are:

- Trend
- Seasonal
- Cyclical
- Irregular

In our data, we won’t be dealing with the seasonal component since ours’ is a yearly data which disregards any “seasonal” or periodic fluctuations. We will also be ignoring the cyclical component due to computational and theoretical complexities.

For sake of simplicity, we have divided the GDP figures by a factor of 10^{12}. Thus the response variable in our setup denotes GDP of India in trillions of USD.

<u>*Specification of appropriate Kernels*</u>:

We now present the GDP figures in a graphical form to highlight its features.

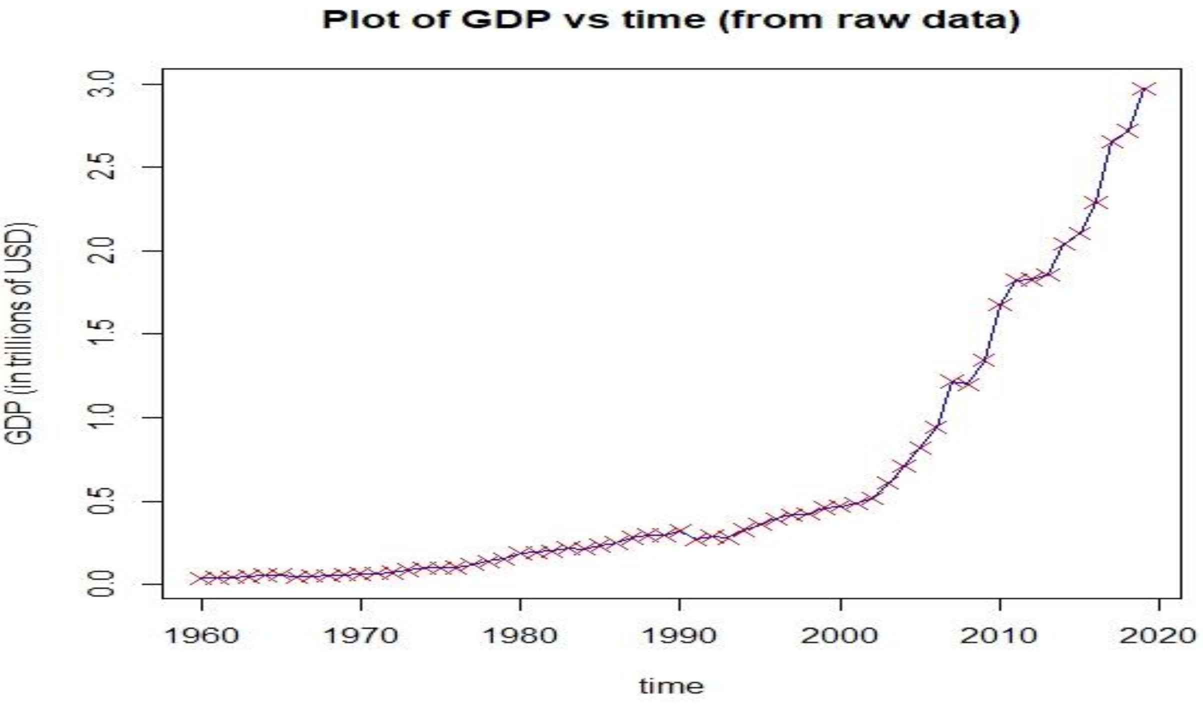


Figure 1: **Plot of GDP of India over time**

A long term rising trend and some smaller irregularities are immediately apparent. We will thus find appropriate Kernels which would explain each of these components individually. Our final motive is to find a combined covariance function that takes care of these individual properties. This is meant to illustrate the power and flexibility of the Gaussian process framework.

1. **Trend:** A long term, smooth rising trend is best explained by **Powered Exponential Kernel**. The PE kernel enforces the trend component to be smooth if it has a large length scale. The length scale and the amplitude are the free hyperparameters.
2. **Irregular:** Smaller, medium term irregularities are to be explained by **Rational Quadratic Kernel**, whose length scale and the shape parameter determines the diffuseness of the length scales. The length scale, amplitude and the shape parameters are the free hyperparameters. We could have used RBF kernel but it turns out that the Rational Quadratic works better (gives higher marginal likelihood), as it can accommodate several length scales.

Model structure and assumptions:

Let S be the set of observations in the dataset then any mean function $\mu: S \rightarrow \mathbb{R}$ and any covariance function (or *kernel*) $k: S \times S \rightarrow \mathbb{R}$, ∃ a GP f(x) on S, ∋ $E[f(x)] = \mu$, $Cov[f(x_s), f(x_t)] = k(x_s, x_t)$, ∀x, ; $x_{s,}x_t$ ∈ S. It denotes that $f \sim GP(\mu, k)$.

