A Fully Bayesian Approach to the Efficient Global Optimization Algorithm

Sam D. Tajbakhsh⋆[*], Enrique del Castillo⋆[†] and James L. Rosenberger⋆⋆[‡]

⋆ Department of Industrial and Manufacturing Engineering, The Pennsylvania State University, University Park, PA 16802, USA

⋆⋆ Department of Statistics, The Pennsylvania State University, University Park, PA 16802, USA

April 2012
Last Revision: December 2012

Abstract

Finding the global optimum(s) of a non-convex function is of great importance in numerous applications in science and engineering where the function takes the form of an expensive computer code and its inputs are the independent variables. For this type of problem, Jones et al. [12] proposed the idea of expected improvement (EI) and embedded it in an algorithm called efficient global optimization, or EGO. Neither EI nor EGO consider the uncertainty in the parameter estimates. One way to account for these uncertainties is to use Bootstrapping. In this paper, instead, we formulate the expected improvement method from a fully Bayesian perspective which results in a corresponding Bayesian EGO method. The performance of the proposed Bayesian EGO is illustrated and compared with the standard EGO method of Jones et al. and the bootstrapped EGO of Kleijnen et al. [13]. Furthermore, we apply the Bayesian EGO algorithm for the optimization of a stochastic inventory simulation model.

Keywords: Global optimum, geostatistics, hierarchical model, expected improvement

[*]Mr. Tajbakhsh is a Ph.D. student in Industrial Engineering. e-mail: sdt144@psu.edu.

[†]Dr. Del Castillo is Distinguished Professor of Industrial & Manufacturing Engineering and Professor of Statistics. e-mail: exd13@psu.edu

[‡]Dr. Rosenberger is Professor of Statistics. e-mail: jlr@psu.edu

1 Introduction

In numerous applications in science and engineering, the input/output relation of a system is frequently represented by a computer code with certain inputs we wish to determine. Two such instances are finite element models in aerospace mechanical design and systems of partial differential equations (PDE) that model chemical reactions in a petrochemical plant. The computer codes can be regarded as a function mapping from the input space to the output (response) space. Functions of this type usually share two properties: 1) they are computationally expensive, that is, each run of the code takes considerable amount of time and 2) they are highly non-convex. These two properties make the global optimization of these functions a big challenge.

There have been many attempts at building a "metamodel" to approximate expensive computer code using a statistical model. Polynomial regression, splines, kriging, radial basis functions, neural networks and support vector machine (SVM) have been used as metamodels (for a review of these techniques see [20]). Among these statistical models, kriging has probably received the most attention. Using a suitable covariance structure, kriging can properly approximate even a highly non-convex function; furthermore, in the absence of random error, i.e. if the computer code is deterministic, kriging is an *exact interpolator* that is, kriging predictions perfectly match the observed data [3]. It should be noted that kriging methods are sometimes referred to as Gaussian Process (GP) model in the statistics literature and "Gaussian Random Function" model in the Operations Research literature. In a seminal paper, Sacks et al. [17] proposed universal kriging as a metamodel to approximate a computer code.

To optimize the inputs of a computer code through metamodeling, there exists a tradeoff between exploration and exploitation. By exploration, we mean searching the experimental domain and escaping from local optimums, and by exploitation we mean moving toward the global optimums as close as possible. Most of the procedures which use metamodels consist of sequential iterations between parameter estimation (rebuilding the model) and optimization of the model at the given iteration. These methods are mainly discussed in the context of sequential design of optimization experiments ([22], [23], [18] and [4]).

Mockus et al. [15] and Jones et al. [12] proposed the Expected Improvement (EI) criterion which combines the mean and variance structures of the kriging predictor such that the method explores the experimental domain and at the same time exploits the potential areas where local optima occur. Jones et al. [12] named the resulting algorithm *Efficient Global Optimization* (EGO)– see also [19] and [11].

Den Hertog et al. [5] and Sjostedt de Luna and Young [21] showed that the formula for estimating the kriging prediction variance underestimates the true prediction variance in expectation (as we will see later in (5)). Den Hertog et al. [5] and Sjostedt de Luna and Young [21] both proposed bootstrapping as a means to estimate the true kriging variance. This idea was adopted by Kleijnen et al. [13] in their bootstrapped EGO algorithm and used it for simulation optimization.

In this paper, we propose a fully bayesian approach to evaluate the expected improvement at any given point of a sequence of computer runs using the posterior predictive distribution. This is then embedded in an optimization method which we call Bayesian EGO. The rest of the paper is organized as follows. In Section 2, the classic expected improvement and its bootstrapped counterpart are described. Section 3 contains our proposed bayesian expected improvement approach. In Section 4, the performance of classic EI, bootrapped EI and bayesian EI is compared through several test functions. Section 5 contains implementation of bayesian EGO for a stochastic simulation problem from the inventory control literature. Finally Section 6 gives concluding remarks and directions for further work.

2 Classic and Bootstrapped Expected Improvement

Assume that the response or dependent variable at a generic location $x \in D \subset \mathbb{R}^k$ is $y(x)$. A spatial model has the form

$$y(x) = \mu + z(x) + \epsilon(x) \tag{1}$$

where μ is the mean, $z(x)$ is a Gaussian stochastic process $z(x) \sim N(0, \sigma^2)$ modeling the spatial dependence, and $\epsilon(x)$ is assumed to be an independent Gaussian process $\epsilon(x) \sim N(0, \tau^2)$ modeling the error between the computer code and the real system. The $z(x)$ process is assumed to be *isotropic* [1], in other words, the covariance between any two spatial locations, e.g. $x^{(i)}$ and $x^{(j)}$, is a function of their distance, h, that is $Cov(z(x^{(i)}), z(x^{(j)})) = f(h; \Theta)$

where Θ is the set of covariance function parameters. Therefore, the covariance of the response between any two locations is $Cov(y(x^{(i)}), y(x^{(j)})) = Cov(z(x^{(i)}), z(x^{(j)}))$ and the response variance at any location x is $Var(y(x)) = \sigma^2 + \tau^2$. As originally defined by Jones et al. [12] the Expected Improvement at point x is defined as

$$E[I(x)] = E[\max(f_{min} - Y(x), 0)] \tag{2}$$

where f_{min} is the current best (minimum) function value. Note that $I(x) = \max(f_{min} - Y(x), 0)$ is the improvement at the point x which is a random variable since it is a function of $Y(x)$. If $\widehat{y}(x)$ and $s^2(x)$ are the mean and the variance estimators of Y at point x, then assuming $Y(x) \sim N(\widehat{y}(x), s^2(x))$, the expectation in (2) can be simplified to

$$E[I(x)] = (f_{min} - \widehat{y}(x))\Phi\left(\frac{f_{min} - \widehat{y}(x)}{s(x)}\right) + s(x)\phi\left(\frac{f_{min} - \widehat{y}(x)}{s(x)}\right) \tag{3}$$

where Φ and ϕ are the cdf and the pdf of the standard normal distribution, respectively. Moreover, $\widehat{y}(x)$ is the Best Linear Unbiased Predictor (BLUP) of $Y(x)$ and $s^2(x)$ is the mean squared error of the predictor ([18]) defined as follows:

$$\widehat{y}(x) = \widehat{\mu} + r'R^{-1}(\mathbf{y} - \widehat{\mu}\mathbf{1}) \tag{4}$$

$$s^2(x) = \widehat{\sigma^2}\left(1 - r'R^{-1}r + \frac{(1 - \mathbf{1}'R^{-1}\mathbf{r})^2}{\mathbf{1}'R^{-1}\mathbf{1}}\right) \tag{5}$$

In equations (4) and (5), R is the $n \times n$ correlation matrix between the entries in $\mathbf{Y}^T = [Y(1), ..., Y(n)]$ and r is $n \times 1$ correlation vector between $Z(x)$ and $[Z(1), ..., Z(n)]^T$.

