GLASSES: Relieving The Myopia Of Bayesian Optimisation

Anonymous authors

Abstract

We present GLASSES: Global optimisation with Look-Ahead through Stochastic Simulation and Expected-loss Search. The majority of global optimisation approaches in use are myopic, in only considering the impact of the next function value; the non-myopic approaches that do exist are able to consider only a handful of future evaluations. Our novel algorithm, GLASSES, permits the consideration of dozens of evaluations into the future. We show that the far-horizon planning thus enabled leads to substantive performance gains in empirical tests.

1 Introduction

Global optimisation is core to any complex problem where design and choice play a role. Within Machine Learning, such problems are found in the tuning of hyperparameters [16], sensor selection [2] or experimental design [4, 10]. Most global optimisation techniques are myopic, in considering no more than a single step into the future. Relieving this myopia requires solving the *multi-step lookahead* problem: the global optimisation of an function by considering the significance of the next function evaluation on function evaluations (steps) further into the future. It is clear that a solution to the problem would offer performance gains. For example, consider the case in which we have a budget of two evaluations with which to optimise a function $f(x)$ over the domain $\mathcal{X} = [0, 1] \subset \mathbb{R}$. If we are strictly myopic, our first evaluation will likely be at $x = 1/2$, and our second then at only one of $x = 1/4$ and $x = 3/4$. This myopic strategy will thereby result in ignoring half of the domain $\mathcal{X}$, regardless of the second choice. If we adopt a two-step lookahead approach, we will select function evaluations that will be more evenly distributed across the domain by the time the budget is exhausted. We will consequently be better informed about f and its optimum.

There is a limited literature on the multi-step lookahead problem. [14] perform multi-step lookahead by optimising future evaluation locations, and sampling over future function values. This approach scales poorly with the number of future evaluations considered, and the authors present results for no more than two-step lookahead. [9] reframe the multi-step lookahead problem as a partially observed Markov decision process, and adopt a Monte Carlo tree search approach in solving it. Again, the scaling of the approach permits the authors to consider no more than six steps into the future. In the past, the multi-step look ahead problem was studied by [17] proposing a utility function that maximizes the total 'reward' of the algorithm by taking into account the cost of future computations, rather than trying to find the optimum after a fixed number of evaluations.

2 Background and challenge

Let $f : \mathcal{X} \to \Re$ be well behaved function defined on a compact subset $\mathcal{X} \subseteq \Re^q$. Our goal is to find $\mathbf{x}_M = \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. We assume that f is a *black-box* from which only perturbed evaluations of the type $y_i = f(\mathbf{x}_i) + \epsilon_i$, with $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, are available. Bayesian Optimization (BO) is an heuristic strategy to make a series of evaluations $\mathbf{x}_1, \ldots, \mathbf{x}_n$ of f, typically very limited in number, such that the the minimum of f is evaluated as soon as possible [8, 7, 16, 1]. Assume that N points have been gathered so far, having a dataset $\mathcal{D}_0 = \{(\mathbf{x}_i, y_i)\}_{i=1}^N = (\mathbf{X}_0, \mathbf{y}_0)$. Before collecting any new point, a surrogate probabilistic model for f is calculated. This is typically a Gaussian Process

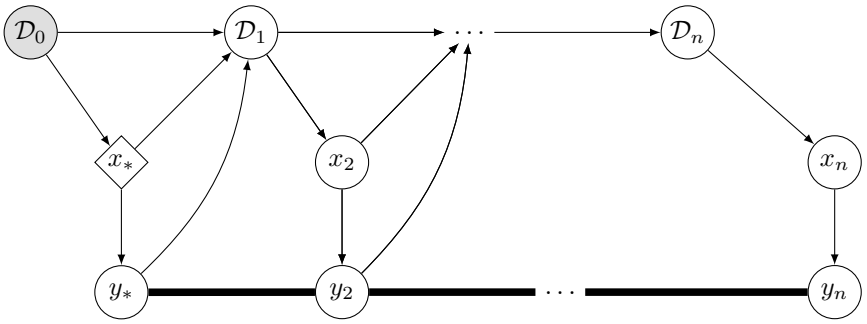

Figure 1: A Bayesian network describing the n-step lookahead problem. The shaded node ($\mathcal{D}_0$) is known, and the diamond node (x_*) is the current decision variable. All y nodes are correlated with one another under the GP model. Note that the nested maximisation problems required for x_i and integration problems required for y_* and y_i (in either case for $i = 2, \dots, n$) render inference in this model prohibitively computationally expensive.

