Improved Gaussian Process Acquisition for Targeted Bayesian Optimization

Peter Mitic

***Abstract*—A black-box optimization problem is considered, in which the function to be optimized can only be expressed in terms of a complicated stochastic algorithm that takes a long time to evaluate. The value returned is required to be sufficiently near to a target value, and uses data that has a significant noise component. Bayesian Optimization with an underlying Gaussian Process is used as an optimization solution, and its effectiveness is measured in terms of the number of function evaluations required to attain the target. To improve results, a simple modification of the Gaussian Process 'Lower Confidence Bound' (LCB) acquisition function is proposed. The expression used for the confidence bound is squared in order to better comply with the target requirement. With this modification, much improved results compared to random selection methods and to other commonly used acquisition functions are obtained.**

***Index Terms*—Acquisition function, bayesian optimization, gaussian process, loss distribution, monte carlo.**

I. Introduction

We consider the problem of optimizing the value of a black-box function $f(x)$, where the domain of x is a random variable defined on a closed real interval I. Most generally, it may be assumed that nothing is known about the function f, other than what values are used for its inputs, and that a process exists to produce outputs. Further, it is assumed that evaluation of $f(x)$ is, in some sense, "expensive". It may be almost intractable, or certain optimization techniques may not be applicable, or it may involve noise, or it may take a long time to evaluate. Search methods often work in those circumstances, although they can be slow. If the interval I is finite, a binary search is possible, but only if a relatively low accuracy is required. A second alternative is random search. Bergstra and Bengio [1] show that random search is more efficient than a binary search, and we concur with that view for the case we consider here. However, a long run of searches, as may happen with random search, is something to avoid. Bayesian Optimization (*BO*), in conjunction with an embedded Gaussian Process (*GP*), is now an established and efficient optimization method when applied to black-box functions [2]. In many cases *BO* does, indeed, work well, but in this paper, we highlight problems that arise in two circumstances. First, when f incorporates a significant noise component, and second, when a target is introduced.

Manuscript received July 8, 2020; revised December 12, 2020.

Peter Mitic is with the Dept. of Computer Science, UCL, London, UK (e-mail: p.mitic@ucl.ac.uk).

II. Noisy Black-Box Function Evaluation in Financial Risk Reverse Stress Testing

The specific function $f(x)$ considered in this paper is a Monte Carlo process that calculates an optimal scale factor x such that, given fat-tailed data D, the value-at-risk (*VaR*) of the scaled data $(1+x)D$ is within a pre-determined limit L from a target value V. This problem occurs in the context of financial risk reverse stress testing. The task is then to determine, for the model used to determine *VaR*, which parameter values should be used to attain the required target, which is the stressed *VaR*. The procedures used to calculate *VaR* are non-linear, stochastic and are frequently expressed in the form of algorithms rather then closed-form formulae.

A. Problem Specification

The problem outlined above is expressed as a function optimization in which the function f encapsulates the *VaR* calculation. The interval I is typically [0, 1], and a reasonable value for L is 0.01 (i.e. L is a 1% limit). There are established methods for calculating *VaR*, the most widely applicable of which is the *Loss Distribution Approach* (*LDA*) method [3]. The optimization problem to be solved, with optimal value $\hat{x}$ and a stochastic error term $\varepsilon \sim N\left(0, \omega^2\right)$, is then given by Equation (1).

$$\hat{x} = \arg\min_{x \in I} \left(\left| \frac{f\left((1+x)D\right) + \varepsilon - V}{V} \right| < L \right) \tag{1}$$

To simplify expressions used later in this paper we will write the term $\left| \frac{f\left((1+x)D\right) + \varepsilon - V}{V} \right|$ in Equation (1) in a simpler form $g(x)$, in which the parameters D, V and ε are implied (Equation 1a).

$$\hat{x} = \arg\min_{x \in I} \left(g\left(x\right) < L \right) \tag{1a}$$

Discussion of the implications of the error term are deferred until Section III and until then, the discussion of *BO* concentrates on the error-free case. Although the details of the data and the particular black-box function that we consider here are known, optimization presents particular problems. They are:

1) Gradient search methods cannot be used with f.
2) The data, D, are not time-homogeneous.
3) Each function evaluation takes a long time.
4) The evaluation error must be within the prescribed limit
5) In multi-stage cases, D changes between evaluations.

DOI: 10.7763/IJMO.2021.V11.771

Although the operation of the *LDA* is well known, it cannot be expressed as an explicit or even an implicit function because it can only be described by an algorithm. That algorithm incorporates Monte Carlo random sampling, and the accuracy of the output depends heavily on the number of Monte Carlo iterations used. Longer calculation times result from more Monte Carlo iterations and increasing size of *D*. However, empirical evidence presented here indicates a lack of success when using standard forms of *BO* in this context. We suggest reasons and propose a solution in Section IV.

III. Bayesian Optimisation

The intention of the combined *BO-GP* process is to minimize the number of slow or difficult evaluations of *g*. We will use the term 'expensive' to cover all aspects of evaluations of *g* that are slow, difficult or random. It proceeds as follows.

