**Tutorial on Bayesian optimization**

Loc Nguyen

Loc Nguyen’s Academic Network

Email: ng\_phloc@yahoo.com

Homepage: www.locnguyen.net

**Abstract**

**1. Introduction**

Before describing subjects related to Bayesian optimization (BO), it is necessary to mention conventions about mathematical notations. As a convention, lowercase and un-bold letters like *a*, *b*, *c*, *x*, *y*, and *z* denote scalar whereas lowercase and bold letters like ***a***, ***b***, ***c***, ***x***, ***y***, and ***z*** denote vector. In some cases, uppercase and un-bold letters like *A*, *B*, *C*, *X*, *Y*, and *Z* can denote vector. Uppercase and bold / un-bold letters like *A*, *B*, *C*,***A***,***B***,***C***, *X*, *Y*, *Z*,***X***,***Y***, and ***Z*** denote matrix. Variables can be denoted as *x*, *y*, *z*, ***x***, ***y***, ***z***, *X*, *Y*, *Z*, ***X***, ***Y***, and ***Z*** whereas constants can be denoted as *a*, *b*, *c*, ***a***, ***b***, ***c***, *A*, *B*, *C*, ***A***, ***B***, and ***C***. Uppercase and bold / un-bold letters like *X*, *Y*, *Z*,***X***, ***Y***, and ***Z*** can denote random variables. However, scalar, vector, matrix, variables, and random variables are stated explicitly in concrete cases.

Given target function *y* = *f*(***x***), optimization problem is to find out extremizer ***x***\* so that *f*(***x***\*) gets extreme value *y* = *f*(***x***\*). The extremizer ***x***\* can be minimizer or maximizer so that *f*(*x*\*) is minimum or maximum, respectively. Optimization problem is specified as follows:

Or

If the extremizer ***x***\* is local minimizer or local maximizer, the optimization problem is local optimization problem where traditional methods such as Newton-Raphson and gradient descent are perfect solutions but they require *f*(***x***) is totally convex (concave). The problem becomes much more complex if *f*(***x***) is not totally convex (concave) which leads that ***x***\* is global extremizer. This is the global optimization problem which is mentioned in this research. In literature, ***x***\* is minimizer by default but in context of Bayesian optimization (BO) it is better to consider ***x***\* as maximizer because BO mainly relates to probabilistic distributions whose peaks are concerned much. However, it is not serious because minimization is the inverse of maximization, for example:

There are three approaches to solve (global) optimization problem such as analytic approach, probabilistic approach, and heuristic approach. Analytic approach applies purely mathematical tools into find out optimizers such as approximation, cutting plane, branch and bound, and interval method, in which these methods focus on analytic essence of algebraic target functions. Probabilistic approach considers looking for optimizers as random selection but such random selection is guided by some probabilistic model so as to reach an optimizer. Heuristic approach which is the most flexible one among three approaches tries to apply or imitate heuristic assumptions into searching for optimizers. It does not concern much mathematical reasonings because feasibility and effectiveness are most important. As usual, heuristic approach imitates natural activities, for example, particle swarm optimization (PSO) simulates how a flock of birds search for food. Evolutional algorithms like PSO and ant bee colony (ABC) which are inspired from biological activities are popular methods of heuristic algorithms. However, there are some implicit connections between heuristic approach (concretely, evolutional algorithms) and probabilistic approach that I mentioned in a research.

Bayesian optimization (BO) belongs to probabilistic approach. It is based on Bayesian inference which considers parameter as random variable and updates posterior probability of parameter based on evidence and prior probability. Because BO does not impact directly on target function *f*(***x***), it must model *f*(***x***) as a probabilistic model and then define an acquisition function for such probabilistic model. BO solves the optimization problem by maximizing the acquisition function instead of maximizing the target function. Shortly, two main tasks of BO are:

1. Modeling *f*(***x***) by the probabilistic model called surrogate model.
2. Defining the acquisition function for the surrogate model so that it is possible to maximize the acquisition function.