Let the regression model be y = f(x) + ε. By Gaussian process method the unknown function f(x) is assumed to follow a $GP(\mu, k)$. Given 60 pairs of observations $(x_1,y_1),...,(x_n,y_n)$, we have $\underline{y} = f(\underline{x}) + \underline{\epsilon}$, where $\underline{y} = (y_1, y_2, \dots, y_n)^T$ are the outputs denoting GDP (in trillions of USD), $\underline{x} = (x_1, x_2, \dots, x_n)^T$ denotes the time (in years), and $\underline{\varepsilon} = (\varepsilon_1, \varepsilon, \dots, \varepsilon_n)^T$ are the independently and identically distributed Gaussian noise with mean 0 and variance σ_n^2.

We consider the mean function μ(x) to be a linear trend line. It can take values of either 0 or 1, denoting zero mean or a constant mean.

Implementation of GPR using different kernels

1. We will first fit the data using the powered exponential kernel as the covariance function.

   - Hyperparameter estimation: To estimate the hyperparameters of the powered exponential kernel, we have to optimise the negative log marginal likelihood (nlml) by applying the optimisation algorithm called Conjugate Gradient Method. We first choose the prior of the hyperparameters as Uniform(0,1).

     The final estimates of the hyperparameters are $\hat{l} = 3.64$ and $\widehat{\sigma_f} = 1.07$. The optimised nlml is 78.266171. The white noise of the process is 0.2. We observe that as we move towards the final optimisation, the magnitude of white noise decreases gradually.

   - Plot: The predictive distribution is $N(\hat{\mu}, \widehat{\sum})$, where the corresponding mean in *b1$pred.mean* and the standard deviation is *b1$pred.sd*, where "b1" is the predictive or the posterior distribution. We thus plot b1 to plot the predictions and their 95% predictive interval. (For R codes refer to Appendix)

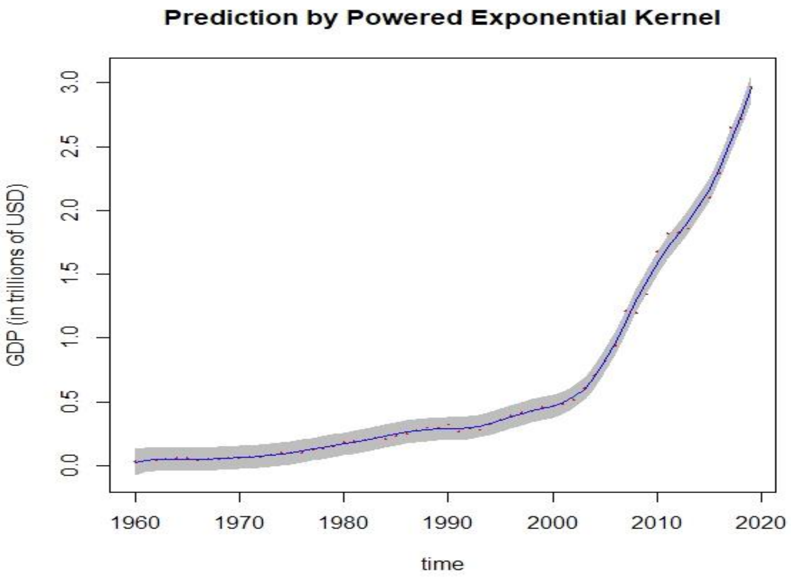


Figure 2: GDP vs time data: raw data (plotted in red), fitted line (in blue) with the 95% predictive interval.

2. We then fit the data using rational quadratic kernel as the covariance function.

- Hyperparameter estimation: We proceed performing the method of optimisation in the same manner as in the previous process.

  The final estimates of the hyperparameters are $\hat{l} = 4.043698$, $\widehat{\sigma_f} = 0.1636531$ and $\hat{\alpha} = 0.9055799$. The value of optimised nlml=106.38535. The white noise due to this process is 1.5.