1) Define an initial set of evaluation points
2) Repeat until converged
   2.1. Define a Bayes prior with the *GP*
   2.2. Supply data and calculate a Bayes likelihood
   2.3. Calculate a Bayes posterior
   2.4. Define sample points to calculate a Bayes predictor
   2.5. Propose a next evaluation point using the *BO Acquisition function* with the predictor

A. Bayesian Optimization: Supporting Literature

Accounts of the *BO* method, including steps in its development, may be found in Rana et al. [4], Rasmussen and Williams [5], and Murphy [6]. The original paper by Mockus [2] proposed the concept that a Gaussian distribution of functions may be used as a proxy for optimizing a function that is difficult to optimize in any other way. Shortly after, those concepts were applied by Mockus and co-workers [7] to real situations, and the concept of an acquisition function using *Expected Improvement* (see Section III) was formalized. Two ideas were central to the original argument. The first was to use a kernel to calculate covariances. The second was a proof that 'expensive' function evaluation at a sequence of evaluation points proposed by the *GP* convergences to a solution of the original optimization problem. The *Probability of Improvement* acquisition function was proposed eleven years later, also by Mockus [8].

The *Confidence Bound* acquisition function was used by Cox and John in 1997 [9], and a significant result for it was proved in 2010 by Srinivas et al [10]. It determined a relationship between the 'expensive' function $g(x)$ and an $(n\text{-}1)^{\text{th}}$-stage (previous historic) approximation to it calculated using a *GP*. The result may be expressed in terms of the expected value $\mu_{n-1}(x)$ and the root mean square error (effectively a standard deviation measure) $\sigma_{n-1}(x)$ of the *GP* at the n^{th} approximation. With high probability, the absolute difference between $g(x)$ and $\mu_{n-1}(x)$ evaluated at *any* point x in the interval I was shown to be bounded by $\sqrt{\beta_n}\sigma_{n-1}(x)$, where β_n is a real number that depends on $\log(n^2)$. That result forms the basis of the *Confidence Bound* acquisition function, which is significant in the analysis presented in this paper.

In 2012 Snoek et al. [11] used a *GP* when training a neural net, and the use of *GP*s in the context of machine learning, including robotics) has been significant since then. A further common use for *BO* using a *GP* is in simulation. A recent account may be found in [12], which contains references to previous work in this area. Other recent application areas are pharmaceuticals [13], particle physics [14], and simulation [15]. The context of this paper is financial risk. Few other risk or financial applications of *BO* appear in the literature to date. Time series modelling is one [16].

B. Gaussian Process: Theory

Rasmussen and Williams [5] define a *GP* as a distribution over functions. Another way of looking at it, is that a *GP* is an infinite collection of random variables, any finite number of which have joint Gaussian distributions. The key to using a *GP* is to condition it on observed function values. Function evaluation is only necessary, at a finite, but arbitrary, set of 'evaluation' points $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$. A Gaussian process generalizes the multivariate normal to *infinite* dimensions. If a Gaussian prior is defined and combined with Gaussian data, both posterior and predictive distributions are Gaussian, and evaluation of them is very fast. In contrast, evaluation of the set $\boldsymbol{y} = \{g(x_1), g(x_2), \ldots, g(x_n)\}$ is slow. The purpose of a *GP* embedded in a *BO* calculation is to propose a next candidate evaluation point x_{n+1}. It does so by replacing 'expensive' evaluations of g by 'cheap' evaluations of normal distributions, and assumes that the joint distribution of the elements of $\boldsymbol{y}$ is multi-variate Normal. The parameters of a *GP* are $\mu(\boldsymbol{x})$ (a vector of function means) and $\boldsymbol{K}$ (a covariance matrix). The latter is often referred to as a *kernel*, since covariances are calculated using a *kernel function*, $k(x_i, x_j)$, where $x_i, x_j \in \boldsymbol{x}$. The mean and covariance functions drive the entire *GP*. Function evaluation is a draw from the Gaussian distribution in Equation 2:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = N\left(\mu(\boldsymbol{x}), \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x})\right) \tag{2}$$

A kernel function should evaluate to approximately 1 if x_i and x_j are close, and to 0 when they are not. An example is the exponential function (Equation 3), in which s is a scale factor.

$$k(x_i, x_j) = e^{-\frac{|x_i - x_j|}{2s^2}} \tag{3}$$

The covariance matrix $\boldsymbol{K}$ has to be amended in the case of noisy function evaluation. This is discussed in Section IV. The posterior for a new set of M points $x_* = \{x_{n+1}, x_{n+2}, \ldots, x_{n+M}\}$ is found from the joint Gaussian distribution of the observed data $\{x, y\}$ and the set $\{x_*, y_*\}$ where $y_* = g(x_*) = \{g(x_{n+1}), g(x_{n+2}), \ldots, g(x_{n+M})\}$ (Equation 4).