Posterior probability of surrogate model in BO is updated continuously along with maximizing acquisition function continuously until an maximizer of target function is reached. Following is the pseudo code of BO.

|  |
| --- |
| Modeling *f*(***x***) by the surrogate model *S*(*f* | ***D****n*) where sample ***D****n* is a set of variable values ***x****i*.  Defining the acquisition function *α*(***x*** | *f*, *S*) based on both *f*(***x***) and *S*(*f* | ***D****n*) whose variable is ***x***.  Initializing randomly ***D****n* = {***x***1, ***x***2,…, ***x****n*} and calculating prior probability of *S*(*f* | ***D****n*) with sample ***D****n*.  While maximizer ***x***\* is not reached or the number of iterations is not many enough  Determine acquisition function *α*(***x*** | *f*, *S*) with *S*(*f* | ***D****n*) if possible.  Find ***x****n*+1 as a maximizer of *α*(***x*** | *f*, *S*) with regard to ***x***.  If acquisition function *α*(***x*** | *f*, *S*) is undetermined yet then randomizing ***x****n*+1.  Checking whether ***x****n*+1 is maximizer ***x***\*. Adding ***x****n*+1 to sample ***D****n*.  Update posterior probability of *S*(*f* | ***D****n*) with updated sample ***D****n*.  Increase *n* = *n* + 1.  End while |

**Table 1.1.** BO algorithm

If the surrogate model *S*(*f* | ***D****n*) has explicit parameters related directly to *f*(***x***), updating posterior probability of *S*(*f* | ***D****n*) is indeed to update posterior probability of its explicit parameters and then, BO is called parametric BO. If the surrogate model *S*(*f* | ***D****n*) has no explicit parameters or its parameters are not related directly to *f*(***x***) then, BO is called nonparametric BO. As seen in BO algorithm, BO does not concern algebraic formulation of the target function *f*(***x***) because it only concerns output *y*=*f*(***x***) and hence, *f*(***x***) is a black box with regard to BO. In other words, *f*(***x***) is an arbitrary mapping with subject to BO and so BO is a flexible solution for global optimization, which has many potential applications. Moreover, the essence of BO is continuous improvement via updating posterior probability, which follows ideology of reinforcement learning. Therefore, it is possible to consider BO as a reinforcement learning technique. Before focusing on BO, it is better to describe shortly some relevant basic concepts but you can ignore such concepts in order to research directly BO.

**2. Basic concepts**

Bayes’ rule which is based on conditional probability improves conditional probability by proposing prior probability, considering conditional probability as posterior probability, proposing likelihood function, and apply total probability rule into calculating likelihood function. Bayesian inference or Bayesian statistics, which is based on Bayes’ rule but goes beyond Bayes’ rule, considers parameter as a random variable and so parameter is also called hypothesis. Because parameter is random variable, prior and posterior probabilities in Bayes’ rule are prior and posterior probabilities of parameter. Anytime evidence as sample occurs, posterior probability of parameter is updated such sample and prior probability in order to consolidate or weaken the hypothesis.

**3. Bayesian optimization**

As aforementioned, BO solves the optimization problem by maximizing the acquisition function *α*(***x*** | *f*, *S*) which is derived from the surrogate model *S*(*f* | ***D****n*) where ***D****n* is the current sample. The surrogate model *S*(*f* | ***D****n*), in turn, was a probabilistic representation of target function *f*(***x***). The way to define surrogate model decides the kind of BO where there are two kinds of BO such as parametric BO (PBO) and nonparametric BO (NBO). PBO implies that parameters of surrogate model *S*(*f* | ***D****n*) were included explicitly in target function *f*(***x***). Otherwise, surrogate model *S*(*f* | ***D****n*) of NBO has no explicit parameters or its parameters were not included explicitly in target function *f*(***x***). Firstly, PBO is surveyed and then NPO is researched.