(GP) $p(f) = \mathcal{GP}(\mu; k)$ with mean function μ and a covariance function k, whose parameters will be denoted by $\boldsymbol{\theta}$. Let $\mathcal{I}_0$ be the current available information: the conjunction of $\mathcal{D}_0$, the model parameters and the model likelihood type. Given the GP model, we need to determine the best location to sample. Denote by $\eta = \min\{\mathbf{y}_0\}$, the current best found value. If only one remaining evaluation is available ($n = 1$) we can define the loss of evaluating f this last time at $\mathbf{x}_*$ assuming it is returning y_* as

$$\lambda(y_*) \triangleq \begin{cases} y_*; & \text{if} \quad y_* \leq \eta \\ \eta; & \text{if} \quad y_* > \eta. \end{cases}$$

Its expectation is $\Lambda_1(\mathbf{x}_*|\mathcal{I}_0) \triangleq \mathbb{E}[\min(y_*, \eta)] = \int \lambda(y_*) p(y_*|\mathbf{x}_*, \mathcal{I}_0) \mathrm{d}y_*$ where the subscript in Λ refers that we are considering one future evaluation. The next evaluation is located where $\Lambda_1(\mathbf{x}_*|\mathcal{I}_0)$ gives the minimum value. This point can be obtained by any gradient descent algorithm since analytical expressions for the loss and its gradients are explicitly available [13].

$\Lambda_1(\mathbf{x}_*|\mathcal{I}_0) \triangleq \mathbb{E}[\min(y_*, \eta)] = \int \lambda(y_*) p(y_*|\mathbf{x}_*, \mathcal{I}_0) \mathrm{d}y_*$ can be used as a myopic approximation to the optimal decision when n evaluations of f remain available. Indeed, most BO methods are myopic and ignore the future decisions that will be made in the future steps. Let $\{(\mathbf{x}_j, y_j)\}$ for $j = 1, \dots, n$ be the remaining n available evaluations and by $\mathcal{I}_j$ the available information after the data set $\mathcal{D}_0$ has been augmented with $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_j, y_j)$ and the parameters $\boldsymbol{\theta}$ of the model updated. We use $\Lambda_n(\mathbf{x}_*|\mathcal{I}_0)$ to denote the expected loss of selecting $\mathbf{x}_*$ given $\mathcal{I}_0$ and considering n future evaluations. A proper Bayesian formulation allows us to define this *long-sight* loss [13] as[1]

$$\Lambda_n(\mathbf{x}_*|\mathcal{I}_0) = \int \lambda(y_n) \prod_{j=1}^{n} p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) p(\mathbf{x}_j|\mathcal{I}_{j-1}) \mathrm{d}y_* \dots \mathrm{d}y_n \mathrm{d}\mathbf{x}_2 \dots \mathrm{d}\mathbf{x}_n \tag{1}$$

where $p(y_j|\mathbf{x}_j, \mathcal{I}_{j-1}) = \mathcal{N}\left(y_j; \mu(\mathbf{x}_j; \mathcal{I}_{j-1}), \sigma^2(\mathbf{x}_j|\mathcal{I}_{j-1})\right)$ is the predictive distribution of the GP at $\mathbf{x}_j$ and $p(\mathbf{x}_j|\mathcal{I}_{j-1}) = \delta\big(\mathbf{x}_j - \arg\min_{\mathbf{x}_* \in \mathcal{X}} \Lambda_{n-j+1}(\mathbf{x}_*|\mathcal{I}_{j-1})\big)$ reflects the optimization step required to obtain $\mathbf{x}_j$ after all previous the evaluations f have been iteratively optimized and marginalized. The graphical probabilistic model underlying (1) is illustrated in Figure 1. To evaluate Eq. (1) we can successively sample from y_1 to y_{j-1} and optimize for the appropriate $\Lambda_{n-j+1}(\mathbf{x}_*|\mathcal{I}_{j-1})$. This is in done in [13] for only two steps look ahead. The reason why further horizons remain unexplored is the computational burden required to compute this loss for many steps ahead. Note that analytical expression are only available in the myopic case $\Lambda_1(\mathbf{x}_*|\mathcal{I}_0)$.

3 The GLASSES algorithm

A proper multi-step look ahead loss function requires the iterative optimization-marginalization of the future steps, which is computationally intractable. A possible way of dealing with this issue is to jointly model our epistemic uncertainty over the future locations $\mathbf{x}_2, \dots, \mathbf{x}_n$ with a

[1] We assume that $p(\mathbf{x}_*|\mathcal{I}_0) = 1$.