$$p\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{pmatrix} \sim N\left(\begin{pmatrix} \mu(\boldsymbol{x}) \\ \mu(\boldsymbol{x}_*) \end{pmatrix}, \begin{pmatrix} \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_*) \\ \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{x}) & \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{x}_*) \end{pmatrix}\right) \tag{4}$$

The predictive distribution for the new points, $g(x_*)$, (Equation 5a) is a marginal of the multi-variate distribution in Equation (4). Hence it is also Gaussian. Its mean is a vector

$\mu(\boldsymbol{x}_*)$ of dimension M and its covariance matrix, $\sigma^2(\boldsymbol{x}_*)$ has dimension $M \times M$. They are given in Equations 5b and 5c respectively).

$$g(x_*) = N\left(\mu(x_*), \sigma^2(x_*)\right) \tag{5a}$$

$$\mu(\boldsymbol{x}_*) = \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_*)\left(\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x})\right)^{-1}\left(\boldsymbol{y} - \mu(\boldsymbol{x})\right) \tag{5b}$$

$$\sigma^2(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{K}(\boldsymbol{x}_*, \boldsymbol{x})\left(\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x})\right)^{-1}\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_*) \tag{5c}$$

The expressions $\mu(\boldsymbol{x}_*)$ and $\sigma^2(\boldsymbol{x}_*)$ in Equations (5a-c) can be used to search for another evaluation point in the interval *I*. That is the *acquisition* stage.

C. The Acquisition Function

Two metrics based on $\mu(\boldsymbol{x}_*)$ and $\sigma^2(\boldsymbol{x}_*)$ are particularly useful: the mean of the elements in vector $\mu(\boldsymbol{x}_*)$ and root mean square error components from $\sigma^2(\boldsymbol{x}_*)$. Full details of the calculation of the latter may be found in Ranjan et al. [17]. Calculating the mean of each element of $\mu(\boldsymbol{x}_*)$ results in a vector of means, $\bar{\boldsymbol{m}}$. Next, denote the entries in the i^{th} column of $\sigma^2(\boldsymbol{x}_*)$ by $\boldsymbol{s}_i$ and estimators of them, obtained from the kernel function, by $\hat{\boldsymbol{s}}_i$. Equations 6a and 6b respectively then give the mean metric, $\bar{\boldsymbol{m}}$, and the root mean square error metric, $\bar{\boldsymbol{s}}$.

$$\bar{\boldsymbol{m}} = \mathbb{E}\left(\mu(\boldsymbol{x}_*)\right) \tag{6a}$$

$$\bar{\boldsymbol{s}} = \left\{\sqrt{\mathbb{E}\left(\left(\boldsymbol{s}_i - \hat{\boldsymbol{s}}_i\right)^2\right)}\right\}_{i=1}^{M} \tag{6b}$$

Both vectors $\bar{\boldsymbol{m}}$ and $\bar{\boldsymbol{s}}$ are inputs to an *acquisition function*, which is the actual predictor of the next evaluation point, x_{n+1}. Three acquisition functions are in common use. *POI* ("Probability of Improvement") [8] calculates the value of the next evaluation point that has a maximum probability of improving the optimization result. *EI* ("Expected Improvement") [7] calculates the value of the next evaluation point that maximizes the expected value that improves the optimization result. The third is *CB* [9, 10] ("Confidence Bound"), usually prepended by *L* ("Lower") if a minimization is required, or by *U* ("Upper") if a maximization is required. We have found that all three fail to yield satisfactory results without amendment. We concentrate on *CB* (*LCB* in particular) because a simple amendment of it gives very satisfactory results (see Section IV). Berk et al. [18] indicate that *EI* is prone to detecting a local rather than a global optimum. A common heuristic is that *EI* is not appropriate for noisy problems because when noise is present, the optimal observation may not correspond to the true optimal function value.

D. The Bayesian Optimal Solution

In all cases (*EI, POI, CB* or any other), the solution to the optimization problem, $\hat{x}$, in Equation (1) is the same. The optimal solution is the first evaluation point x_n that deviates from the target value *V* by no more than the limit *L*. That is:

$$\hat{x} = \min\left\{x_n : g(x_n) < L\right\}, n = 1, 2, \ldots$$

We continue with the *CB* acquisition function only, since a simple modification of it yields excellent results. The *CB* acquisition function resembles a confidence bound (Equation 7). In the case of a maximization, the minus sign is replaced by plus. The hyper-parameter κ can be used to tune the optimization.

$$x_{n+1} = \min_{i=1..M}\left(\bar{\boldsymbol{m}}[i] - \kappa\bar{\boldsymbol{s}}[i]\right) \tag{7}$$

A small value of κ stresses the $\bar{\boldsymbol{m}}$ parameter, and rewards 'exploitation'. The focus is on the calculated values of $\bar{\boldsymbol{m}}$. A larger value of κ stresses 'exploration': choosing more distant values in the hope that an improvement results.

