GLASSES: Relieving The Myopia Of Bayesian Optimisation

Anonymous Author(s)
Affiliation
Address
email

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 [13], sensor selection [6] or experimental design [11]. 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. [12] 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. [10] 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.

There is a clear link between the multi-step lookahead problem and that considered in the literature as *batch* Bayesian optimisation. The two problems are distinct but related: the multi-step lookahead problem requires the challenging marginalisation over unknown future evaluation *locations*, in addition to the unknown future evaluation *values* also marginalised by batch approaches. Similarly to the state-of-the-art in multi-step lookahead, the batch literature provides only poor scaling with the number of evaluations. [7] present results for no more than six simultaneous function evaluations. [1, 2] use the surrogate model for f to generate 'fake' observations and avoid the marginalization

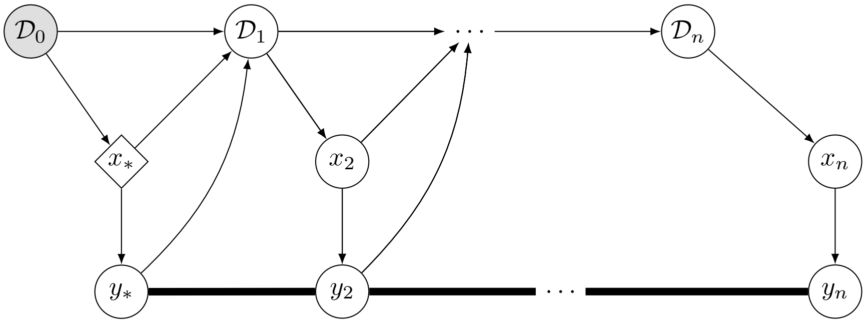

Figure 1: A Bayesian network describing the n-step lookahead problem. The shaded node ($\mathcal{D}_0$) is known, and the diamond node (x_*) is the current decision variable. All y nodes are correlated with one another under the GP model.

step. This produce a large accumulation of errors that does not allow the use of these techniques for the collection of large batches.

We propose an algorithm, GLASSES, that provides scaling superior to existing alternatives.

2 Bayesian Optimisation

[9] [8] [13] [4]

3 Look-Ahead through Stochastic Simulation and Expected-loss Search (perhaps other title is better?)

3.1 Predicting future algorithm evaluations

3.2 Expectation Propagation to compute the Expected Loss

Our goal is to compute $E_{p(\mathbf{y})}[\min(\mathbf{y},\eta)]$ for $\mathbf{y} = \{y_1,\dots,y_n\}$ and $p_0(\mathbf{y}) \sim \mathcal{N}(\mathbf{y};\mu,\Sigma)$. This approximates the n-steps ahead expected loss $\Lambda_n(\mathbf{x}_*)$. The goal of this section is to write $E_{p(\mathbf{y})}[\min(\mathbf{y},\eta)]$ in a way that it is suitable to be computed by Expectation Propagation. Next proposition will do the work (I think).

Proposition 1

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

where $h_i(\mathbf{y}) = \mathbb{I}\{y_i > \eta\}$ *and*

$$t_{j,i}(\mathbf{y}) = \begin{cases} \mathbb{I}\{y_j \leq \eta\} & \textit{if i=j} \\ \mathbb{I}\{0 \leq y_i - y_j\} & \textit{otherwise.} \end{cases}$$

All the elements in (1) can be rewritten in a way that can be computed using EP but the work in [5].