The EGO algorithm consists of maximizing the expected improvement criterion given the vector of observed function values $\mathbf{Y}$ and the matrix of locations $\mathbf{X}$ and obtaining an optimal solution location (x^*). We then evaluate the function (computer code) at x^*, find y^* and add x^* to the bottom of X and y^* to the bottom of $\mathbf{Y}$. This procedure is repeated until a stopping criterion is satisfied.

The notion of expected improvement has gained considerable attention in recent years. Ginsbourger and Riche [8] show suboptimality of the 1-point ahead maximization of EI at each iteration of the EGO algorithm by means of a counterexample. They further proposed a finite time horizon dynamic programming approach to maximize a multipoint expected improvement criterion given a finite budget of experimentation. Huang et al. [10] proposed an *augmented* expected improvement criterion for optimizing stochastic responses which accounts for the uncertainty in the current best solution.

Using the kriging variance estimate (5) in the expected improvement formula (3), we get the *classic EI* formula. However, Den Hertog et al. [5] showed how the plug-in variance estimator (5) underestimates the true variance of the kriging model and proposed instead parametric bootstrapping to estimate $MSE[y(x)]$. They first calculated the maximum likelihood estimates of the parameter set of the kriging model (the mean, variance and correlation parameters) using the original data. The resulting model with MLE parameters is then used to sample the bootstrapped observations.

Kleijnen et al. [13] incorporated the *parametric* bootstrapped estimate of kriging variance in (3) and named it *bootstrapped EI.* From the original dataset $(\mathbf{X}, \mathbf{Y})$, they first find the Maximum Likelihood Estimates (MLE) of the kriging parameters and after replacing the parameters with their ML estimates and assuming gaussian distribution, they sampled at the new point where a prediction is desired x_{new}, namely $y^*_{new;b}$. They also sampled a bootstrapped dataset at the locations of the original data points $\mathbf{X}$, namely $\mathbf{Y}^*_b$. Next, using the bootstrapped dataset, $(\mathbf{X}, \mathbf{Y}^*_b)$, they calculated the bootstrapped ML estimates of the kriging model and used that model to predict at the new point x_{new}, namely $\hat{y}^*_{new;b}$. This procedure is repeated B time, $b = 1, ..., B$ (where B is the bootstrap sample size), and the bootstrapped variance estimate of the kriging model is estimated as

$$s^2_B(x_{new}) = \frac{1}{B} \sum_{b=1}^{B} (\hat{y}^*_{new;b} - y^*_{new;b})^2 \tag{6}$$

The *Bootstrapped EI* uses $s^2_B(x_{new})$ as the estimate of the kriging prediction variance at any new point. Note that in principle, the whole procedure should be followed for each candidate point x_{new}. However, to speed-up the computations, they use the same bootstrapped MLE computed from $(X, \mathbf{Y}^*_b)$ for all candidates.

A major issue to tackle using either classic EI or bootstrapped EI is finding the maximum likelihood estimate of the kriging parameters. Note that using bootstrapped EI, in each iteration of the bootstrapped EGO algorithm, $B+1$ maximization problems need to be solved compared to only one maximization for classic EGO. Generally, the constrained maximization of the likelihood function even for the simplest model structure is not an easy task and the routine may converge to a local maximum.

3 Bayesian EI

Given that the estimated kriging variance in classic EGO is biased and repeated optimization problems in bootstrapped EGO are difficult, we take instead a bayesian approach to calculate the expected improvement while considering the uncertainty of the parameters in the predictions. The posterior predictive distribution of $Y(x)|\mathbf{Y}$ is then used to calculate the expected improvement through its original definition given in (2).

The model in equation (1) is assumed. Furthermore, we assume that the stochastic process $Z(x)$ is isotropic, that is, the covariance between $Z(x^i)$ and $Z(x^j)$ depends only upon the distance between the points $d(x^i, x^j)$ and not on the direction of the vector joining them [3]. We use an exponential covariance function to model $Cov(Z(x^{(i)}), Z(x^{(j)}))$ which is popular in the metamodeling literature. This makes $\Sigma_{ij} = Cov(Y(x^{(i)}), Y(x^{(j)})$ equal to

$$\Sigma_{ij} = \begin{cases} \sigma^2 \exp\left(-\phi d(x^i, x^j)\right) & \text{if } d(x^i, x^j) > 0 \\ \sigma^2 + \tau^2 & \text{otherwise} \end{cases} \tag{7}$$

where $d(x^i, x^j)$ is the Euclidean distance between x^i and x^j. Given the above model, the kriging parameter vector is $\Theta = (\mu, \phi, \sigma^2, \tau^2)$ where $\phi, \sigma^2, \tau^2 > 0$. For the Gaussian Process model above, the likelihood is

$$L(\Theta) = (2\pi)^{-n/2} |\Sigma|^{-1/2} \exp\left(\frac{-1}{2}(\mathbf{Y} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{Y} - \mu\mathbf{1})\right) \tag{8}$$

where X is some given initial design and $\mathbf{1}$ is $n \times 1$ vector of ones.

Given a set of prior distributions $\pi(\cdot)$ for the parameters of the model, the posterior distribution $p(\Theta|\mathbf{Y})$ which is proportional to the likelihood multiplied by the priors, can be derived. We then need to find the posterior predictive distribution of $Y(x)|\mathbf{Y}$. In order to do this, first note that

$$\begin{bmatrix} \mathbf{Y} \\ Y(x) \end{bmatrix} \sim N\left(\mu\mathbf{1}, \begin{bmatrix} \Sigma & \gamma \\ \gamma^T & \sigma^2 + \tau^2 \end{bmatrix}\right)$$

where $\mathbf{1}$ is now an $(n+1) \times 1$ vector of ones and γ is $n \times 1$ vector of covariances between $Y(x)$ and the elements of $\mathbf{Y}$. Given a well-known property of the multivariate Gaussian distribution, we obtain the posterior predictive density:

$$p(Y(x)|\mathbf{Y}, \Theta) = N\left(\mu + \gamma^T \Sigma^{-1}(\mathbf{Y} - \mu\mathbf{1}), \sigma^2 + \tau^2 - \gamma^T \Sigma^{-1} \gamma\right). \quad (9)$$

3.1 Definition of Prior Distributions

We use a *normal* prior for μ to allow this parameter to have positive or negative values and a *lognormal* prior for ϕ, σ^2 and τ^2 since these parameters can take only positive values. These distributions are simple to interpret and tune to make them as non-informative as one may wish.

$$\begin{aligned} \mu &\sim N(\mu_\mu, \sigma_\mu^2) \\ \phi &\sim logN(\mu_\phi, \sigma_\phi^2) \\ \sigma^2 &\sim logN(\mu_{\sigma^2}, \sigma_{\sigma^2}^2) \\ \tau^2 &\sim logN(\mu_{\tau^2}, \sigma_{\tau^2}^2) \end{aligned}$$

These priors were set quite none-informative to allow the posterior distributions to be solely influenced by the data. Hence, the variance parameter of the normal prior for μ, σ_μ^2, was set to 10^{40} and the variance parameters for *lognormal* priors, $\sigma_\phi^2, \sigma_{\sigma^2}^2$ and $\sigma_{\tau^2}^2$, were all set to 100.