- Plot: The predictive distribution is $N(\hat{\mu}, \widehat{\sum})$, where the corresponding mean in *b2$pred.mean* and the standard deviation is *b2$pred.sd*, where "b2" is the predictive or the posterior distribution. We thus plot b2 to plot the predictions and their 95% predictive interval. (For R codes refer to Appendix)

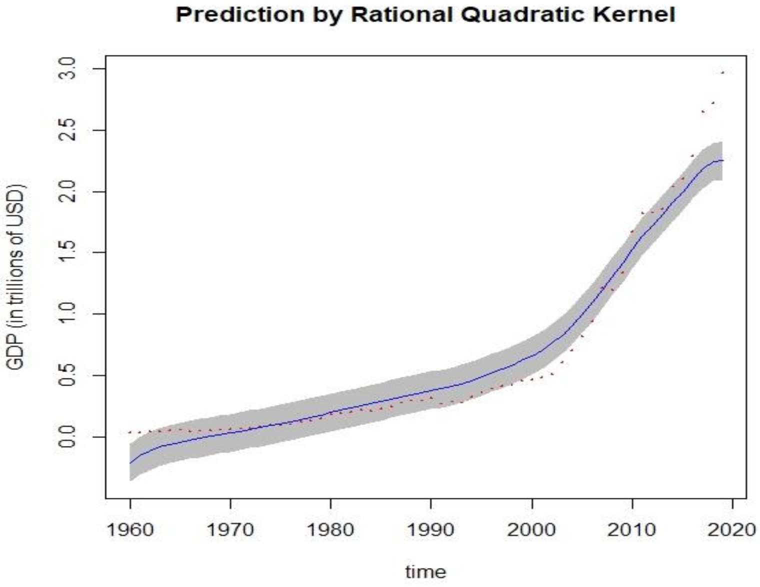


Figure 3: GDP vs time data: raw data (plotted in red), fitted line (in blue) with the 95% predictive interval.

3. We now define a new GPR model, in which the covariance function combines the two kernels used earlier i.e. squared exponential and rational quadratic kernels.

- Hyperparameter estimation: We use the three kernels by simply adding them (by performing matrix addition) and perform the optimisation of nlml.

  The hyperparameter estimates for SE kernel component are $\hat{l} = 2.0950755$ and $\widehat{\sigma_f} = 0.04150447$. The hyperparameter estimates for RQ kernel component are $\hat{l} = 1.519364$, $\widehat{\sigma_f} = 0.2738827$ and $\hat{\alpha} = 0.37019$. The value of optimised nlml is 91.154450. The white noise in this process is 0.5.

- Plot: The predictive distribution is $N(\hat{\mu}, \widehat{\sum})$, where the corresponding mean in *b3$pred.mean* and the standard deviation is *b3$pred.sd*, where “b3” is the predictive or the posterior distribution. We thus plot b3 to plot the predictions and their 95% predictive interval. (For R codes refer to Appendix)

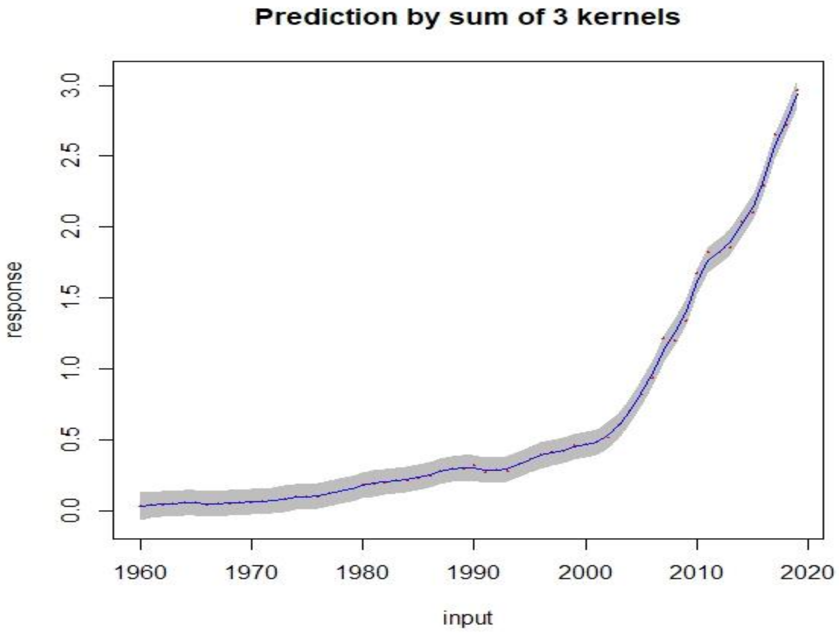


Figure 4: GDP vs time data: raw data (plotted in red), fitted line (in blue) with the 95% predictive interval using combination of the 2 kernels.