- The second term is a Gaussian probability on unbounded polyhedron in which the limits are aligned with the axis.
- The first term requires some more processing but it is still computable under the assumptions in [5]. Let $\mathbf{w}_j$ the jth canonical vector. Then we have that

$$\begin{aligned} \int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(\mathbf{y})\mathcal{N}(\mathbf{y};\mu,\Sigma)d\mathbf{y} &= \mathbf{w}^T\int_{\mathbb{R}^n} \mathbf{y} \prod_{i=1}^{n} t_{j,i}(\mathbf{y})\mathcal{N}(\mathbf{y};\mu,\Sigma)d\mathbf{y} && (2) \\ &= \mathbf{w}^T E[\mathbf{y}]z_j && (3) \end{aligned}$$

Algorithm 1 Decision process of the GLASSES algorithm.

```
Input: dataset $\mathcal{D}_0 = \{(\mathbf{x}_0, y_0)\}$, number of remaining evaluations ($n$), representer points ($r$) and DPP replicates (s).

Fit a GP with kernel $k$ to $\mathcal{D}_0$.
Select $\mathbf{x}_{1*}, \dots, \mathbf{x}_{r*}$ representer points of the loss.
for $j = 1$ to $r$ do
    Take $s$ samples from a conditional n-DPP of kernel $k$ given $\mathbf{x}_{j*}$.
    Approximate the expected loss at $\mathbf{x}_j^*$ for the $s$ samples computing $E[\min(\mathbf{y}, \eta)]$.
    Average the expected loss for the $s$ samples and obtain $\tilde{\Lambda}_n(\mathbf{x}_j^*)$.
end for
Approximate $\Lambda_n(\mathbf{x}_*)$ fitting a GP$_2$ to $\{(\mathbf{x}_{j*}, \tilde{\Lambda}_n(\mathbf{x}_{j*}))\}_{j=1}^{r}$ with posterior mean $\mu_2$.
Returns: New location at $\arg\min_{x \in \mathcal{X}} \{\mu_2(\mathbf{x})\}$.
```

where the expectation is calculated over the normalized distribution over P_j, the one EP approximates with $q(\mathbf{y})$, and for z_j being the normalizing constant

$$z_j = \int_{\mathbb{R}^n} \prod_{i=1}^{n} t_{j,i}(\mathbf{y}) \mathcal{N}(\mathbf{y}; \mu, \Sigma) d\mathbf{y}$$

Because EP does moments matching, both the normalizing constant and the expectation are available.

3.3 Approximating the Expected loss function

[We can describe]

3.4 Algorithm

4 Results

5 Conclusions

References

[1] Javad Azimi, Ali Jalali, and Xiaoli Fern. Dynamic batch Bayesian optimization. *CoRR*, abs/1110.3347, 2011.

[2] Javad Azimi, Ali Jalali, and Xiaoli Zhang Fern. Hybrid batch Bayesian optimization. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[3] Alexei Borodin and Eric M. Rains. Eynard–Mehta theorem, Schur process, and their Pfaffian analogs. *Journal of Statistical Physics*, 121(3-4):291–317, Nov 2005.

[4] 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.

[5] John P. Cunningham, Philipp Hennig, and Simon Lacoste-Julien. Gaussian probabilities and expectation propagation. *arXiv:1111.6832 [stat]*, Nov 2011. arXiv: 1111.6832.

[6] R. Garnett, M. A. Osborne, and S. J. Roberts. *Bayesian optimization for sensor set selection*, page 209–219. ACM, 2010.

[7] David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. A multi-points criterion for deterministic parallel global optimization based on Gaussian processes. *HAL: hal-00260579*, 2009.

[8] Donald R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383, 2001.

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

[10] 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.

[11] Ruben Martinez-Cantin, Nando de Freitas, Eric Brochu, José 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.

[12] 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.

[13] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. *Practical Bayesian optimization of machine learning algorithms*, page 2951–2959. 2012.