Based on the given prior distributions, the joint posterior distribution of the parameters is derived in Appendix A. Furthermore, the full conditional distributions for ϕ, σ^2 and τ^2 are provided in Appendix A. Since these distributions are not known probability distributions, we need to use the Metropolis-Hastings algorithm to sample from the posterior distributions of these three parameters [2]. However, given the normal prior for μ, its full conditional can be written as a normal distribution (as shown in Appendix B); therefore, Gibbs sampling is used for this parameter [2].

3.2 The Bayesian EI Algorithm

- Design the initial experiment $\mathbf{X}$ and evaluate the function (i.e., run the computer code) at the design points to get $\mathbf{Y}$
- **While** stopping criteria for Bayesian EGO algorithm not met **do**
    1. run the MCMC and sample from the posterior distribution of $\Theta|\mathbf{Y}$
    2. **while** stopping criteria for optimization procedure not met **do**

```
            i. at any new point $x_{new}$, proposed by the optimization procedure, sample from the posterior predictive distribution of $Y(x)|\mathbf{Y}$
            ii. evaluate the Bayesian EI at $x_{new}$ directly using equation (2)
        3. **end while**
        4. let the final solution of optimization procedure to be $x^*$. Add $x^*$ to the bottom of $\mathbf{X}$
        5. run the function at $x^*$ to get $y^*$ and add it to the bottom of $\mathbf{Y}$
• **End while**
```

4 Numerical Results for Some Test Functions

In this section, the performance of the bayesian EGO is studied using four test functions which are included in [13] and the results are compared with both the *classic EGO* and the *bootstrapped EGO*. The test functions include a one dimensional Forrester function, two dimensional "six-hump camel-back" function, three dimensional "Hartmann-3" function and the six dimensional "Hartmann-6" function.

The initial design $\mathbf{X}$ for each of the tests is a space filling design in the function's domain [18]. This was the same design used for all the three algorithms (classic EGO, bootstrapped EGO and bayesian EGO). The stopping criterion is either reaching *maximum number of allowable iterations* (which is different for each of the four test functions and are set equal to those used in [13]) or attaining the expected improvement threshold which is set to e^{-20} (again, similar to [13]). Furthermore, to have a sense on variability, the algorithm is replicated 5 times for each test function where each replication is starting from a different pseudorandom number seed. All of the calculations are done in MATLAB and the codes are available at `http://www2.ie.psu.edu/Castillo/research/EngineeringStatistics/software.htm`.

4.1 Computational Details

In this section, we describe some implementation details of the MCMC sampling routine.

- As discussed in the previous section, we use Gibbs sampling for μ and Metropolis-Hastings for the remaining three parameters of the covariance function, namely ϕ, σ^2 and τ^2. To determine the length of the Markov chains, visual inspection of the trend plot, a plot of the variance and the mean of the variables and also plot of the standard deviation of the mean were considered (see Appendix C for a sample of these monitoring plots). The chains should be long enough so that the trend plot shows stationarity and the other three plots are stabilized [16].

- In addition to sample inspection, we calculate the Effective Sample Size (ESS) to determine the number of independent samples out of the total generated sample. The ESS mainly depends on autocorrelation structure within the generated sample: the more autocorrelated the samples are, the longer chains are required to achieve a given number of independent samples. The ESS was then used to perform "thinning" of the chains so that the remaining samples are no longer autocorrelated [16].

- To sample from the posterior distributions of the three parameters ϕ, σ^2 and τ^2, we need to use the Metropolis-Hastings algorithm; therefore, a proposal distribution is required for each of them. We have used a log-normal distribution with $\mu_{proposal}$ equal to the logarithm of the parameter value at the previous MCMC iteration and $\sigma^2_{proposal}$ was initially set such that there is a good balance between acceptance rate (around 40%) and rapid tail off of the autocorrelation function [16]. The $\sigma^2_{proposal}$ are not necessarily the same across different parameters and also the test functions. Furthermore, we incorporate an adaptive Metropolis-Hastings technique, proposed by Haario et al. [9], to change $\sigma^2_{proposal}$ along the MCMC iterations.

- To calculate the MCMC variance of the parameters, batch means [16] were used given any possible autocorrelation exists in the Markov chains.

- The number of posterior samples was set equal to $min(ESS, 1000)$ in all test functions. Note that the samples from the posterior distribution of the parameters are changing at each new point x_{new} proposed by the optimization routine.

- As noted later, the optimization routine is simply a search routine over a set of points defined by a fine grid or a space filling design [18] similar to Kleijnen et al. [13].

- The stopping criterion is either a predefined *maximum number of allowable iterations* or arriving at bayesian EI less than a predefined threshold value, e^{-20}, the same as in [13].

4.2 The Forrester Function

The first function that is evaluated is the one dimensional Forrester function:

$$y(x) = (6x-2)^2 \sin(12x-4) \quad 0 \leq x \leq 1$$

This function has one local minimum at $x_L = 0.01$ and one global minimum at $x_G = 0.7572$ where $y(x_G) = -6.0207$. The same initial design is used as [13] that is [0,0.5,1]. The optimization routine is a search over a grid with the step size equal to 0.01 in $(0, 1)$. Similar to [13], maximum number of allowable iterations is set to 8. Table 1 illustrates the results.

Table 1: Comparison of Classic EGO, Bootstrapped EGO and the proposed Bayesian EGO methods for the 1-D Forrester Function (Bootstrapped EGO results are included from [13])

	Rep.	x_{opt}	y_{opt}	n_{opt}	n_{tot}	d
Classic EGO	1	0.76	-6.017	10	11	0.0028
Boots. EGO	1	0.76	-6.017	9	11	0.0028
	2	0.76	-6.017	10	11	0.0028
	3	0.76	-6.017	9	10	0.0028
	4	0.76	-6.017	10	10	0.0028
	5	0.76	-6.017	8	10	0.0028
Bayes. EGO	1	0.76	-6.017	8	11	0.0028
	2	0.76	-6.017	7	11	0.0028
	3	0.76	-6.017	9	11	0.0028
	4	0.78	-5.7282	10	11	0.0228
	5	0.76	-6.017	11	11	0.0028

Table 1 shows the coordinate of the optimal solution (x_{opt}) at each replication, the function value at that solution (y_{opt}), the iteration number which result into the optimal solution (n_{opt}), the total number of iterations until stopping (n_{tot})and finally the Euclidean distance between the optimal solution and the true global minimum (d). As we can see, the bayesian EGO method finds the true global minimum in four replications similar to the classic and the bootstrapped EGO. Furthermore, the bayesian EGO and the bootstrapped EGO seems to be almost the same in the speed of finding the optimum (based on n_{opt}) while both of them are faster than the classic EGO. Notice that n_{opt} and n_{tot} include the initial design points, as well. Figure 1 illustrates the mean of the posterior predictive distribution, the variance of the posterior predictive distribution and also the bayesian EGO in $(0, 1)$ after 8 iterations of the algorithm. Note that the variance is higher at locations where the density of the observed data points are lower and vice versa. If there was a ninth iteration, it would be at the point which has the maximum bayesian EI in the plot.