Implementation of OLS linear model

We consider the given data and try to analyse the relationship between GDP and time. For sake of simplicity, we will redefine the time points and change them from 1960-2019 to 1-60. Clearly, the GDP (in trillions of USD) is the response variable and time is the predictor. We will try to fit a line through the given observations.

We consider the dependence of GDP(in trillions of USD) which is being denoted by **y** on time (being denoted by **x**) is of the form: $y = a + bx + \varepsilon$. Here "a" and "b" are model parameters to be estimated by the method of least squares.

We are provided with n = 60 pairs of observations, the ith pair being denoted by (x_i, y_i), i=1(1)60.

Thus the model in terms of the observation data: $y_i = a + bx_i + \varepsilon_i$. ε_i's are error terms that are independently and identically distributed normal variables with mean zero and variance σ^2. The estimated values of the response variable are given as: $Y_i = \hat{a} + \hat{b}x_i$, where $\hat{a}$ and $\hat{b}$ are the least square estimates of the model parameters.

To obtain the above mentioned estimates, we need to minimize the sum of squares of the error terms i.e. $S^2 = \sum_{i=1}^{60}(y_i - Y_i)^2 = \sum_{i=1}^{60} \hat{\varepsilon}_i{}^2 = \sum_{i=1}^{60}(y_i - \hat{a} - \hat{b}x_i)^2$.

We obtain a set of equations called the normal equations. The final estimates of model parameter are obtained by solving these normal equations, the method of which is explained earlier.

We get, $\hat{a} = -0.51871$ and $\hat{b} = 0.03758$.

The fitted linear regression line is given by:

$$y = -0.51871 + 0.03758x$$

The plot of the fitted line is given below:

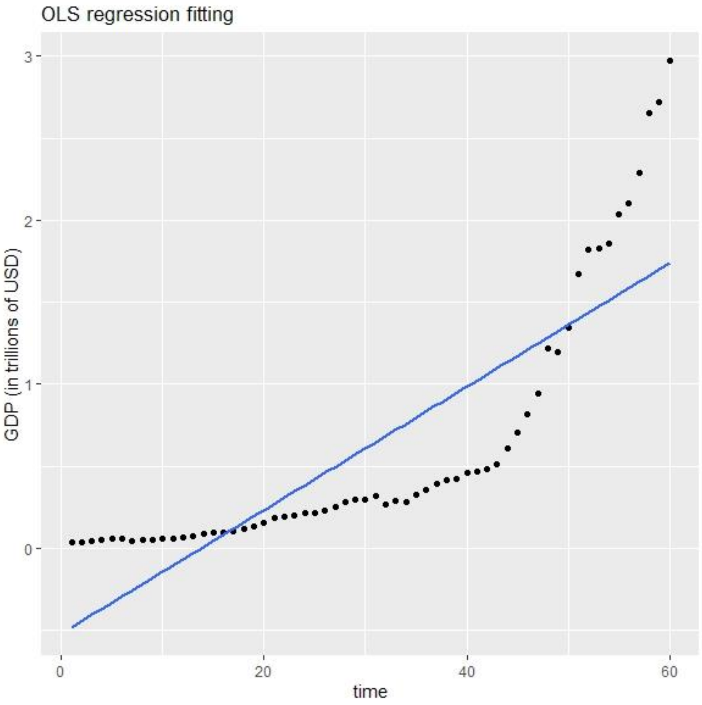


Figure 5: Least Squares Linear Regression line with scatter plot

The residuals obtained post application of the OLS method are plotted against time.

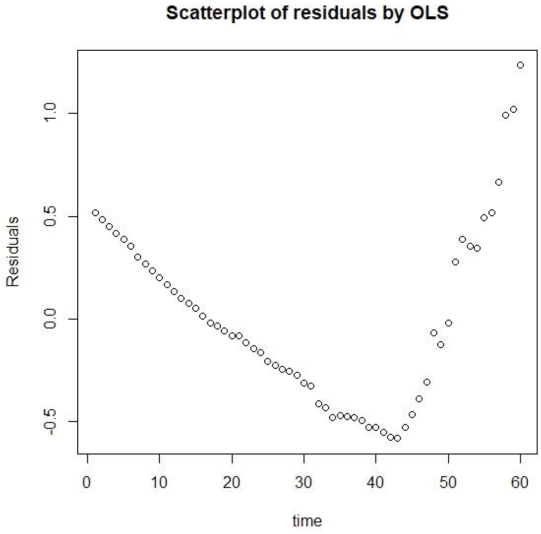


Figure 6: Scatter plot of residuals against time

Conclusion:

There is no standard device for comparison of the two models. Hence both are interpreted separately.