When *f*(***x***) does not have specific (explicit) aspect or property which becomes an explicit parameter of the surrogate model *S*(*f* | *Dn*), the corresponding BO becomes nonparametric BO (NBO). The most popular technique to establish *S*(*f* | *Dn*) for NBO is to use Gaussian process regression (GPR) for modeling *S*(*f* | *Dn*). In other words, GPR is a surrogate model of NBO. Because kernel function is very important to GPR when it is not only used to build up GPR but also used to make GPR line smoother. Kernel function measures similarity between two variables (two points), according to that, the closer the two variables are, the larger their kernel function is. Let Σ(***x****i*, ***x****j*) denote kernel function of two variables ***x****i* and ***x****j*, for example, a popular kernel function is simple squared exponential function.

In this research the notation |.| can denote absolute value of scalar, length (module) of vector, determinant of matrix, and cardinality of set. Concretely, kernel function Σ(***x****i*, ***x****j*) is used to define covariance matrix in GPR. As a convention, let ***x****i*:*j* where *i*≤*j* denote a sequential subset of ***X*** such that ***x****i*:*j* = {***x****i*, ***x****i*+1,…, ***x****j*}. Of course, we have ***x****i*:*i* = ***x****i*. Given two subsets ***x****i*:*j* and ***x****k*:*l*, their covariance matrix is:

For NBO, given prior sample ***D****n* = ***x***1:*n* = {***x***1, ***x***2,…, ***x****n*}, the consequential sequence *y*1:*n* = {*y*1 = *f*(***x***1), *y*2 = *f*(***x***2),…, *yn* = *f*(***x****n*)} is established to distribute normally with prior probability density function (prior PDF) as follows:

Note, in this research, notation *f*(.|.) often denote PDF, which is not concrete function. Indeed, *μ*(*.*) and Σ(.) are mean function and covariance function, respectively. Mean function *μ*(*.*) is often set to be zero as *μ*(*y*1:*n*) = **0***T*. In this special case, covariance function Σ(.) is matrix function defined based on kernel function.

Of course, each element Σ(***x****i*, ***x****j*) of Σ(***x***1:*n*, ***x***1:*n*) is kernel function. Suppose we have a new value ***x*** = ***x****n*+1 which is a maximizer of acquisition function and then obtain the posterior sample ***D****n* = ***x***1:(*n*+1) = {***x***1, ***x***2,…, ***x****n*, ***x****n*+1}. Surrogate model GPR is represented by the posterior PDF of *y* = *f*(***x***) given ***Y***, ***X***, and ***x****n*+1 as follows:

Where *μn*(*y*) and *σn*2(*y*) are mean and variance of the posterior PDF of *y* given *y*1:*n*, ***x***1:*n*, and ***x*** = ***x****n*+1.

Recall that:

It is common to define *μ*(*y*1:*n*) based on *μ*(*y*) separately:

Although the mean function *μ*(*y*) is not important as the kernel function Σ(***x****i*, ***x****j*) in NBO but it should not be ignored. As usual, *μ*(*y*) is set to be 0 as *μ*(*y*) = 0 or itself as *μ*(*y*) = *y*.

Indeed, GPR is represented by the two parameters *μn*(*y*) and *σn*2(*y*) but such parameters are not included in the target function *f*(***x***) and so this is called NBO. The pseudo code of NBO is finetuned as follows:

|  |
| --- |
| Initializing randomly ***D****n* = ***x***1:*n* = {***x***1, ***x***2,…, ***x****n*}.  While maximizer ***x***\* is not reached or the number of iterations is not many enough  Determine acquisition function *α*(***x*** | *μn*(*y*), *σn*2(*y*)) based on mean *μn*(*y*) and variance *σn*2(*y*) of GPR if possible.  Find ***x****n*+1 as a maximizer of *α*(***x*** | *μn*(*y*), *σn*2(*y*)) with regard to ***x***.  If acquisition function *α*(***x*** | *μn*(*y*), *σn*2(*y*)) is undetermined yet then randomizing ***x****n*+1.  Checking whether ***x****n*+1 is maximizer ***x***\*. Adding ***x****n*+1 to sample ***D****n*.  Update posterior mean *μn*(*y*) and variance *σn*2(*y*) of GPR with updated sample ***D****n* as follows:  Increase *n* = *n* + 1.  End while |