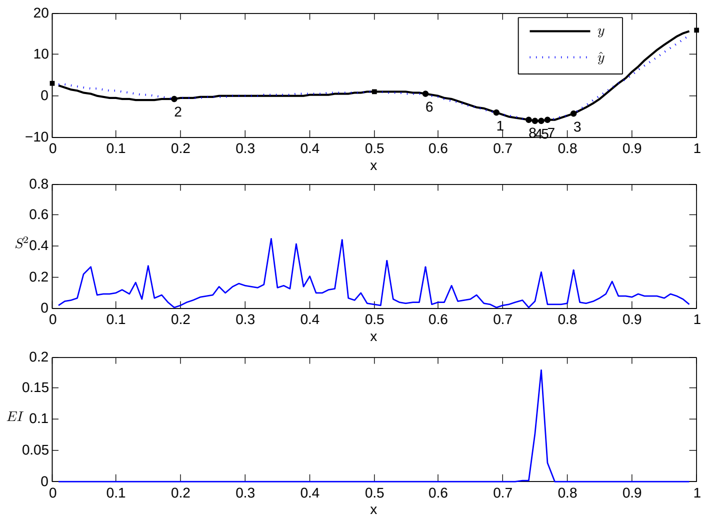


Figure 1: **Top:** The Forrester function (y) and the mean of the posterior predictive distribution ($\widehat{y}$). The squares are the initial design points and the round dots are the points proposed by bayesian EGO algorithm. **Middle:** The variance of the posterior predictive distribution. **Bottom:** The bayesian expected improvement

4.3 The Six-Hump Camel-Back Function

The six-hump camel-back function is defined as

$$y(x_1,x_2) = 4x_1^2 - 2.1x_1^4 + x_1^6/3 + x_1x_2 - 4x_2^2 + 4x_2^4 \quad -2 \leq x_1 \leq 2, -1 \leq x_2 \leq 1$$

In the given domain the function has two global minima which are $x_G^1 = (0.089842, -0.712656)$ and $x_G^2 = (-0.089842, 0.712656)$ with the function value equal to -1.031628. Also, the function has two local minima.

Similar to [13], the initial design is a type of space filling design called maximin Latin Hypercube Sampling (LHS) with 21 points [18]. Furthermore, the optimization routine is a search over 200 candidate points generated from a maximin LHS design in the above domain. The maximum number of allowable iterations is set to 40 which is the same as [13]. Table 2 shows the results of the three methods.

Table 2: Comparison of Classic EGO, Bootstrapped EGO and the proposed Bayesian EGO methods for the 2-D Six-hump camel-back function(Bootstrapped EGO results are included from [13])

	Rep.	x_{opt}	y_{opt}	n_{opt}	n_{tot}	d
Classic EGO	1	(-0.0302,0.7688)	-0.9863	31	41	0.0819
Boots. EGO	1	(0.0302,-0.7688)	-0.9863	29	43	0.0819
	2	(-0.0302,0.7688)	-0.9863	29	41	0.0819
	3	(-0.0302,0.7688)	-0.9863	29	42	0.0819
	4	(0.0302,-0.7688)	-0.9863	29	42	0.0819
	5	(0.0302,-0.7688)	-0.9863	29	43	0.0819
Bayes. EGO	1	(-0.0971,0.7333)	-1.0280	51	61	0.0219
	2	(0.0864,-0.7256)	-1.0301	61	61	0.0134
	3	(-0.0926,0.7065)	-1.0313	46	61	0.0067
	4	(0.0980,-0.7051)	-1.0308	32	61	0.0111
	5	(0.1086,-0.7188)	-1.0301	38	61	0.0198

The bayesian EGO method unanimously achieved better solutions compared to the classic EGO and the bootstrapped EGO considering that it uses all of its 40 allowable iterations. The distance of the final solutions from the global minima are lower for all of the replications compared to the other two approaches. However, both the classic and the bootstrapped EGO were faster than the bayesian EGO in finding their final solutions except for replication 3.

Figure 2 illustrates the contour plot of the six-hump camel-back function and its prediction by the mean of the posterior predictive distribution. The square black points are the 21 initial design points and the round red points are the final solutions of the proposed algorithm at each iteration. The final solutions of the algorithm are almost concentrated around the two global minima which confirms the capability of the algorithm to locate both of the global minima.

Figure 3 shows the contours of the bayesian expected improvement for the six-hump camel-back function. Notice that the expected improvement function is maximized around the two global minima.

4.4 The Hartmann-3 Function

The Hartmann-3 is a three dimensional function defined as

$$y(x_1, x_2, x_3) = -\sum_{i=1}^{4} \alpha_i \exp\left[-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right] \quad 0 \le x_i \le 1, i = 1, 2, 3$$

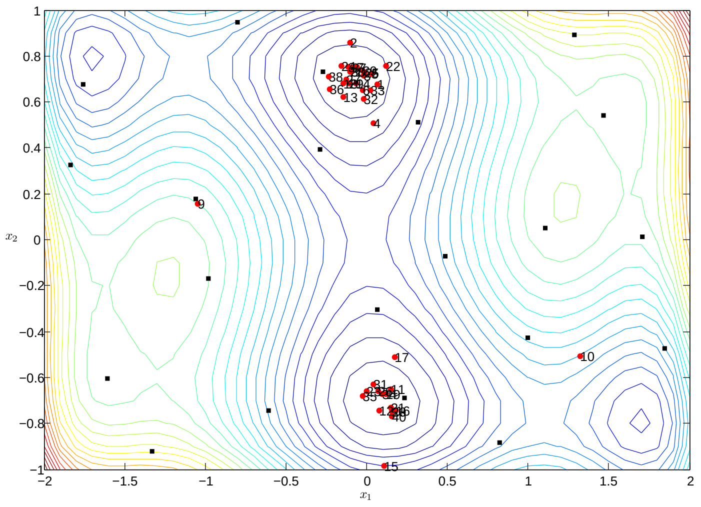
where $\alpha = (1.0, 1.2, 3.0, 3.2)$ and A_{ij} and P_{ij} are given in Table 3. The function has one global minimum at $x_G = (0.114614, 0.555649, 0.852547)$ with function value equal to -3.86278 and also three local minima.

Table 3: parameters A_{ij} and P_{ij} for the Hartmann-3 Function

A_{ij}			P_{ij}		
3.0	10	30	0.36890	0.11700	0.26730
0.1	10	35	0.46990	0.43870	0.74700
3.0	10	30	0.10910	0.87320	0.55470
0.1	10	35	0.03815	0.57430	0.88280

The initial design is a maximin LHS design with 30 points. To perform the optimization, the bayesian EGO is evaluated over a maximin LHS design with 300 points and the maximum is the final solution. Furthermore, the maximum number of iterations is set to 35 (all of the settings are similar to [13]). Table 4 shows the results for the Hartmann-3 function.