E. Adjustment for Noisy Evaluation

Elements of noise, usually modelled by the error term $\varepsilon \sim N\left(0, \omega^2\right)$ in Equation 1, imply a replacement of matrix $\boldsymbol{K}$ in Equations (4-5) by $\boldsymbol{K} + \omega^2\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix with the same dimensions as $\boldsymbol{K}$. Several factors complicate this simple replacement. In the case we consider, it is possible that the noise component should include a non-zero drift term, ν, so that $\varepsilon \sim N\left(\nu, \omega^2\right)$. This would imply that samples are correlated. Boyle [19] makes the point that noise can dominate a signal so much that the length of path to convergence can be an order of magnitude greater than a change in the noise rate. Overall, it is less safe to assume that if two points x_i and x_j are close, then $g(x_i)$ is close to $g(x_j)$. The *LDA* Monte Carlo process that we use is very susceptible to noise, as it the data. It is also possible that there is a *GP* consistency problem over time, which could lead to unreliable predictions. Therefore we do not expect a smooth path to convergence. It seems reasonable that more Monte Carlo cycles should stabilize the calculation, but a suitable value for κ in Equation (7) is unclear.

IV. Improved Optimization

The results in Section V illustrate the non-satisfactory nature of the *CB*, *POI* and *EI* acquisition functions. It is likely that they fail because the optimisation rule in equation (1) contains the additional requirement that the minimum deviation from zero must be within a pre-determined limit *L*. The following amendment to the *CB* acquisition function makes a dramatic difference. We call it the *ZERO* acquisition function since the optimal solution should result in an error (relative to the target) of approximately zero.

$$x_{n+1} = \min_{i=1..M}\left(\bar{\boldsymbol{m}}[i] - \kappa\bar{\boldsymbol{s}}[i]\right)^2 \tag{8}$$

The decision of which acquisition to use is very problem dependent, and two elements of the optimisation problem in Equation (1) are particularly pertinent. First, evaluation of the function $f((1+x)D)$ has a random component. Second, the optimisation criterion has an accuracy target. The intuition behind the proposal in Equation (8) is that a quantity has to be as close as possible to a target, and a simple way to measure closeness is "deviation-squared" (absolute deviation works

just as well). Ultimately the justification for using Equation (8) is empirical, and results are given in Section V.

F. The 'Zero Point' amendment

A further amendment improves the accuracy of the next evaluation point estimate. In order to 'pull' the estimate nearer to the 'zero point' $g(x) = L$ (Equation 1a), a current 'zero point' x' is calculated from the gradient, θ, and intercept, c, of the least squares best fit line to the known set $\{\boldsymbol{x}, \boldsymbol{y}\}$. The evaluation point estimate returned is the mean of the estimate from Equation (8) and x'. In cases where x' is outside the interval I, calculating x' in the same way would produce another value outside I. In those cases x' is set to a small value γ marginally inside I. The details are in Equations (9a and 9b). The amended value of x_{n+1} is denoted by x'_{n+1}.

$$\begin{aligned} y_i &= \theta x_i + c \qquad x_i \in I, i = 1..n \\ x' &= \frac{-\theta}{c} \\ x' &= \max(I) - \gamma \quad x_i > \max(I), i = 1..n \\ x' &= \max(I) + \gamma \quad x_i < \max(I), i = 1..n \end{aligned} \tag{9a}$$

$$x'_{n+1} = \frac{(x' + x_{n+1})}{2} \tag{9b}$$

The results in Section V show that using the evaluation point x'_{n+1} of Equation (9b) is a considerable improvement on using x_{n+1} alone.

G. Improved Optimization: Analysis

The distinction between the acquisition functions in Equations (7) and (8) is small, but the difference in performance is significant. The reasons are due to the presence of a target in the optimization function (Equation 1) and to the "sticking" phenomenon, where the *GP* persistently proposes a non-optimal next evaluation point. They are inter-related, and the improvement using the acquisition function in Equation (8) can be rationalized as follows. The underlying concept of a *GP* is that a small change in inputs should result in a small change in outputs. They are respectively the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ defined in Section III(B). In other words, the *GP* should define a well-conditioned system. Formally, as in [20], a Lipschitz-continuous condition is assumed. Thus, in Equation 10, below, C is some (typically unknown) constant and g is more generally any continuous function).

$$\left|g(x_i) - g(x_j)\right| < C\left|x_i - x_j\right| \quad \forall \{x_i, x_j\} \in \boldsymbol{x} \tag{10}$$

Consider the case where the current evaluation point x_n results in an optimal value which is close to the target but has not attained the target. At the next evaluation, if the evaluation point x_{n+1} is close to x_n, the Lipschitz-continuous condition implies that there is a good chance that the target will be achieved, since $g(x_{n+1})$ is expected to be close to $g(x_n)$. Similarly, consider the case where $g(x_n)$ is a long way from the target. In that case, $g(x_{n+1})$ is also expected to be a long way from the target. That is the origin of the "sticking" problem. If subsequent evaluation points remain a long way from the target, they will continue to miss the target so that the probability that the target is attained is small. Additionally, the acquisition functions in Equations (7) and (8) are designed for different purposes. The *LCB* acquisition function is appropriate for finding an absolute function minimum, whereas The *ZERO* acquisition function is designed to minimize a non-negative function subject to a condition.