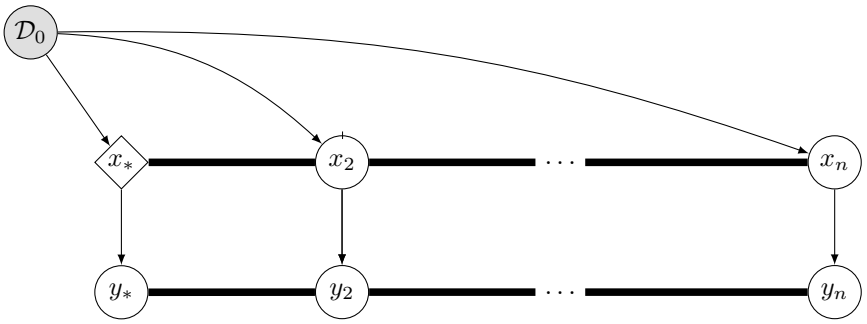

Figure 2: A Bayesian network describing our approximation to the n-step lookahead problem. The shaded node ($\mathcal{D}_0$) is known, and the diamond node (x_*) is the current decision variable, which is now directly connected with all future steps of the algorithm. Compare with Figure 1: the sparser structure renders our approximation computationally tractable.

Algorithm 1 Decision process of the GLASSES algorithm.

Input: dataset $\mathcal{D}_0 = \{(\mathbf{x}_0, y_0)\}$, number of remaining evaluations (n), look-ahead predictor $\mathcal{F}$.
for $j = 0$ **to** n **do**
1. Fit a GP with kernel k to $\mathcal{D}_j$.
3. Select next location $\mathbf{x}_j$ by taking $\mathbf{x}_j = \arg\min_{\mathbf{x}\in\mathcal{X}} \Lambda_{n-j}(\mathbf{x}_*|\mathcal{I}_0, \mathcal{F}_{n-j}(\mathbf{x}_*))$.
4. Evaluate f at $\mathbf{x}_j$ and obtain y_j.
5. Augment the dataset $\mathcal{D}_{j+1} = \{\mathcal{D}_j \cup (\mathbf{x}_j, y_j)\}$.

end for
Fit a GP with kernel k to $\mathcal{D}_n$
Returns: Propose final location at $\arg\min_{\mathbf{x}\in\mathcal{X}} \{\mu(\mathbf{x}; \mathcal{D}_n)\}$.

joint probability distribution $p(\mathbf{x}_2, \ldots, \mathbf{x}_n|\mathcal{I}_0, \mathbf{x}_*)$ and to consider the expected loss $\Gamma_n(\mathbf{x}_*|\mathcal{I}_0) = \int \lambda(y_n)p(\mathbf{y}|\mathbf{X}, \mathcal{I}_0, \mathbf{x}_*)p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)\mathrm{d}\mathbf{y}\mathrm{d}\mathbf{X}$ for $\mathbf{y} = \{y_*, \ldots, \ldots, y_n\}$ the vector of future evaluations of f and $\mathbf{X}$ the $(n-1) \times q$ dimensional matrix whose rows are the future evaluations $\mathbf{x}_2, \ldots, \mathbf{x}_n$. $p(\mathbf{y}|\mathbf{X}, \mathcal{I}_0, \mathbf{x}_*)$ is multivariate Gaussian, since it corresponds to the predictive distribution of the GP at $\mathbf{X}$. The graphical probabilistic model underlying (1) is illustrated in Figure 2. Now, all future evaluations are modelled jointly rather then sequentially. A proper choice of $p(\mathbf{X}|\mathcal{I}_0, \mathbf{x}_*)$ is crucial here, which may be difficult to define and obtain. Instead is this work with a fixed $\mathbf{X}$, which we assume it is computed beforehand. Although this approach ifnores our epistemic uncertainty on the future steps, it drastically reduces the computational burden of approximating $\Lambda_n(\mathbf{x}_*|\mathcal{I}_0)$.