- *GPR model*: We can clearly see, with the help of the GPR model (both figures 1 and 4), that the trend is rising. Further due to incorporation of both the kernels, trend component has been smoothed and the GPR function explains the data very efficiently owing to the value of white noise of the process being 0.5.
- *OLS model*: We obtain the value of coefficient of determination i.e. R^2 as 69.62% indicating that linear regression line does not explain the observed response very efficiently. It is a moderately good fit. The value of correlation coefficient comes up as 0.834386. An interpretation of the estimated slope is that each year GDP (in trillions of USD) increases by 0.03758 points (in the same scale) with a standard error of 0.00322 points (also in the same scale). Residual plot shows pattern indicating fit is not efficient.

Predictions:

- *Using GPR models*: Figures 2, 3 and 4 indicate the existence of **prediction regions**. They are prediction bands implying the 95% predictive intervals. The 95% predictive interval is given by *b3$pred.mean ± 1.96* b3$pred.sd* for the final model (Refer to R codes in Appendix). The model shows increasing confident predictions with time.
- *Using OLS models*: Let us try to predict the GDP (in trillions of USD) for time point 41 corresponding to the year 2000.

  We use the fitted equation i.e. $y = -0.51871 + 0.03758x$ for prediction, where x = 41. We get the corresponding $\widehat{Y_{41}} = 1.02207$. Thus the GDP of India in the year 2000 is predicted to be \$1.022×10^{11}. However, the actual figure of GDP in the year 2000 was \$4.683×10^{11}. This indicates that linear regression method is not appropriate for the data.

ADVANTAGES AND DISADVANTAGES

Advantages of GPR models:

- Gaussian process directly captures model uncertainties. The model actually given the distribution of the prediction value and not just unique prediction values.
- GPR models can model arbitrary complex systems, provided a large number of data is available.
- The main advantage of GP is that prior knowledge and specifications about the shape of the models can be incorporated by the process of kernel selection.
- GPs come up with a neat way to tune hyper-parameters by maximising the marginal likelihood. This tends to give very good fits without the need for cross-validation.

Advantages of OLS models:

- LS estimates correspond to the maximum likelihood solution. This allows us to obtain the guarantees of maximum likelihood estimates (consistency, asymptotic normality etc). This then allows us to build hypothesis tests and obtain confidence intervals for estimated regression coefficients.
- It provides a consistent theory and methods for regression, ANOVA and ANOCOVA.
- These models produce solutions which are easy to interpret.
- It helps us measure the efficiency of the model in explaining the observed response variance and hence helps us to analyse the relationship between two or more variables.
- By Gauss-Markov theorem, if the system being studied is truly linear with additive uncorrelated noise with constant variance, the LS regression line is the Best Linear Unbiased Estimator of the true coefficients.

Disadvantages of GPR models:

- The model assumes a Gaussian uncertainty on the y-values which may not be true in its actuality (for example, if output values are bounded, then the assumption of Gaussian uncertainty is not efficient in the modelling).
- The process is quite expensive.
- The process involves time consuming calculations.
- The modelling is not that efficient in case of a very low or a very large number of observations.

Disadvantages of OLS models:

- The models in this case can give biased outcomes in presence of outliers in the training data.
- It is not a good choice if the model is actually non-linear.
- Gives poor predictions if previously assumed independent variables are actually correlated.
- Performs badly if too many variables are considered.
- It is designed in a way to minimise noise in the response variable but does not consider the noise in the predictor variable.

CONCLUSION

Through this project, I have tried to focus on the applications of GPR and OLS regression models and have also tried to compare them, using both simulations and applying the models on a real life dataset. As can be clearly inferred, GPR models are in essence more flexible in modelling any kind of dataset available. However, under certain conditions OLS models clearly yield better fits. The main feature of GPR modelling is that we can incorporate prior knowledge and specifications before fitting the data in the form of kernels, which is perhaps a drawback in case of OLS models which is constrained by a set of assumptions. However, OLS model is more easily interpretable than GPR model.

It is clear from this project that the OLS and GPR regression models have a vast and quite different field of applications, each of which is efficient in some special circumstances.