The following is a formalization of the above argument in the case where a current error estimate $\left|g(x_n) - g(\hat{x})\right|$ is a 'near miss', meaning that it is close to but slightly greater than the target value L. That is:

$$L < \left|g(x_n) - g(\hat{x})\right| < L + \varepsilon_1, \tag{11a}$$

where ε_1 is small.

Also assume that the *GP* produces a next evaluation point that is 'close to' the current evaluation point.

$$\left|x_{n+1} - x_n\right| < \varepsilon_2 \tag{11b}$$

First consider an upper bound for $\left|g(x_{n+1}) - g(\hat{x})\right|$. The second line of Equations (11c), below, uses the Lipschitz condition of Equation 10).

$$\begin{aligned} &\left|g(x_{n+1}) - g(\hat{x})\right| < \left|g(x_{n+1}) - g(x_n)\right| + \left|g(x_n) - g(\hat{x})\right| \\ &< C\left|x_{n+1} - x_n\right| + L + \varepsilon_1 \\ &< C\varepsilon_2 + L + \varepsilon_1 = L + \varepsilon \end{aligned} \tag{11c}$$

The last term in Equation (11c), with $\varepsilon = C\varepsilon_2 + \varepsilon_1$ indicates a 'near miss' for $\left|g(x_{n+1}) - g(\hat{x})\right|$.

Now consider the case $\left|g(x_{n+1}) - g(\hat{x})\right| < L$. The Lipschitz condition for x_{n+1} and $\hat{x}$ gives $\left|g(x_{n+1}) - g(\hat{x})\right| < C\left|x_{n+1} - \hat{x}\right|$.

Therefore, in order to also satisfy the inequality in Equation (11c),

$$C\left|x_{n+1} - \hat{x}\right| < L + \varepsilon \Rightarrow \left|x_{n+1} - \hat{x}\right| < \frac{L + \varepsilon}{C} \tag{12a}$$

Then, to ensure that $\left|g(x_{n+1}) - g(\hat{x})\right| < L$,

$$\frac{L + \varepsilon}{C} < L \Rightarrow C > 1 + \frac{\varepsilon}{L} \tag{12b}$$

Effectively, Equation (12b) says that C need only be marginally larger than 1.

V. Results

A. Data

The data set D of Equation (1) comprises aggregated daily operational losses collected from January 2010 to December 2018. That period has a total of approximately 2500 such losses, ranging from \$1 to approximately \$60m. They have a fat-tailed distribution, and only 7.5% exceed \$1m. All calculations were done using *R* on an i7 Windows processor with 16GB RAM.

B. Results: EI, POI, LCB and Random

The results in the tables in this section show the mean and standard deviation of the number evaluations of the

'expensive' function g of Equation 1a in order for the required optimization error to be within a limit $L = 0.01$ (i.e. within 1% of the target). Table I shows the results for random selection of the parameter x in Equation (1), with the corresponding values using *BO* with the *EI, POI* and *LCB* acquisition functions, discussed in Section III(C). In those cases, 3 'expensive' evaluations of g were done to initialize the *GP*. In general, the parameter κ in the *LCB* acquisition function (Equation 7) was not significant in determining the number of 'expensive' evaluations of g, and was set to 0.5. The number of Monte Carlo iterations used in evaluating g was more important, and Table I shows the results for the maximum (5 million) and minimum (1 million) Monte Carlo iterations considered. Normally for this type of calculation we would opt for the minimum number (and often fewer) to save time.

TABLE I: MEAN AND SD OF FUNCTION EVALUATIONS FOR RANDOM SEARCH, AND THE EI, POI AND LCB ACQUISITION FUNCTIONS

Acquisition function	5m Mean	1m Mean	5m SD	1m SD
EI	11.45	18.08	10.43	12.15
LCB(1)	9.85	17.56	7.72	15.70
POI	14.60	22.92	14.41	15.87
Random	10.84	12.42	10.88	12.56

The most notable observation from Table I is the poor performance of the common acquisition functions. Two gave worse results (i.e. greater mean and/or standard deviation) than a random search. These results provide the motivation for the amended (*ZERO*) acquisition function.

C. Results: ZERO Acquisition

The tables in this section show the corresponding results for the *ZERO* acquisition function with and without using the '*Zero Point*' amendment (Section IV A). The results are analyzed by the number of Monte Carlo iterations used for evaluation of f, and the value of κ (Equation 8).

D. Zero Acquisition without 'Zero Point'

The values obtained for the mean (μ) and standard deviation (σ) of the number of 'expensive' evaluations of g are shown separately (Tables II and III respectively).