The final solutions of all of the replications of the bayesian EGO method are lower than those of the classic and the bootstrapped EGO. However, both


(a)

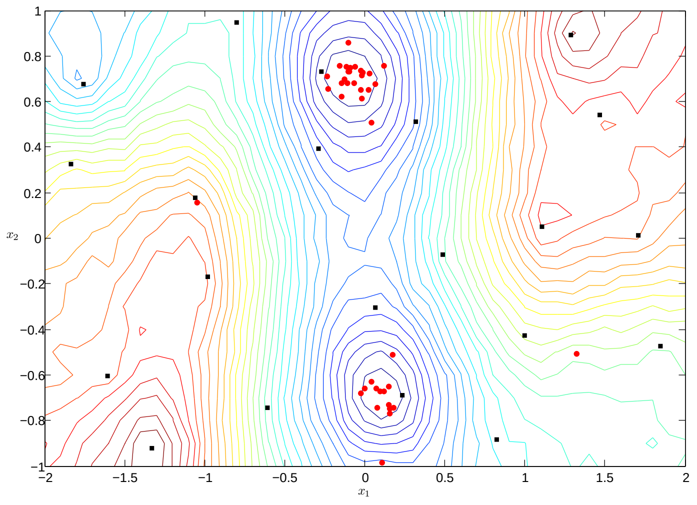

(b)

Figure 2: **(a)** Contour plot of the six-hump camel-back function. The square (black) points show the initial LHS design and the round (red) points are the final solutions of the bayesian EGO algorithm. The number beside each round (red) point is the iteration number of that solution **(b)** Contour plot of the predicted six-hump camel-back function through the mean of the posterior predictive distribution.

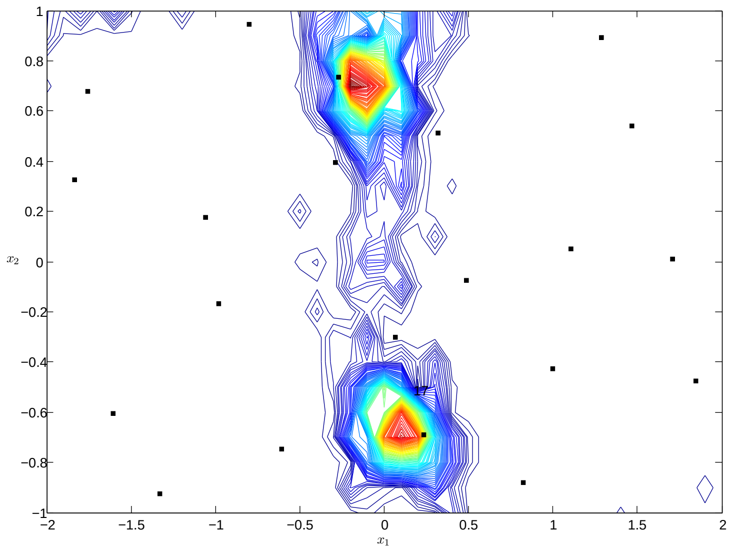


Figure 3: Contour plot of the bayesian expected improvement for the six-hump camel-back function. The square (black) points are the initial LHS design.

Table 4: Comparison of the Classic EGO, the Bootstrapped EGO and the Bayesian EGO methods for the 3-D Hartmann-3 function(The Bootstrapped EGO results are included from [13])

	Rep.	x_{opt}	y_{opt}	n_{opt}	n_{tot}	d
Classic EGO	1	(0.2088,0.5465,0.8767)	-3.7956	44	65	0.0977
Boots. EGO	1	(0.2088,0.5465,0.8767)	-3.7956	34	65	0.0977
	2	(0.2088,0.5465,0.8767)	-3.7956	34	65	0.0977
	3	(0.2088,0.5465,0.8767)	-3.7956	41	65	0.0977
	4	(0.2088,0.5465,0.8767)	-3.7956	34	65	0.0977
	5	(0.2088,0.5465,0.8767)	-3.7956	44	65	0.0977
Bayes. EGO	1	(0.0780,0.5615,0.8628)	-3.8510	35	65	0.0385
	2	(0.2752,0.5618,0.8643)	-3.8330	60	65	0.1611
	3	(0.1639,0.5637,0.8431)	-3.8501	34	65	0.0509
	4	(0.1388,0.5818,0.8553)	-3.8381	53	65	0.0357
	5	(0.0083,0.5598,0.8506)	-3.8548	61	65	0.1064

the classic and the bootstrapped EGO found their final solutions faster than the bayesian EGO method except for one replication.

Figure 4 shows the location of the final solutions found by the bayesian EGO algorithm (round red points) and the global minimum (star shaped green point).

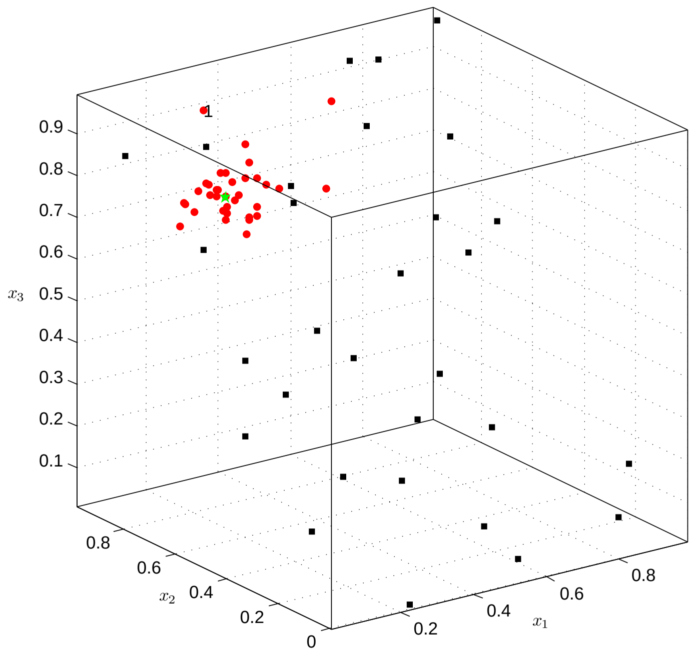


Figure 4: Square (black) points are the initial LHS design, round (red) points are the solutions of the bayesian EGO algorithm and the star (green) shows the global minimum of the Hartmann-3 function.

4.5 The Hartmann-6 Function

The last test function which is evaluated is the Hartmann-6 function with six variables. It is defined as

$$y(x_1,...,x_6) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} \alpha_{ij}(x_j - p_{ij})^2\right] \quad 0 \leq x_i \leq 1, i = 1,...,6$$

where $c = (1.0, 1.2, 3.0, 3.2)$ and α_{ij} and p_{ij} are given in Table 5. This function has a global minimum at $x_G = (0.2017, 0.1500, 0.4768, 0.2753, 0.3116, 0.6573)$ with function value equal to -3.32237 and five local minima.

Table 5: Parameters α_{ij} and p_{ij} of Hartmann-6 Function

α_{ij}	10.0	3.0	17.0	3.5	1.7	8.0
	0.05	10.0	17.0	0.1	8.0	14.0
	3.0	3.5	1.7	10.0	17.0	8.0
	17.0	8.0	0.05	10.0	0.1	14.0
p_{ij}	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

The initial design is a maximin LHS with 51 points. Furthermore, the set of candidate points in the search optimization routine is from a maximin LHS design with 500 points in the function space. Finally, the maximum number of allowable iterations is set to 50 (all of these settings are similar to [13]). Table 6 compares the results of the bayesian EGO for the Hartmann-6 function versus the classic and the bootstrapped EI methods.