**Table 3.1.** NBO algorithm with GPR

In general, there are two important tasks of NBO in which the first one is to determine the posterior mean *μn*(*y*) and variance *σn*2(*y*) of GPR. The second one is to specify the acquisition function *α*(***x*** | *μn*(*y*), *σn*2(*y*)) which is described shortly here. Detailed description of GPR and acquisition function will be mentioned in the next sections. Note that *α*(***x*** | *μn*(*y*), *σn*2(*y*)) is function of ***x***.

**4. Gaussian process regression**

Nonparametric BO is based on Gaussian process regression (GPR) which, in turn, is based on Gaussian process. Therefore, we start the description of GPR with a concept of Gaussian process. Given a random process ***X*** = (*X*1, *X*2,…, *Xn*) over *n* timepoints in which each *Xt* where *t* belongs to {1, 2,…, *n*} is random variable, if and only if for any finite index set {*t*1, *t*2,…, *tk*} of {1, 2,…, *n*} where *tj* belongs to {1, 2,…, *n*}, the subset considered as *tk*-dimension random vector follows multinormal distribution known as multivariate Gaussian distribution then ***X*** is called *Gaussian random process* or Gaussian process (GP) in brief. Note, each represents one dimension of the *k*-dimension random vector variable . Moreover, please pay attention that any combination of follows normal distribution too. Without loss of generality, we denote the random process as random variable ***X*** = (*X*1, *X*2,…, *Xn*)*T* where {*t*1, *t*2,…, *tk*} = {1, 2,…, *n*} obeying multinormal distribution (multivariate normal distribution or multivariate Gaussian distribution) as follows:

Where *μ*(***X***) and Σ(***X***) are mean vector and covariance matrix of ***X***, respectively. Note, the superscript “*T*” denotes transposition operator of vector and matrix whereas the tilde sign “~” indicates probabilistic distribution of a random variable. GP is known as infinite multinormal distribution which is the generalized form of *n*-dimension multinormal distribution because *n* can approach positive infinity *n* = +∞. Let *f*(***X*** | *μ*(***X***), Σ(***X***) be probability density function (PDF) of ***X*** when ***X*** is continuous, we have:

Indeed, mean *μ*(***X***) and covariance Σ(***X***) are functions of ***X***, respectively. Note, in this research, notation *f*(.|.) often denote PDF, which is not concrete function. In literature, *μ*(***X***) is assumed to be zero as *μ*(***X***) = **0***T* for convenience but it can be defined as the random process ***X*** itself, *μ*(***X***) = ***X***.Besides, *μ*(***X***) can be customized according to concrete applications, for example it can be constant as *μ*(***X***) = ***μ***. Covariance function Σ(***X***) measures the correlation between random variables when the random process “moves” them, in which the closer the given two random variables are, the larger their covariance. It is invertible, positive definite, and symmetric. The two most important properties based on covariance function Σ(***X***) of random process are stationarity and isotropy among four basic properties: stationarity, isotropy, smoothness, and periodicity. Stationarity implies that the PDF *f*(***X*** | *μ*(***X***) of random process ***X*** will not be changed when the process is shifted in time, for example, if new random variable *Xn*+1 raises to be added then means and covariances of old (previous) variables *Xi* where 1 ≤ *i* ≤ *n* in ***X*** will not be changed. It is proved that if GP ***X*** satisfies stationarity, Σ(***X***) will depend only on the deviation *Xi*–*Xj* but the inversed statement is not asserted. However, if Σ(***X***) depends only on the Euclidean distance |*Xi*–*Xj*| then, GP ***X*** will satisfy isotropy. If ***X*** satisfies both stationarity and isotropy, ***X*** is called homogeneous process. In cases where each element of matrix function Σ(***X***) depends only on *Xi* and *Xj* like stationarity case and isotropy, it will be formulated as following matrix.