TABLE II: MEAN OF FUNCTION EVALUATIONS FOR ZERO ACQUISITION, **WITHOUT** THE 'ZERO POINT' AMENDMENT, 25 TRIALS

	Monte Carlo iterations (millions)				
κ	1m	2m	3m	4m	5m
0	7.48	5.28	5.32	4.72	5.64
0.25	8.64	5.68	4.36	5.16	4.48
0.50	6.60	5.12	4.44	4.88	5.00
0.75	6.80	5.52	6.12	4.48	4.24
1.00	7.28	6.32	4.36	3.60	5.60
1.25	6.72	5.00	4.68	5.20	5.56
1.50	7.52	5.12	5.20	5.08	5.84
1.75	7.56	8.36	6.60	5.44	6.48
2.00	6.68	6.76	7.80	6.32	6.12

TABLE III: SD OF FUNCTION EVALUATIONS FOR ZERO ACQUISITION, **WITHOUT** THE 'ZERO POINT' AMENDMENT

	Monte Carlo iterations (millions)				
κ	1m	2m	3m	4m	5m
0	5.03	2.05	3.52	3.02	3.24
0.25	5.74	3.93	3.03	2.98	2.54
0.50	4.70	4.34	1.53	2.45	2.77
0.75	6.61	3.99	2.99	2.82	2.40
1.00	6.62	5.11	3.08	1.76	3.85
1.25	3.71	4.65	3.42	2.99	2.92
1.50	5.78	3.18	3.97	3.75	3.98
1.75	5.42	6.67	4.56	2.31	5.04
2.00	5.17	6.04	5.61	4.63	3.98

It is clear that the results for 1 and 5 million Monte Carlo iterations in Tables II and III are a considerable improvement on those in Table I. In all cases the means and standard deviations are reduced using *ZERO* acquisition. The real gain is that long runs in which the sequence of 'expensive' fits fails to attain the required target are relatively uncommon (as evidenced by the lower standard deviations). Tables II and III reveal two broad trends. First, accuracy (i.e. a minimal number of runs to attain the target) generally improves with an increasing number of Monte Carlo iterations. Second, accuracy is better for a lower value of κ, indicating that exploration (i.e. searching away from the *GP*-suggested mean) is an inferior policy.

E. Zero Acquisition with 'Zero Point'

Tables IV and V show the results (mean μ and standard deviation σ) of applying the '*Zero Point*' amendment under the same conditions that were used for Tables II and III. In both Tables IV and V, the entries marked in bold text are the ones that are greater than the corresponding entries in Tables II and III. They represent cases where the '*Zero Point*' amendment failed to reduce the number of 'expensive' evaluations of function g (Equation 2). All other entries in Tables IV and V are less than the corresponding entries in Tables II and III. They represent cases where the '*Zero Point*' amendment successfully reduced the number of 'expensive' evaluations of function g.

TABLE IV: MEAN OF FUNCTION EVALUATIONS FOR ZERO ACQUISITION, **WITH** THE 'ZERO POINT' AMENDMENT, 25 TRIALS

	Monte Carlo iterations (millions)				
κ	1m	2m	3m	4m	5m
0	5.58	**5.60**	5.28	4.68	5.20
0.25	6.09	5.56	4.72	5.32	4.36
0.50	5.43	**6.60**	**4.28**	4.84	4.32
0.75	6.24	4.72	4.80	4.20	**4.80**
1.00	5.96	6.02	**5.88**	**5.00**	4.72
1.25	4.96	5.68	3.94	4.18	4.86
1.50	6.12	4.70	5.68	5.92	4.84
1.75	5.44	4.68	6.12	4.80	4.84
2.00	6.19	5.52	4.96	4.88	5.08

TABLE V: SD OF FUNCTION EVALUATIONS FOR ZERO ACQUISITION, **WITH** THE 'ZERO POINT' AMENDMENT

	Monte Carlo iterations (millions)				
κ	1m	2m	3m	4m	5m
0	3.89	**2.90**	2.69	2.81	2.60
0.25	3.95	3.33	2.21	2.29	1.80
0.50	3.24	3.03	**2.07**	2.27	2.41
0.75	3.33	2.62	2.48	1.55	**2.84**
1.00	3.32	3.03	3.00	**2.97**	1.79
1.25	3.06	2.90	2.25	2.58	2.24
1.50	2.22	2.78	2.61	2.58	3.36
1.75	2.01	2.75	3.22	**2.64**	3.17
2.00	3.20	2.71	2.72	3.81	3.87

The results in Tables IV and V show that the 'Zero Point'

amendment has the effect of reducing the mean and standard deviation of the number of runs needed to attain the target by about 25%.

F. Results Using 0.5 Million Monte Carlo Cycles

Table VI shows the results of applying *ZERO* acquisition, with and without the 'Zero Point' amendment, using only 0.5 million Monte Carlo cycles in function g (Equation 2). The results shows that 0.5 million cycles are too few to be acceptable if the 'Zero Point' amendment is not used. Even if the mean number of 'expensive' evaluations of g is acceptable, the corresponding standard deviation is not. Without the 'Zero Point' amendment, 32% of optimization runs were longer than 10 (expensive function evaluations), 6% were longer than 20 and 2% were longer than 30. In contrast, if 'Zero Point' is applied, the corresponding figures are 15%, 0% and 0%. At best, the '0.5m' result with the 'Zero Point' amendment is approximately equivalent to the '5m' result without.