FURTHER WORKS

Some notable research papers and works on OLS and GPR models:

- *A unifying view of sparse approximate Gaussian Process Regression*- J Quiñonero-Candela, CE Rasmussen [2005].
- *Sparsely greedy Gaussian Process Regression*- AJ Smola, PL Bartlett [2001].
- *Sparse spectrum Gaussian Process Regression*- M Lázaro-Gredilla [2010].
- *Nonstationary covariance functions for Gaussian Process Regression*- CJ Paciorek, MJ Schervish [2004].
- *Single image super-resolution using Gaussian Process Regression*- H He, WC Siu [2011].
- *Automatic Gait Optimisaton with Gaussian Process Regression*- DJ Lizotte, T Wang, MH Bowling, D Schuurmans [2007].
- *Forward Modelling of Ground Penetrating Radar for the Reconstruction of Models Redponse Profiles using Synthetic Data*.
- *Estimating replicate time shifts using Gaussian Process Regression*.
- *How do patient characteristics influence informal payments for inpatient and outpatient health care in Albania: Results of logit and OLS models using Albanian LSMS 2005.*
- *Runoff forecasting using RBF networks with OLS algorithm*- DAK Fernando, AW Jayawardena [1998].

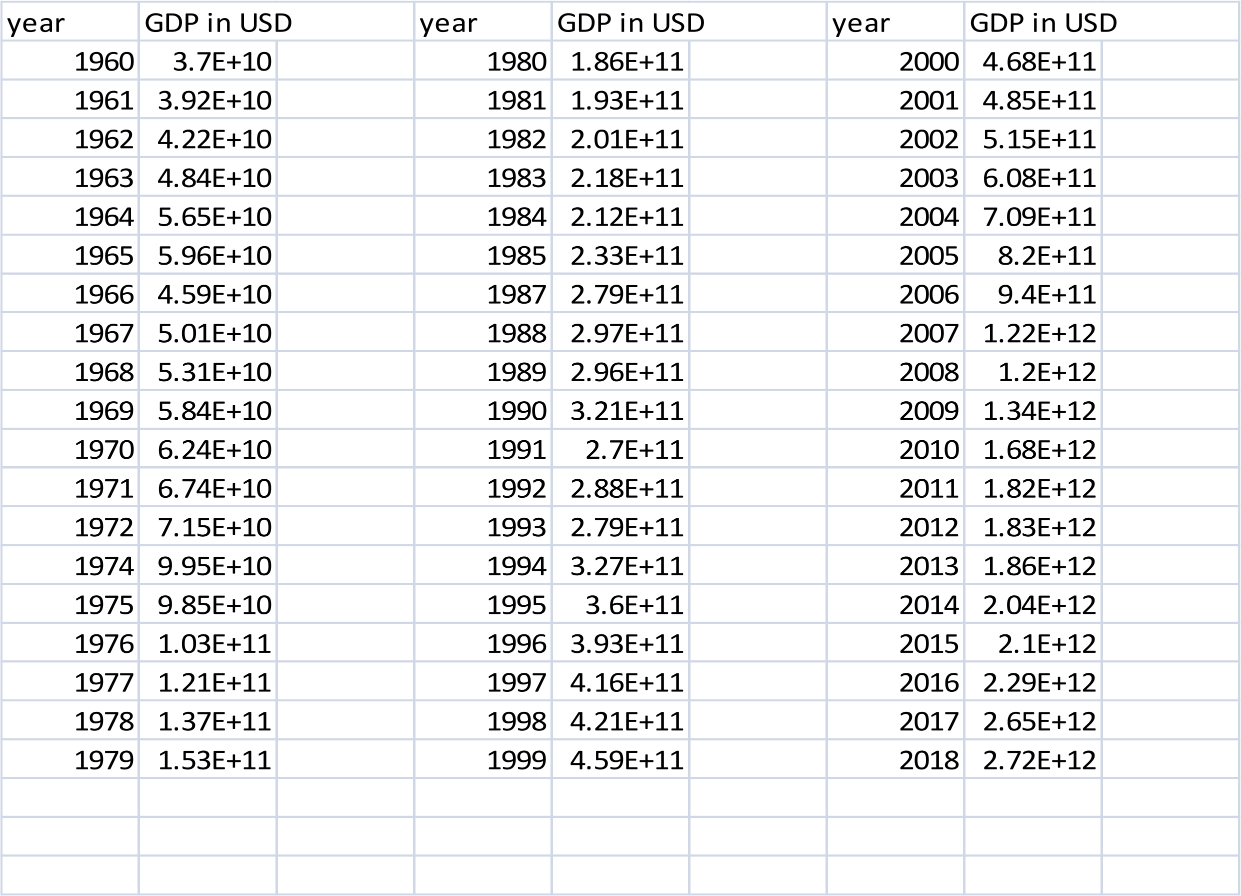
BIBLIOGRAPHY

Mentioned below are the sources from where I have taken invaluable guidance and study materials:

- https://en.wikipedia.org/wiki/Nonparametric_regression
- https://en.wikipedia.org/wiki/Kernel_regression
- https://en.wikipedia.org/wiki/Positive-definite_kernel
- https://www.colorado.edu/lab/lisa/services/short-courses/parametric-versus-seminonparametric-regression-models
- *Introduction to Gaussian Process Regression*- Hanna M. Wallah [2005].
- http://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/R/R5_Correlation-Regression/R5_Correlation-Regression4.html
- https://towardsdatascience.com/quick-start-to-gaussian-process-regression-36d838810319
- https://www.researchgate.net/post/What_are_the_advantages_of_using_Gaussian_Process_Models_against_Neural_Networks
- *A tutorial on Gaussian Process Regression with a focus on exploration-exploitations scenarios*- Eric Schulz, Maarten Speekenbrink, Andreas Krause.
- *Gaussian Processes for Machine Learning, MIT Press, 2006*- C. E. Rasmussen, C. K. I. Williams.
- *Gaussian Process for Regression*: *A Quick Introduction*- M Ebden [2008].
- *How priors of initial hyperparameters affect Gaussian Process regression model*: Zexun Chen [2015].
- *Gaussian Process Models for Robust Regression, Classification and Reinforcement Learning*.
- *Gaussian Process Function Data Analysis- R Package GPFDA*- Jian Qing Shi, Yafeng Chen.
- https://data.worldbank.org/indicator/NY.GDP.MKTP.CD?locations=IN

Appendix:

The data on GDP (in USD) vs time for India:



year	GDP in USD	year	GDP in USD	year	GDP in USD
1960	3.7E+10	1980	1.86E+11	2000	4.68E+11
1961	3.92E+10	1981	1.93E+11	2001	4.85E+11
1962	4.22E+10	1982	2.01E+11	2002	5.15E+11
1963	4.84E+10	1983	2.18E+11	2003	6.08E+11
1964	5.65E+10	1984	2.12E+11	2004	7.09E+11
1965	5.96E+10	1985	2.33E+11	2005	8.2E+11
1966	4.59E+10	1987	2.79E+11	2006	9.4E+11
1967	5.01E+10	1988	2.97E+11	2007	1.22E+12
1968	5.31E+10	1989	2.96E+11	2008	1.2E+12
1969	5.84E+10	1990	3.21E+11	2009	1.34E+12
1970	6.24E+10	1991	2.7E+11	2010	1.68E+12
1971	6.74E+10	1992	2.88E+11	2011	1.82E+12
1972	7.15E+10	1993	2.79E+11	2012	1.83E+12
1974	9.95E+10	1994	3.27E+11	2013	1.86E+12
1975	9.85E+10	1995	3.6E+11	2014	2.04E+12
1976	1.03E+11	1996	3.93E+11	2015	2.1E+12
1977	1.21E+11	1997	4.16E+11	2016	2.29E+12
1978	1.37E+11	1998	4.21E+11	2017	2.65E+12
1979	1.53E+11	1999	4.59E+11	2018	2.72E+12

R codes for “Real life Illustraton”

```
> library(GPFDA)
> library(ggplot2)
> library(Matrix)
> require(MASS)
```

Loading Data

```
> gdp1=c(37029883875,39232435784,42161481859,48421923459,56480289941,...
```

Storing data into matrix

```
> gdp2=gdp1/1e+12
> y=data.matrix(gdp2)
> x=1960:2019
> mat=cbind(y,x)
> mat=na.omit(mat)
> X <- as.matrix(mat[,2])
> Y <- as.matrix(mat[,1])
> x <- as.matrix(1960:2019)
```

First Kernel

```
> system.time(a1 <- gpr(as.matrix(X),as.matrix(Y),c('pow.ex'),trace=2))

          title: -likelihood:    pow.ex.v    pow.ex.w    vv

  0:     78.266171: -0.00438975 -0.0103399 0.200000

  2:    -54.091070: -3.01283 -3.33061 -5.92089

  4:    -66.539667: -1.70239 -3.30502 -6.58794

  6:    -68.072900: -1.13623 -3.63433 -6.21146

  8:    -68.229625: -1.11939 -3.63431 -6.34247

 10:    -68.233991: -1.09101 -3.64607 -6.32928

 12:    -68.234659: -1.07250 -3.64949 -6.33141

 14:    -68.234659: -1.07283 -3.64926 -6.33145

> system.time(b1 <- gppredict(a1,Data.new=as.matrix(x)))
```