Suppose that we had access to an oracle function $\mathcal{F}_n : \mathcal{X} \to \mathcal{X} \times \overset{n}{\cdots} \times \mathcal{X}$ able to predict the n future locations that the loss $\Lambda_n(\cdot)$ would suggest if we started evaluating f at $\mathbf{x}_*$. In other words, $\mathcal{F}_n$ takes the putative location $\mathbf{x}_*$ as input and it returns $\mathbf{x}_*$ and the predicted future locations $\mathbf{x}_2, \ldots, \mathbf{x}_n$. This is an unrealistic assumption in practice, but it will help us to set-up our algorithm. We leave for the next section the details of how to marginalise over $\mathcal{F}_n$. Assume, for now, that $\mathcal{F}_n$ exists and that we have access to it. We it and denote by $\mathbf{y} = (y_*, \ldots, \ldots, y_n)^T$ the vector of future locations evaluations of f at $\mathcal{F}_n(\mathbf{x}_*)$. Assuming that $\mathbf{y}$ is known, it is possible to rewrite the expected loss in Eq. (1) as $\Lambda_n\big(\mathbf{x}_* \mid \mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)\big) = \mathbb{E}\big[\min(\mathbf{y}, \eta)\big]$ where the expectation is taken over the multivariate Gaussian distribution that gives rise after marginalizing the posterior distribution of the GP at $\mathcal{F}_n(\mathbf{x}_*)$. The intuition behind $\Lambda_n\big(\mathbf{x}_* \mid \mathcal{I}_0, \mathcal{F}_n(\mathbf{x}_*)\big)$ is as follows: the expected loss at $\mathbf{x}_*$ is the best possible function value that we expect to find in the next n steps, conditional on the first evaluation being made at $\mathbf{x}_*$. Rather than merely quantifying the benefit provided by the next evaluation, this loss function accounts for the expected gain in the whole future sequence of evaluations of f. As we analyse in the experimental section of this work, the effect of this is an adaptive loss that tends to be more explorative when more remaining evaluations are available and more exploitative as soon as we approach the final evaluations of f.

	MPI	GP-LCB	EL	EL-2	EL-3	EL-5	EL-10	GLASSES
SinCos	0.7147	0.6058	0.7645	*0.8656*	0.6027	0.4881	*0.8274*	***0.9000***
Cosines	0.8637	0.8704	0.8161	*0.8423*	*0.8118*	0.7946	0.7477	***0.8722***
Branin	0.9854	0.9616	**0.9900**	0.9856	0.9673	0.9824	0.9887	0.9811
Sixhumpcamel	0.8983	**0.9346**	0.9299	0.9115	0.9067	0.8970	0.9123	0.8880
Mccormick	**0.9514**	0.9326	0.9055	*0.9139*	*0.9189*	*0.9283*	*0.9389*	*0.9424*
Dropwave	0.7308	0.7413	0.7667	0.7237	0.7555	0.7293	0.6860	***0.7740***
Powers	0.2177	0.2167	0.2216	*0.2428*	*0.2372*	*0.2390*	*0.2339*	***0.3670***
Ackley-2	0.8230	**0.8975**	0.7333	0.6382	0.5864	0.6864	0.6293	0.7001
Ackley-5	0.1832	0.2082	0.5473	*0.6694*	0.3582	0.3744	***0.6700***	0.4348
Ackley-10	0.9893	0.9864	0.8178	*0.9900*	*0.9912*	***0.9916***	*0.8340*	*0.8567*
Alpine2-2	**0.8628**	0.8482	0.7902	0.7467	0.5988	0.6699	0.6393	0.7807
Alpine2-5	0.5221	0.6151	**0.7797**	0.6740	0.6431	0.6592	0.6747	0.7123

Table 1: Average results for the average 'gap' measure. EL-k is the expect loss function computed with k steps ahead at each iteration. The best result for each function is bolded. In italic, the cases in which a non-myopic loss outperforms the myopic loss are highlighted.

To compute $\mathbb{E}[\min(\mathbf{y}, \eta)]$ we use Expectation Propagation, EP, [11]. This turns out to be a natural operation by observing that

$$\mathbb{E}[\min(\mathbf{y}, \eta)] = \eta \int_{\mathbb{R}^n} \prod_{i=1}^{n} h_i(\mathbf{y}) \mathcal{N}(\mathbf{y}; \mu, \Sigma) \mathrm{d}\mathbf{y} + \sum_{j=1}^{n} \int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(\mathbf{y}) \mathcal{N}(\mathbf{y}; \mu, \Sigma) \mathrm{d}\mathbf{y} \qquad (2)$$

where $h_i(\mathbf{y}) = \mathbb{I}\{y_i > \eta\}$ and $t_{j,i}(\mathbf{y}) = \mathbb{I}\{y_j \leq \eta\}$ if $i = j$ and $t_{j,i}(\mathbf{y}) = \mathbb{I}\{0 \leq y_i - y_j\}$ otherwise.