TABLE VI: MEAN AND SD, ZERO ACQUISITION, 0.5M MONTE CARLO WITH ($\boldsymbol{W}$) AND WITHOUT ($\bar{\boldsymbol{W}}$) 'ZERO POINT', 25 TRIALS OF EACH

κ	Mean $\boldsymbol{W}$	SD $\boldsymbol{W}$	Mean $\bar{\boldsymbol{W}}$	SD $\bar{\boldsymbol{W}}$
0	5.40	3.97	9.64	6.21
0.25	6.88	4.24	7.48	6.36
0.50	5.48	4.43	7.84	7.88
0.75	6.84	4.07	9.08	8.24
1.00	5.32	3.54	6.92	5.30
1.25	7.40	4.01	9.60	7.98
1.50	5.28	3.27	8.84	7.70
1.75	7.52	4.26	9.76	4.41
2.00	5.32	3.49	10.04	8.24

If the zero point amendment is used, the results are borderline. Together, the two cases establish a minimum number of Monte Carlo cycles that should be used: 1 million.

G. Confidence Bound illustration

To illustrate the effect of the 'Zero Point' amendment, Fig. 1 shows surfaces plots of upper confidence bounds derived from Tables II-V. The upper surface is the combination $\mu + \sigma$ from Tables II and III (no 'Zero Point' amendment). The lower surface is the combination $\mu + \sigma$ from Tables IV and V (the 'Zero Point' amendment is included).

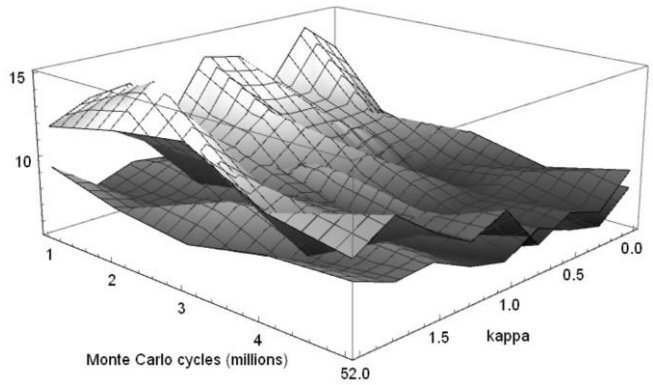

Fig. 1. Upper confidence surfaces. Upper surface ($\mu+\sigma$): *ZERO* acquisition confidence without the 'Zero Point' amendment. Lower surface ($\mu+\sigma$): *ZERO* acquisition with the 'Zero Point' amendment. Vertical axis is the number of 'expensive' evaluations of g.

Both surfaces show the same characteristics. Optimal results correspond to a large number of Monte Carlo iterations, and there is no clear dependence on κ. The surfaces intersect on a small subset of combinations of κ and Monte Carlo iterations, mostly coincident with the indications marked in Tables 4 and 5. Overall, the 'Zero Point' amendment is successful in reducing the number of 'expensive' evaluations of g.

H. Discussion of Results

The difference between using a small and a large number of Monte Carlo iteration is illustrated in the figures below. Both show the objective to be optimized: components of the vectors $\bar{\boldsymbol{m}}[i]$ and $\bar{\boldsymbol{s}}[i]$ from Equation (7) with κ = 0.5. The black traces show the objective function $\left(\bar{\boldsymbol{m}}[i]-\kappa\bar{\boldsymbol{s}}[i]\right)^2$, the dark gray traces show the squared means $\left(\bar{\boldsymbol{m}}[i]\right)^2$ and the light gray traces show the variances $\left(\bar{\boldsymbol{s}}[i]\right)^2$. In each case, the index i = 1..100 represents the sample for the Bayes predictor of Equations (5). Figs. (2) and (3) show those traces for 0.25m and 4m Monte Carlo iterations respectively. The former shows that the standard deviation component in the objective can be significant compared to the mean component. In the latter case it is not.

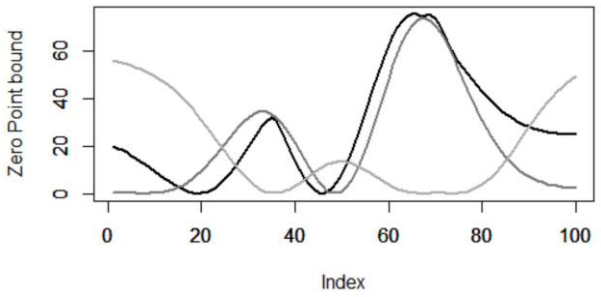

Fig. 2. Predictor distributions, 0.25m Monte Carlo iterations (optimal value at index 46; black – objective; dark gray – squared means; light gray – variances).

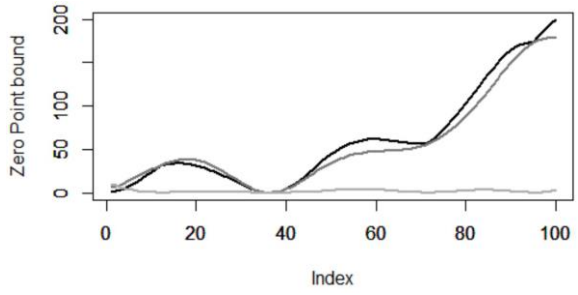

Fig. 3. Predictor distributions, 4m Monte Carlo iterations (optimal value at index 36; black – objective; dark gray – squared means; light gray – variances).