In these cases, covariance function Σ(***X***) is identified with its element function Σ(*Xi*, *Xj*) if the formulation of Σ(*Xi*, *Xj*) is not changed formally and hence, Σ(*Xi*, *Xj*) is called *kernel function*. Kernel function is used to make GPR line smoother. The following are some kernel functions.

Where *l* is the characteristic length-scale of the process which reinforces similarity of *Xi* and *Xj*. In this research the notation |.| can denote absolute value of scalar, length (module) of vector, determinant of matrix, and cardinality of set. As a convention, let *Xi*:*j* where *i*≤*j* denote a sequential subset of ***X*** such that *Xi*:*j* = {*Xi*, *Xi*+1,…, *Xj*}. Of course, we have *Xi*:*i* = *Xi*. In general case, *Xi*:*j* is arbitrary subset of distinguish variables. Let Σ(*Xi*, *Xj*:*k*) denote a row covariance vector of *Xi* and *Xj*:*k* as follows:

Let Σ(*Xj*:*k*, *Xi*) denote a column covariance vector of *Xj*:*k* and *Xi* as follows:

As a convention, let Σ(*Xi*:*j*, *Xk*:*l*) denote a pseudo covariance matrix of *Xi*:*j* and *Xk*:*l* as follows:

Note, Σ(*Xi*:*j*, *Xk*:*l*) is not an actual covariance matrix because it may not be squared. Obviously, Σ(*Xi*:*j*, *Xk*:*l*) is a partition of Σ(***X***) and we have Σ(***X***) = Σ(*Xi*:*n*, *Xi*:*n*). If denoting ***X***1 =(*Xi*, *Xi*+1,…, *Xj*)*T* and ***X***2 = (*Xk*, *Xk*+1,…, *Xl*)*T*, we denote:

Gaussian process repression (GPR) is based on Gaussian process (GP) when there is a target function which attaches to each *X* where *X* represents any *Xi* in ***X*** such that *Y* = *f*(*X*). Please distinguish the target function *f* from the formal notations *f*(.|.) of probability density function (PDF). Of course, *Y* or *Yi* is also random variable and we also have *Yi* = *f*(*Xi*). Besides, in context of regression model, the target function *f*(*X*) is not a formal function with arithmetic operators and exactly, it is a mapping between *X* and *Y*. For example, sample {***X***, ***Y***} has two paired datasets ***X*** and ***Y*** in which for every *Xi* belonging to ***X*** there is a *Yi* belong to ***Y*** and hence, the equation *Y* = *f*(*X*) only indicates such mapping. GPR model tries to represent or draw a regressive PDF of ***Y*** from its previous ones (which will be explained later) and ***X***. Suppose we had a GP ***X*** = (*X*1, *X*2,…, *Xn*)*T* and target function ***Y*** = *f*(***X***), assuming that the prior PDF of ***Y*** = (*Y*1, *Y*2,…, *Yn*)*T* given ***X*** is also derived from the multinormal PDF of ***X***.

Therefore, we have:

Where *μ*(.) and Σ(.) denote mean function and covariance function (kernel function) in formality, respectively, and Σ(***X***, ***X***) = Σ(***X***). The equation above implies that the prior PDF of GP ***Y*** is initialized with the PDF of GP ***X***. Mean vector and covariance matrix of ***Y*** are *μ*(***Y***) and Σ(***X***, ***X***) *\**) within the PDF *f*(***Y*** | ***X***, *μ*(.), Σ(.)), respectively. The mean function *μ*(***Y***) is redefined here with variable ***Y*** but it should be the same to the mean function *μ*(***X***) in formulation if *μ*(***X***) can be extended with the lower/higher dimensional space. For instance, if *μ*(***X***) is defined as itself *μ*(***X***) = ***X*** then, *μ*(***Y***) will be defined as itself *μ*(***Y***) = ***Y*** = (*Y*1, *Y*2,…, *Yn*)*T*. In literature, for simplicity, *μ*(***Y***) is set to be zero as *μ*(***Y***) = **0***T*. It is better if setting *μ*(***Y***) = **0***T*.