Note that the y_{opt} for the bayesian EGO algorithm is lower in all of the replications compared to the bootstrapped and the classic EGO methods. Furthermore, the bayesian EGO is faster in two replications (replications 4 and 5) in finding its final solution compared to the bootstrapped and the classic EGO methods.

4.6 Computation Time

In this section, we give insight on the computational time of the bayesian EGO algorithm. Duration of each run of the bayesian EGO algorithm depends mainly on the following factors: 1. Number of input variables - dimension (Dim) 2. Initial design size (IDS) 3. Maximum number of allowable iterations (MNAI) 4. MCMC Chains length (CL) 5. Number of the posterior samples used to evaluate the posterior predictive distribution at a given location (NPS). Table 7 shows some statistics on computational time of a single iteration for the four test functions on a 3.60 GHz Intel pentium processor with 4.00 GB of RAM.

The iteration times increase as the number of rows of **X** and **Y** go up. The iteration times are the mean iteration time within one replication of the

Table 6: Comparison of the Classic EGO, the Bootstrapped EGO and the Bayesian EGO methods for the 6-D Hartmann-6 function(the Bootstrapped EGO results are included from [13])

	Rep.	x_{opt}	y_{opt}	n_{opt}	n_{tot}	d
Classic EGO	1	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	79	101	1.0442
Boots. EGO	1	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	92	101	1.0442
	2	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	89	101	1.0442
	3	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	78	101	1.0442
	4	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	86	101	1.0442
	5	(0.3535,0.8232,0.8324,0.4282,0.1270,0.0013)	-2.3643	92	101	1.0442
Bayes. EGO	1	(0.1570,0.1386,0.4755,0.3421,0.2334,0.6297)	-2.8681	94	101	0.1160
	2	(0.2113,0.0702,0.4313,0.3100,0.2410,0.6060)	-2.9112	83	101	0.1317
	3	(0.0933,0.2175,0.3270,0.2391,0.3446,0.5733)	-2.6851	95	101	0.2196
	4	(0.1321,0.1530,0.5626,0.3280,0.2955,0.6399)	-3.0836	70	101	0.1246
	5	(0.2290,0.0795,0.2978,0.2877,0.2405,0.6839)	-2.8464	70	101	0.2091

Table 7: Computational time of a single iteration for the four test functions on a 3.60 GHz Intel pentium processor with 4.00 GB of RAM

						Iteration Time (sec)			
Function	Dim	IDS	MNAI	CL	NPS	Min	Mean	Max	Std.
Forrester	1	3	8	1e4	200	57.56	58.85	61.04	1.48
Six-Hump	2	21	40	1e4	200	100.57	107.15	113.62	5.80
Hartmann-3	3	30	35	1e4	200	114.10	122.68	130.51	5.97
Hartmann-6	6	51	50	3e4	1e3	600.62	659.49	732.24	55.42

bayesian EGO algorithm and the provided statistics are over five different replications.

5 Application of Bayesian EGO to a Stochastic Inventory Simulation

This section contains implementation of the proposed algorithm for a stochastic function from the inventory systems literature. The example which is discussed in Law and Kelton [14] is about finding the optimal reorder point (s) and the maximal holding quantity (S) in an (s, S) inventory policy. The objective function is the expected total cost per month which is the sum of the ordering cost, the holding cost and the shortage cost. The company reviews it's inventory at the beginning of each month and decides how much to order. Based on the (s, S) policy the order quantity is

$$Z = \begin{cases} S - I & \text{if } I < s \\ 0 & \text{if } I \geq s \end{cases}$$

where I is the inventory level and Z is the order quantity at the beginning of each month. The time between demands are i.i.d. exponential random variables with mean of 0.1 month. Furthermore, the size of the demands, D, are i.i.d. random variable with the following probability mass function:

$$D = \begin{cases} 1 & \text{w.p. } \frac{1}{6} \\ 2 & \text{w.p. } \frac{1}{3} \\ 3 & \text{w.p. } \frac{1}{3} \\ 4 & \text{w.p. } \frac{1}{6} \end{cases}$$

Finally, the supplier's *lead time* is a uniform random variable between 0.5 and 1 month. Each order has a fixed setup cost of $K = \$32$ and a linear incremental cost of $i = \$3$ per item (if the order quantity is zero then the setup cost is also zero). The holding cost is $h = \$1$ per item per month and the shortage cost is $p = \$5$ per item per month.

Optimal (s, S) inventory policy has an extensive literature, especially, through using the dynamic programming techniques both in finite and infinite time horizon. Here, we want to find the optimal policy in infinite time horizon such that the optimal reorder point (s) and the maximal holding quantity (S) do not change over time. Given that the lead time is stochastic, there does not exist any theoretical result for an optimal (s, S) inventory policy. Actually, this is the reason why the simulation optimization procedures are so much popular for the inventory models with stochastic lead time. The closet theoretical work that was found for this inventory problem is by Ehrhardt [7] which is based on the author's other work with fixed lead time [6] using power approximation. We compare our results with this method.

The initial design that we used is a maximin LHS with 21 points and the maximum number of iteration is set to 25 (similar to the 2-dimensional six-hump camel-back function). A realization of the stochastic process along with the bayesian prediction through the mean of the posterior predictive distribution is illustrated in Figure 5.

The posterior predictive mean seems to be able to predict the stochastic function to a reasonable extent. The optimal solutions based on the theoretical approximation method of Ehrhardt and our proposed bayesian EGO method are provided in Table 8. Furthermore, the inventory simulation model was ran 1000 times for each solution and the corresponding mean, standard deviation and 95% confidence intervals are reported.

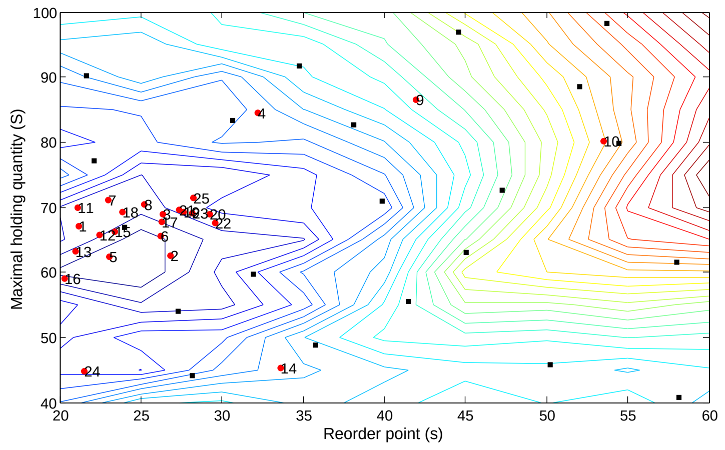
Table 8: The mean, the standard deviation and the 95% CI for total inventory costs based on 1000 replications of the inventory simulation model at the optimal solutions

Method	(s^*, S^*)	Mean	Std.	95% CI
Ehrhardt	(39.72,79.03)	125.99	2.35	(125.85,126.14)
Bayesian EGO	(24.84,61.65)	118.91	3.40	(118.70,119.12)

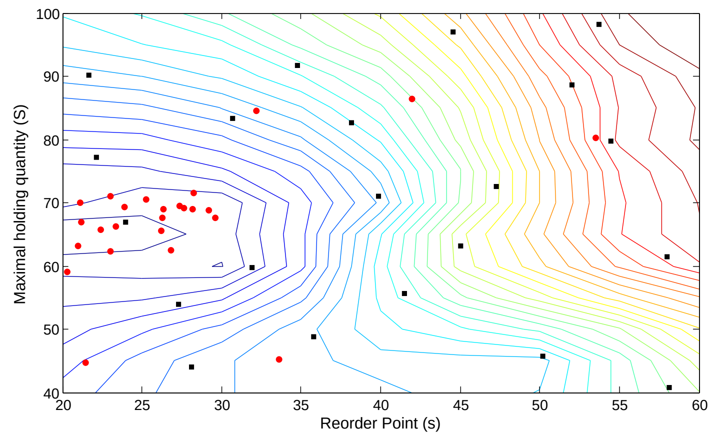
Figure 6 shows the box-plots of the total inventory costs for each inventory policy. The bayesian EGO method came up with a solution with lower inventory costs compare to the Ehrhardt method.