Plotting Figure 2

```
> upper=b1$pred.mean+1.96*b1$pred.sd
> lower=b1$pred.mean-1.96*b1$pred.sd
>plot(-100,-100,col=0,xlim=range(X,x),ylim=range(upper,lower,Y),main="Prediction Powered Exponential Kernel",xlab="time",ylab="GDP (in trillions of USD)")
> polygon(c(x,rev(x)),c(upper,rev(lower)),col="grey",border=NA)
> lines(x,b1$pred.mean,col=4,lwd=0.8)
> points(X,Y,pch=2,col=2,cex=0.1)
```

Second Kernel

```
> system.time(a2 <- gpr(as.matrix(X),as.matrix(Y),c('rat.qu'),trace=2))

      title: -likelihood:    pow.ex.v    pow.ex.w    vv

  0:    94.830230: 0.00335929 -0.00619596  1.00000

  2:   -26.560037: -2.40622 -1.36771 -8.17606

  4:   -38.264599: -2.06937 -1.33276 -7.69471

  6:   -49.652458: -2.09151 -1.82604 -5.94004

  8:   -65.778960: -1.14078 -3.64850 -6.76332

 10:   -68.197479: -1.13485 -3.65644 -6.28624

 12:   -68.233685: -1.06703 -3.64526 -6.32666

 14:   -68.234659: -1.07277 -3.64927 -6.33142
```

```
> system.time(b2 <- gppredict(a2,Data.new=as.matrix(x)))
```

Plotting Figure 3

```
> upper=b2$pred.mean+1.96*b2$pred.sd
> lower=b2$pred.mean-1.96*b2$pred.sd
>plot(-100,-100,col=0,xlim=range(X,x),ylim=range(upper,lower,Y),main="Prediction Rational Quadratic Kernel",xlab="time",ylab="GDP (in trillions of USD)")
> polygon(c(x,rev(x)),c(upper,rev(lower)),col="grey",border=NA)
> lines(x,b2$pred.mean,col=4,lwd=0.8)
> points(X,Y,pch=2,col=2,cex=0.1)
```

Sum of two kernels

```
> system.time(a3 <- gpr(as.matrix(X),as.matrix(Y),c('pow.ex','rat.qu'),trace=2))
```

```
    title: -likelihood:   pow.ex.v   pow.ex.w   rat.qu.w   rat.qu.s   rat.qu.a   vv
  0:   110.54225: 0.00662774 -0.0140595 0.00752139 0.321277 0.300286  1.50000
  2:   88.627155: -0.158767 -0.0490920 0.691586 0.230140 -2.29934 0.639263
  4:   83.926912: -0.207644 -0.0616716 0.740495 0.218004 -3.78907 0.484082
  6:   82.147321: -0.231931 -0.0679099 0.748590 0.215731 -4.15633 0.413268
  8:   81.630216: -0.239365 -0.0698637 0.750434 0.215193 -4.24734 0.392207
 10:    81.495813: -0.241322 -0.0703820 0.750884 0.215060 -4.27003 0.386708
 12:    81.461883: -0.241817 -0.0705135 0.750996 0.215027 -4.27570 0.385318
 14:    81.453380: -0.241942 -0.0705465 0.751024 0.215018 -4.27712 0.384969
 16:    81.451252: -0.241973 -0.0705548 0.751031 0.215016 -4.27747 0.384882
 18:    81.450721: -0.241981 -0.0705569 0.751032 0.215016 -4.27756 0.384860
 20:    81.450588: -0.241983 -0.0705574 0.751033 0.215016 -4.27758 0.384855
 22:    81.450554: -0.241983 -0.0705575 0.751033 0.215016 -4.27759 0.384854
 24:    81.450546: -0.241983 -0.0705575 0.751033 0.215016 -4.27759 0.384853
```

```
> system.time(b3 <- gppredict(a3,Data.new=as.matrix(x)))
```

Plotting Figure 4

```
> upper=b3$pred.mean+1.96*b3$pred.sd
> lower=b3$pred.mean-1.96*b3$pred.sd
>plot(-100,-100,col=0,xlim=range(X,x),ylim=range(upper,lower,Y),main="Prediction by sum of 3 Kernels",xlab="time",ylab="GDP (in trillions of USD)")
> polygon(c(x,rev(x)),c(upper,rev(lower)),col="grey",border=NA)
> lines(x,b3$pred.mean,col=4,lwd=0.8)
> points(X,Y,pch=2,col=2,cex=0.1)
```