Now suppose we randomize a new set of variables ***X***\* = (*Xn*+1, *Xn*+2,…, *Xn*+*k*)*T* and obtain their evaluated values ***Y***\* = *f*(***X***\*) = (*Yn*+1, *Yn*+2,…, *Yn*+*k*)*T*. Similarly, mean vector and covariance matrix of ***Y****\** are *μ*(***Y****\**) and Σ(***X****\**, ***X****\**) within the PDF *f*(***Y***\* | ***X***\*, *μ*(.), Σ(.)), respectively. In literature, for simplicity, *μ*(***Y***) and *μ*(***Y****\**) are set to be zero as *μ*(***Y***) = **0***T* and *μ*(***Y****\**) = **0***T*. Consequently, the joint PDF of ***Y*** and ***Y***\* which is of course multinormal distribution is denoted as follows:

Where mean vector and covariance matrix of the joint PDF of ***Y*** and ***Y***\* are and , respectively. As a result, the Gaussian process regression (GPR) model of dependent variable *Y* is conditional PDF of ***Y***\* given ***Y***, ***X***, and ***X***\* such as *f*(***Y***\* | ***Y***, ***X***, ***X***\*, *μ*(.), Σ(.)) which is also predictive PDF of ***Y***\* because it can be used to predict next occurrence of *Y*. From interesting properties of multinormal distribution, it is easy to drawn GPR from the joint PDF *f*(***Y***, ***Y***\* | ***Y***, ***X***, ***X***\*, *μ*(.), Σ(.)) as follows:

Where,

Obviously, the GPR of ***Y***\* distributes normally with mean vector *μ*(***Y***\* | ***Y***, ***X***, ***X***\*) and covariance matrix Σ(***Y***\* | ***X***, ***X***\*). Indeed, the GPR *f*(***Y***\* | ***Y***, ***X***, ***X***\*, *μ*(.), Σ(.)) is posterior PDF of ***Y*** whose prior PDF is *f*(***Y*** | ***X***, *μ*(.), Σ(.)). Initializing such prior PDF by the PDF of ***X*** as *f*(***Y*** | ***X***, *μ*(.), Σ(.)) = *f*(***X*** | *μ*(.), Σ(.)) is not totally correct but implicit biases will be decreased after the posterior PDF *f*(***Y***\* | ***Y***, ***X***, ***X***\*, *μ*(.), Σ(.)) is updated. We have following summary:

Although the GPR of ***Y***\* depends on ***Y***, ***X***, and ***X***\*, the main semantic meaning of regression model here is mentioned mainly that ***Y***\* is determined based on its previous one ***Y*** when both of them are assumed to be based on *X* via the PDF of ***Y*** as *f*(***Y*** | ***X***, *μ*(.), Σ(.)) and the PDF of ***Y***\* as *f*(***Y***\* | ***X***\*, *μ*(.), Σ(.)). This implies that *X* is the intermediate point for the probabilistic dependence of ***Y***\* on ***Y***.

If ***X***\* and ***Y***\* are 1-element vectors such that ***X***\* = *Xn*+1 and ***Y***\* = *Yn*+1 and let ***x***1:*n* = ***X***, ***x*** = ***X***\*, *y*1:*n* = ***Y***, and *y* = ***Y***\*, the GPR becomes:

Where *μn*(*y*) and *σn*2(*y*) are mean and variance of the posterior PDF of *y* given *y*1:*n*, ***x***1:*n*, and ***x***.