6 Concluding Remarks

In this paper a fully bayesian implementation of the EGO method was presented. Instead of using a plug-in estimator for the kriging variance which underestimates the true variance, or using a bootstrapped estimate of this variance which entails repeated difficult optimizations, a bayesian expected improvement was proposed and embedded within the EGO algorithm for optimization of unknown and non-convex functions. The performance of our approach was then compared with the classic and the bootstrapped EGO methods for four different deterministic test functions from the literature. The function values at final solutions, distance of the final solutions from the true global optimum(s) and the speed of achieving the final solu-



(a)



(b)

Figure 5: **(a)** Contour plot of a realization of the stochastic cost function. The square (black) points are the initial LHS design and the round (red) points are the solutions of the bayesian EGO algorithm. The number beside each red dot is the iteration number of that solution. **(b)** Contour plot of the predicted simulation model through the mean of the posterior predictive distribution

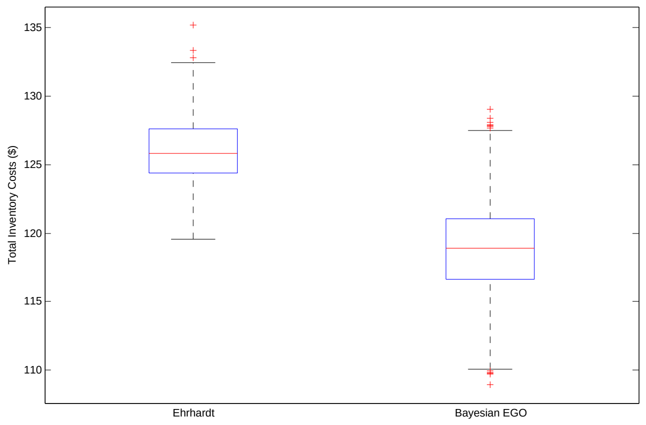


Figure 6: Total Inventory cost box-plots based on running the inventory simulation model at the optimal solution given by each inventory policy for 1000 replications

tions were then compared across the three different methods. In general, bayesian EGO showed to find solutions with better function values and locations closer to the global optimum(s) especially for higher dimensional functions. However, on average, the bayesian EGO showed to be slower in finding the optimal solution compared to the classic and the bootstrapped EGO methods. Speeding up bayesian EGO is therefore a matter of further research.

Furthermore, the bayesian EGO approach was implemented for a stochastic inventory cost function to find the optimal reorder point (s) and the maximal holding quantity (S) in an (s, S) inventory policy. The performance of the proposed approach was compared with the theoretical power approximation method of Ehrhardt [7]. The results showed the advantage of the bayesian EGO method in optimizing the stochastic simulation model as compared to the theoretical power approximation method.

Appendix A. Posterior and Full Conditional Distributions

The posterior distribution is

$$
\begin{aligned}
p(\mu, \phi, \sigma^2, \tau^2 | \mathbf{Y}) \propto\ & (2\pi)^{-n/2} |\Sigma|^{-1/2} \exp\left(\frac{-1}{2} (\mathbf{Y} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{Y} - \mu\mathbf{1}) \right) \\
& \times (2\pi\sigma_\mu^2)^{-1/2} \exp\left(\frac{(\mu - \mu_\mu)^2}{-2\sigma_\mu^2} \right) \\
& \times \phi^{-1} (2\pi\sigma_\phi^2)^{-1/2} \exp\left(\frac{(\ln\phi - \mu_\phi)^2}{-2\sigma_\phi^2} \right) \\
& \times (\sigma^2)^{-1} (2\pi\sigma_{\sigma^2}^2)^{-1/2} \exp\left(\frac{(\ln\sigma^2 - \mu_{\sigma^2})^2}{-2\sigma_{\sigma^2}^2} \right) \\
& \times (\tau^2)^{-1} (2\pi\sigma_{\tau^2}^2)^{-1/2} \exp\left(\frac{(\ln\tau^2 - \mu_{\tau^2})^2}{-2\sigma_{\tau^2}^2} \right)
\end{aligned}
$$

which is basically the likelihood multiplied by the priors for μ, ϕ, σ^2 and τ^2. Given the joint posterior distribution of the parameters, the full conditional distribution of ϕ, σ^2 and τ^2 can be written as follows:

$$
p(\phi | \mu, \sigma^2, \tau^2, \mathbf{Y}) = |\Sigma|^{-1/2} \exp\left(\frac{-1}{2} (\mathbf{Y} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{Y} - \mu\mathbf{1}) \right) (\phi)^{-1} \exp\left(\frac{(\ln\phi - \mu_\phi)^2}{-2\sigma_\phi^2} \right)
$$

$$
p(\sigma^2 | \mu, \phi, \tau^2, \mathbf{Y}) = |\Sigma|^{-1/2} \exp\left(\frac{-1}{2} (\mathbf{Y} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{Y} - \mu\mathbf{1}) \right) (\sigma^2)^{-1} \exp\left(\frac{(\ln\sigma^2 - \mu_{\sigma^2})^2}{-2\sigma_{\sigma^2}^2} \right)
$$

$$
p(\tau^2 | \mu, \phi, \sigma^2, \mathbf{Y}) = |\Sigma|^{-1/2} \exp\left(\frac{-1}{2} (\mathbf{Y} - \mu\mathbf{1})^T \Sigma^{-1} (\mathbf{Y} - \mu\mathbf{1}) \right) (\tau^2)^{-1} \exp\left(\frac{(\ln\tau^2 - \mu_{\tau^2})^2}{-2\sigma_{\tau^2}^2} \right)
$$

Note that covariance matrix Σ contains the covariance parameters ϕ, σ^2 and τ^2.