VI. CONCLUSION

The most surprising result from this analysis is that the 'traditional' acquisition functions (*EI*, *POI* and *CB*) performed very poorly in the context described. The performance measure (the number of runs required to attain the target) is pertinent because a large number of runs is undesirable given the lengthy time required for each one. Indeed, better performance is achieved using random selection. It is equally surprising is that the solution is very simple: use the square of the lower confidence bound instead

of the lower confidence bound itself. Despite improvements that result in an acceptable mean number of runs, the standard deviations often show that the possibility of recording long runs persists.

A particular finding is that it is not productive to optimize κ. Although particular values of κ are more beneficial than others, no consistent pattern is discernable. It seems that the stochastic process g renders a *GP* proposal as an approximation rather than an accurate value that should be consistent with the required target.

Extensions of our work on *BO* are already well advanced. We are formulating a more rigorous proof that the *ZERO* acquisition function is superior to the *LCB* acquisition function for the type of optimization problem under consideration. In this context, 'superior' means that the expected number of 'expensive' function evaluations needed to attain the target is fewer for *ZERO* than for *LCB*. This work uses the concept of 'regret' – the difference between a proposed function estimate $g(x_{n+1})$ and $g(\hat{x})$.

More generally, the use of *BO* and *GP*s have hitherto not been used in the context of financial risk. The context described in this paper is simple in the sense that only one parameter needs to be optimized. We hope to provide *BO* solutions to more complicated financial risk problems in the future.

Conflict of Interest

The author declares no conflict of interest.

Author Contributions

All contributions were due to the author, PM.

References

[1] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Jnl. Machine Learning Research*, vol. 13, pp. 281-305, 2012.

[2] J. Mockus. (1974). On bayesian methods for seeking the extremum. [Online]. Available: http://dl.acm.org/citation.cfm?id=646296.687872

[3] A. Frachot, P. Georges, and T. Roncalli. (2001). Loss distribution approach for operational risk. [Online]. Available: http://ssrn.com/abstract=1032523

[4] S. Rana, C. Li, and S. Gupta, "High dimensional bayesian optimization with Elastic gaussian process," in *Proc. 34th Int. Conf. on Machine Learning, Sydney*, 2017.

[5] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, MIT Press, 2006.

[6] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, Chapter 15, MIT Press, 2012.

[7] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of Bayesian methods for seeking the extremum," *Towards Global Optimisation*, vol. 2, 1978.

[8] J. Mockus, "The bayesian approach to local optimization," *Bayesian Approach to Global Optimization. Mathematics and Its Applications*, vol. 37, 1989.

[9] D. D. Cox and S. John, "SDO: A statistical method for global optimization," *Multidisciplinary Design Optimization*, pp. 315-329, SIAM, Philadelphia, 1997.

[10] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104322.3104451

[11] J. Snoek, H. Larochelle, and R. P. Adams, "Practical optimization of machine learning algorithms," *Advances in Neural Information Processing Systems*, pp. 2951-2959, 2012.

[12] H. J. Kushner, "A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise," *J. Basic Eng*, vol. 86, no. 1, pp. 97-106, 1964.

[13] S. Sano, T. Kadowaki, K. Tsuda, *et al*. (2019). Application of bayesian optimization for pharmaceutical product development. [Online]. Available: https://doi.org/10.1007/s12247-019-09382-8

[14] P. Ilten, M. Williams, and Y. Yang, "Event generator tuning using Bayesian optimization," *Preprint JINST 12*, 2017.

[15] E. Mehdad and J. P. Kleijnen, "Efficient global optimisation for black-box simulation via sequential intrinsic kriging," *Jnl. Operational Research Society*, vol. 69, pp. 1-13, 2018.

[16] S. Roberts, M. Osborne, M. Ebden, *et al*., "Gaussian processes for time-series modelling," *Phil Trans R Soc A* 371, 2013.

[17] P. Ranjan, R. Haynes, and R. Karsten, "A computationally stable approach to gaussian process interpolation of deterministic computer simulation data," *Technometrics*, vol. 53, no. 4, pp. 366–378, 2011.

[18] J. Berk, V. Nguyen, S. Gupta, *et al*., "Exploration enhanced expected improvement for bayesian optimization," *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*: *LNCS* 11052, pp. 621-637, 2018.

[19] P. Boyle, "Gaussian processes for regression and optimisation," Ph.D, University of Wellington, 2006.

[20] E. Brochu, V. M. Cora, and N. D. Freitas, "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," 2019.

Peter Mitic is a honorary professor in the Department of Computer Science, University College London, and is the head of Operational Risk at Santander Bank UK. He has published extensively on operational and reputational risk, and also on simulation and machine learning.