The equation above with single variables ***x*** and *y* (single posterior processes) is popular in BO, especially ***x*** is vector and *y* is scalar although ***x*** and *y* can be arbitrary such as scalar, vector, and matrix with note that high dimensional spaces require tensor products for example.

GPR can be executed continuously with new ***X***\* while the old ***X***\* is incorporated into larger GP ***X***. In practice, such continuous execution is implemented as iterative algorithm whose each iteration has two following steps:

|  |
| --- |
| *Step* 1:  Take new ***X***\*  *Step* 2: |

**Table 4.1.** GPR algorithm

In step 1, the covariance matrix Σ(***X***, ***X***) is not recomputed entirely because some of its elements Σ(*Xi*, *Xj*) were determined before.

Because GP ***X*** will get huge when the iterative algorithm repeats many iterations and ***X***\* is incorporated into ***X*** over and over again, it is possible to apply first-order Markov property so that each iteration remembers only one previous *X*. Therefore, we can assign ***X***\* to ***X*** as ***X*** = ***X***\* in step 2 so that the iterative algorithm runs faster and saves more computational resources as follows:

|  |
| --- |
| *Step* 1:  Take new ***X***\*  *Step* 2: |

**Table 4.2.** GPR algorithm with first-order Markov property

However, this advantage of first-order Markov property cannot be applied into BO (by defining how to take new ***X***\* with acquisition functions) because it is necessary to look entire *X* domain forward and backward to find out potential global optimizer. The use of Markov property will be appropriate to tasks of prediction and estimation in which entire time series are split into many smaller time windows where it is necessary to remember *w* windows with *w*-order Markov property. Shortly, the assignment ***X*** = ***X***\* is suitable to forward modeling.

GPR focuses mainly on the regressive (posterior) covariance matrix Σ(***Y***\* | ***X***, ***X***\*) via covariance function Σ(*Xi*, *Xj*) because adjusting the regressive (posterior) mean vector *μ*(***Y***\* | ***Y***, ***X***, ***X***\*) via adjusting the mean function *μ*(***Y***) is unnecessary due to:

As usual, *μ*(***Y***) and *μ*(***Y****\**) are set to be zero as *μ*(***Y***) = **0***T* and *μ*(***Y****\**) = **0***T* so that

Which is indeed an arithmetic regression of ***Y***\* on ***Y***, ***X***, and ***X***\* in addition to the main semantic meaning that the posterior PDF of ***Y***\* given ***Y*** is the main regression. However, it is still better to define exactly the mean function *μ*(***Y***) in cases of predicting more precisely confident intervals of *Y* based on *X* because the equation above:

Which implies variation of *Y* according to variation of *X* from the origin. It is not a real value of *Y*. By another way, if setting *μ*(***Y***) = ***Y*** and *μ*(***Y****\**) = ***Y****\** so that

That is not a regression function, which is impossible to predict *Y*\* based on *X*\* when *Y*\* is unknown. Therefore, I propose a technique based on linear regression model to define *μ*(***Y***\*) with constraint that each element *μ*(*Y*\*) of *μ*(***Y***\*) depend only on *Y*\*, for instance:

Note, *Y*\* represents any *Yi* belonging to the subprocess ***Y***\* = (*Yn*+1, *Yn*+2,…, *Yn*+*k*)*T* and *X*\* represents any *Xi* belonging to the subprocess ***X***\* = (*Xn*+1, *Xn*+2,…, *Xn*+*k*)*T*. Let *φ*(*X*) be the transformation function which transforms *X* space into *Y* space such that *φ*(.) is invertible.

This means that *φ*(.) can be defined only if *X* space and *Y* space have the same dimension. The simplest form of *φ*(.) is identity function.

Let *Z\** be a regressive estimate of *X*\*:

Where ***α****\** is regressive coefficient vector with suppose that ***X***\* is also vector. How to calculate ***α****\** from sample {***X***\*, *φ*–1(***Y***\*)} will be described later. Please pay attention that this linear regression is totally different from the regression meaning of GPR via posterior PDF. Let be an estimate of *Y*\* from ***X***\* with association of linear regression model and transformation function *φ*(.).