Appendix B. Gibbs Sampling

We show that the full conditional distribution of μ is a normal distribution. By completing the square we have

$$
\begin{aligned}
p(\mu|\phi,\sigma^2,\tau^2,\mathbf{Y}) &\propto \exp\left(\frac{(\mathbf{Y}-\mu\mathbf{1})^T\Sigma^{-1}(\mathbf{Y}-\mu\mathbf{1})}{-2}\right)\exp\left(\frac{(\mu-\mu_\mu)^2}{-2\sigma_\mu^2}\right)\\
&= \exp\left(\frac{-1}{2}(-\mu\mathbf{1}^T\Sigma^{-1}\mathbf{Y}+\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{1}-\mu\mathbf{Y}^T\Sigma^{-1}\mathbf{1})-\frac{1}{2}\left(\frac{\mu^2-2\mu_\mu\mu}{\sigma_\mu^2}\right)\right)\\
&= \exp\left(\frac{-1}{2}\left((\mathbf{1}^T\Sigma^{-1}\mathbf{1}+\frac{1}{\sigma_\mu^2})\mu^2-2(\mathbf{1}^T\Sigma^{-1}\mathbf{Y}+\frac{\mu_\mu}{\sigma_\mu^2})\mu\right)\right)\\
&= \exp\left(\frac{\mu^2-2\left(\frac{\mathbf{1}^T\Sigma^{-1}\mathbf{Y}+\frac{\mu_\mu}{\sigma_\mu^2}}{\mathbf{1}^T\Sigma^{-1}\mathbf{1}+\frac{1}{\sigma_\mu^2}}\right)\mu}{-2\left(\frac{1}{\mathbf{1}^T\Sigma^{-1}\mathbf{1}+\frac{1}{\sigma_\mu^2}}\right)}\right)\\
&= \exp\left(\frac{\left(\mu-\frac{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{Y}+\mu_\mu}{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{1}+1}\right)^2}{-2\left(\frac{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{1}+1}{\sigma_\mu^2}\right)^{-1}}\right)
\end{aligned}
$$

which is a normal distribution $N\left(\frac{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{Y}+\mu_\mu}{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{1}+1}, \left(\left(\frac{\sigma_\mu^2\mathbf{1}^T\Sigma^{-1}\mathbf{1}+1}{\sigma_\mu^2}\right)^{-1/2}\right)^2\right)$.

Appendix C. Checking the Convergence of the MCMC Chains

Figure 7 shows the monitoring plots for μ and ϕ parameters selected at random which are mainly used for determining the MCMC chains length. The plot in the first row is the empirical density function. The second plot is the trend plot which is usually used as a means to check for the convergence to the stationarity distribution. The third plot illustrates the autocorrelation function. As you may see autocorrelation is insignificant for μ; however, the samples of ϕ seems to be autocorrelated. This autocorrelation structure is also verified by looking at the trend plot. High autocorrelation between the samples causes the Effective Sample Size (ESS) to decrease. Note that the "thinning" process eliminates any remaining autocorrelation. The fourth plot shows the variance of the sampled parameters which will become stable (converge to a constant) after reaching the stationarity. The fifth plot demonstrated the mean of the parameter as a function of the sample index. Finally, the last plot shows the standard error of the mean of the parameter which will converge to 0 as the number of samples goes to infinity.

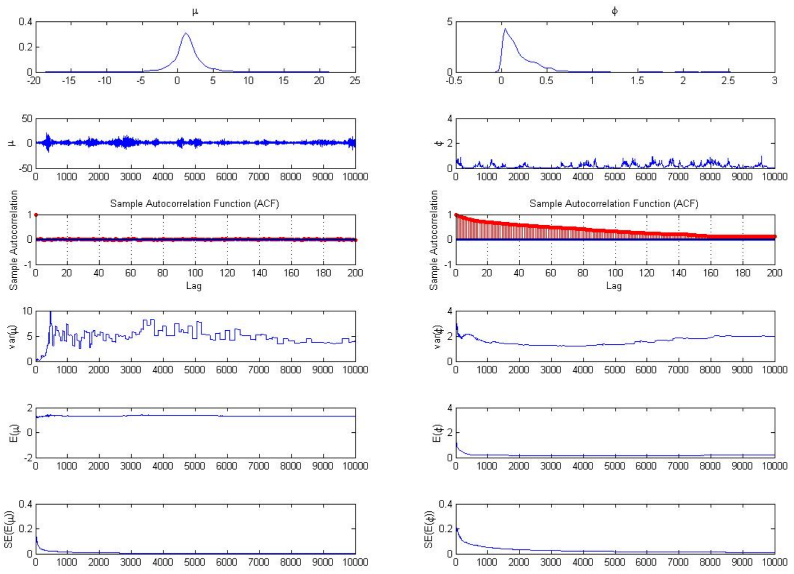


Figure 7: Monitoring plots used to check for convergence of MCMC chains to their stationary distributions - μ (left) and ϕ(right)

References

[1] Banerjee, S., Carlin, B. P., and Gelfand, A. E. *Hierarchical Modeling and Analysis for Spatial Data.* CRC Press, 2004.

[2] Carlin, B. P., and Louis, T. A. *Bayesian methods for data analysis.* CRC Press, 2009.

[3] Cressie, N. A. C. *Statistics for spatial data.* John Wiley and Sons, 1993.

[4] del Castillo, E., and Santiago, E. A matrix-t approach to the sequntial design of optimization experiments. *IIE Transactions 43* (2011), 54–68.

[5] den Hertog, D., Kliejnen, J. P. C., and Siem, A. Y. D. The correct kriging variance estimated by bootstrapping. *Journal of Operational Research Society 57*, 4 (2006), 400–409.

[6] Ehrhardt, R. The power approximation for computing (s,s) inventory policies. *Management Science 25*, 8 (1979).

[7] Ehrhardt, R. (s,s) policies for dynamic inventory model with stochastic lead times. *Operations Research 32*, 1 (1984), 121–132.

[8] Ginsbourger, D., and Riche, R. L. Towards GP-based optimization with finite time horizon. *HAL-EMSE 00424309* (2009).

[9] Haario, H., Saksman, E., and Tamminen, J. An adaptive metropolis algorithm. *Bernouli 7*, 2 (2001), 223–242.

[10] Huang, D., Allen, T. T., Notz, W. I., and Zeng, N. Global optimization of stochastic black-box systems via sequential kriging metamodels. *Journal of Global Optimization 34* (2006), 441–466.

[11] Jones, D. R. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization 21* (2001), 345–383.

[12] Jones, D. R., Schonlau, M., and Welch, W. J. Efficient global optimization of expensive black-box functios. *Journal of Global Optimization 13*, 4 (1998), 455–492.

[13] Kleijnen, J. P. C., van Beers, W., and van Nieuwenhuyse, I. Expected improvement in efficient global optimization through bootstrapped kriging. *Journal of Global Optimization 54* (2012), 59–73.

[14] Law, A. M., and Kelton, D. *Simulation modeling and analysis.* McGraw-Hill, 2000.

[15] Mockus, J., Tiesis, V., and Zilinskas, A. *Towards Global Optimisation.* North Holland, Amsterdam, 1978, ch. The application of bayesian methods for seeking the extremum, pp. 117–129.

[16] Robert, C., and Casella, G. *Introducing Monte Carlo Methods with R.* Springer, 2010.

[17] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. Design and analysis of computer experiments. *Statistical Science 4*, 4 (1989), 409–435.

[18] Santner, T. J., Williams, B. J., and Notz, W. *The Design and Analysis of Computer Experiments.* Springer, 2003.

[19] Sasena, M. J., Papalambros, P., and Goovaerts, P. Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering optimization 34* (2002), 263–278.

[20] Simpson, T. J., Koch, P., and Allen, J. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers 17* (2001), 129–150.

[21] Sjostedt De Luna, S., and Young, A. The bootstrap and kriging prediction intervals. *Scandinavian Journal of Statistics 30* (2003), 175–192.

[22] Williams, B. J., Santner, T. J., and Notz, W. I. Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica 10* (2000), 1133–1152.

[23] Williams, B. J., Santner, T. J., Notz, W. I., and Lehman, J. S. Sequential design of computer experiments for constrained optimization. *Statistical Modeling and Regression Structures* (2010), 449–472.