To obtain a surrogate $\hat{\mathcal{F}}_n(\mathbf{x}_*)$ for $\mathcal{F}_n(\mathbf{x}_*)$ we use some ideas that have been recently developed in the batch Bayesian optimisation literature. In a nutshell, batch BO methods aim to define sets of points in $\mathcal{X}$ where f should be evaluated in parallel, rather than sequentially. In essence, a key aspect to building a 'good' batch is the same as to computing a good approximation for $\Lambda_n(\mathbf{x}_*|\mathcal{I}_0)$: to find a set of good locations at which to evaluate the objective. In this work we predict the steps ahead using the method proposed in by [3] since it is computationally tractable and scales well with the size of the batches (steps ahead in our context) and the dimensionality of the problem.

4 Experiments

To study the validity of our approximation we choose a variety of functions with a range of dimensions and domain sizes. We use the full GLASSES algorithm and we show the results when 2, 3, 5, and 10 steps look-ahead are used to compute the loss function. Each problem is solved 5 times with different random initialisations of 5 data points. The number of allows evaluations is 10 times the dimensionality of the problem. For comparative purposes we used other two loss functions are commonly used in the literature: the Maximum Probability of Improvement, MPI, and the GP lower confidence bound, GP-LCB (see [16]). All acquisition functions are optimised using the dividing rectangles algorithm DIRECT [6]. As surrogate model for the functions we used a GP with a squared exponential kernel plus a bias kernel [15]. The hyper-parameters of the model were optimised by the standard method of maximising the marginal likelihood, using L-BFGS [12] for 1,000 iterations and 5 random restarts. To compare the methods we used the 'gap' measure of performance [5]. Table 1 shows the comparative across different functions and methods. None of the methods used is universally the best but a non myopic loss is the best in 6 of the 11 cases. In 3 cases the full GLASSES approach is the best of all methods. Note as well that when the GLASSES algorithm is not the best global method it typically performs closely to the best alternative.

5 Conclusions

For the first time in the literature, we have proposed an non-myopic loss that allows taking into account dozens of future evaluations before making the decision of where to sample the objective function. As previously suggested in [14], our results confirm that using a non-myopic loss helps, in practice, to solve global optimisation problems. Interestingly, and as happens with any comparison of loss functions across many objective functions, there is not a universal best method. However, in cases in which GLASSES is not superior, it performs very closely to the myopic loss, which makes it an interesting default choice in most scenarios.

References

[1] Eric Brochu, Vlad M. Cora, and Nando De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[2] Roman. Garnett, Michael. A. Osborne, and Stephan. J. Roberts. *Bayesian optimization for sensor set selection*, page 209219. ACM, 2010.

[3] Javier González, Zhenwen Dai, Philipp Hennig, and Neil D Lawrence. Batch Bayesian optimization via local penalization. *arXiv preprint arXiv:1505.08052*, 2015.

[4] Javier González, Joseph Longworth, David James, and Neil Lawrence. Bayesian optimisation for synthetic gene design. *NIPS Workshop on Bayesian Optimization in Academia and Industry*, 2014.

[5] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng. Global optimization of stochastic black-box systems via sequential kriging meta-models. *J. of Global Optimization*, 34(3):441–466, March 2006.

[6] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *J. Optim. Theory Appl.*, 79(1):157–181, October 1993.

[7] Donald R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345383, 2001.

[8] Daniel James Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008. AAINR46365.

[9] Roman Marchant, Fabio Ramos, and Scott Sanner. Sequential Bayesian optimisation for spatial-temporal monitoring. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2014.

[10] Ruben Martinez-Cantin, Nando de Freitas, Eric Brochu, Jos Castellanos, and Arnaud Doucet. A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(2):93–103, August 2009.

[11] Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, UAI '01, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[12] Jorge Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, 1980.

[13] Michael Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature*. PhD thesis, University of Oxford, 2010.

[14] Michael A. Osborne, Roman Garnett, and Stephen J. Roberts. Gaussian processes for global optimization. In *3rd international conference on learning and intelligent optimization (LION3)*, pages 1–15, 2009.

[15] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[16] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. *Practical Bayesian optimization of machine learning algorithms*, page 29512959. 2012.

[17] Simon Streltsov and Pirooz Vakili. A non-myopic utility function for statistical global optimization algorithms. *J. Global Optim.*, 14(3):283–298, 1999.