Obviously, we have:

Let |***X***|, |***X***\*|, |***Y***|, and |***Y***\*| be cardinalities (also dimensions) of ***X***, ***X***\*, ***Y***, and ***Y***\*. Of course, we have |***X***| = |***Y***| and |***X***\*| = |***Y***\*|, for example, we also have |***X***| = |***Y***| = *n* and |***X***\*| = |***Y***\*| = *k*. Concentration ratio (CR) of a subprocess which is point density of such subprocess is defined by the ratio of cardinality of such subprocess to the cardinality of entire process, for example, CR of ***X***\* over {***X***, ***X***\*} is:

Suppose ***α*** being the regressive coefficient vector estimated from sample {***X***, *φ*–1(***Y***)} was determined before, the mean function *μ*(*Y*\*) is proposed as a weighted estimation with concentration ratios of ***X*** and ***X***\* as follows:

As a result, the two steps of the iterative algorithm for estimating GPR are finetuned as follows:

|  |
| --- |
| *Step* 1:  Take new ***X***\*  Calculate ***α****\** from sample {***X***\*, *φ*–1(***Y***\*)}  *Step* 2: |

**Table 4.3.** GPR algorithm with linear regression

Where,

Note, the vector *φ*–1(***Y***\*) is determined:

Both mean vector *μ*(***Y***\* | ***X***, ***X***\*) and covariance matrix Σ(***Y***\* | ***X***, ***X***\*) of GPR with linear regression are free from ***Y*** because they are totally based on only ***X*** and ***X***\*.

For interval estimation of *Y*\* with given ***X***\*, suppose GPR algorithm finished obtaining ***α*** and *μ*(*Y*) after some iterations, we calculate:

The mean function *μ*(*Y*\*) is recalculated with ***X***\* as follows:

As a result, the confident interval of *Y*\* which is the pair {*μ*(***Y***\* | ***X***, ***X***\*), Σ(***Y***\* | ***X***, ***X***\*)} is determined as follows:

Because the GPR algorithm here defines the mean function *μ*(*Y*\*) based on multiple linear regression (MLR) model, it is necessary to describe LRM in short. Given a dependent random variable *Z* and a set of independent random variables *X* = (1, *X*1, *X*2,…, *Xn*)*T*, MLR tries to establish linear relationship between *Z* = (*Z*1, *Z*2,…, *Zm*)*T* and *X* so that *Z* is sum of a linear combination of *X* and an random error vector ***ε***.

As a convention, *Xj* are called regressors and *Zi* is called responsors whereas *A* = (*α*0, *α*1, *α*2,…, *αn*)*T* is *m*x(*n*+1) regressive coefficient matrix.

Suppose ***ε*** distributes normally with mean vector **0***T* and covariance matrix Σ then *Z* distributes normally with mean vector *AX* and covariance matrix Σ due to:

Note, *E*(.) and *V*(.) denote theoretical expectation and variance, respectively and Σ is *m*x*m* invertible matrix. This implies that the PDF of random variable *Z* is:

MLR of *Z* given *X* is built from sample {***X***, **Z**} of size *N* in form of data matrix as follows:

Therefore, *xij* and *zik* is the *i*th instances of regressor *Xj* and responsor *Zk* at the *i*th row of matrix (***X***, **Z**). As a convention, we can connect datasets ***X*** and ***Z*** of MLR here with the GP ***X\**** and the set *φ*–1(***Y***\*) aforementioned, respectively with note that removing values 1 from datasets ***X***.

The essence of MLR is to estimate the regressive coefficient matrix *A* and the covariance matrix Σ. By applying maximum likelihood estimation (MLE) method, we obtain estimates and of *A* and Σ.

**5. Acquisition functions**

**6. Numerical simulation**

**7. Conclusions**

**Appendices**

**References**