Supplementary materials for: ‘GLASSES: Relieving The Myopia Of Bayesian Optimisation”

Authors here

S1 Proofs

Proof 1 *Denote by*

$$\begin{aligned} E_{p(\boldsymbol{y})}[\min(\boldsymbol{y},\eta)] &= \int_{\mathbb{R}^n} \min(\boldsymbol{y},\eta)\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} \\ &= \int_{\mathbb{R}^n-(\eta,\infty)^n} \min(\boldsymbol{y})\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} + \int_{(\eta,\infty)^n} \eta\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} \end{aligned}$$

The first term can be written as follows:

$$\int_{\mathbb{R}^n-(\eta,\infty)^n} \min(\boldsymbol{y})\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} = \sum_{j=1}^{n}\int_{P_j} y_j\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y}$$

where $P_j := \{\boldsymbol{y} \in \mathbb{R}^n - (\eta,\infty)^n : y_j \leq y_i,\ \forall i \neq j\}$. We can do this because the regions P_j are disjoint and it holds that $\cup_{j=1}^{n} P_j = \mathbb{R}^n - (\eta,\infty)^n$. Also, note that the $\min(\boldsymbol{y})$ can be replaced within the integrals since within each P_j it holds that $\min(\boldsymbol{y}) = y_j$. Rewriting the integral in terms of indicator functions we have that

$$\sum_{j=1}^{n}\int_{P_j} y_j\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} = \sum_{j=1}^{n}\int_{\mathbb{R}^n} y_j \prod_{i=1}^{n} t_{j,i}(\boldsymbol{y})\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} \tag{S.1}$$

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

The second term can be written as

$$\int_{(\eta,\infty)^n} \eta\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} = \eta\int_{\mathbb{R}^n} \prod_{i=1}^{n} h_i(\boldsymbol{y})\mathcal{N}(\boldsymbol{y};\mu,\Sigma)d\boldsymbol{y} \tag{S.1}$$

where $h_i(\boldsymbol{y}) = \mathbb{I}\{y_i > \eta\}$. Merge (S.1) and (1) to conclude the proof.

S2 DPPs (old)

To approximate $\Lambda_n(\mathbf{x}_*)$ we compute $E_{p(\mathbf{y})}[\min(\mathbf{y},\eta)]$ where $\mathbf{y} = \{y_1,\ldots,y_n\}$ with $p(\mathbf{y}) \sim \mathcal{N}(\mathbf{y};\mu,\Sigma)$ is the multivariate random vector that we obtain after evaluation the predictive distribution of the GP at locations $\mathbf{x}_*, \mathbf{x}_2, \ldots, \mathbf{x}_n$. Point $\mathbf{x}_*$ is fixed (is the point where we evaluate Λ_n) and we consider $\mathbf{x}_2,\ldots,\mathbf{x}_n$ to be a sample of the conditional k-DPP (for $k = n$) on $\mathbf{x}_*$ with kernel L (in principle, the one from the GP).

[QUICK DESCRIPTION OF WHY WE SAMPLE FROM A DPP GOES HERE]

Let $\mathbf{L}$ be the kernel matrix corresponding to the evaluation of L on a finite set Ω of potential points *i.e* pre-uniformly sampled in the domain of interest. The distribution obtained by conditioning on having observed $\mathbf{x}_* \in \Omega$ can be obtained as follows. Let $B \subset \Omega$ a non intersecting set with $\mathbf{x}_*$. We have that

$$p_L(\mathbf{x}_* \cup B | \mathbf{x}_* \subseteq Z) = \frac{p_L(Z = \mathbf{x}_* \cup B)}{p(\mathbf{x}_* \subseteq Z)} = \frac{\det(\mathbf{L}_{\mathbf{x}_* \cup B})}{\det(\mathbf{L} - \mathbf{I}_{\bar{\mathbf{x}}_*})} \tag{S.2}$$

where $\mathbf{I}_{\bar{\mathbf{x}}_*}$ is the matrix with ones in the diagonal entries indexed by elements of $\Omega - \mathbf{x}_*$ and zeros elsewhere. This conditional distribution is again a DPP over subsets of $\Omega - \mathbf{x}_*$ [3] with kernel

$$\mathbf{L}^{\mathbf{x}_*} = \left([(\mathbf{L} + \mathbf{I}_{\bar{\mathbf{x}}_*})^{-1}]_{\bar{\mathbf{x}}_*}\right)^{-1}.$$

$[\cdot]_{\bar{\mathbf{x}}_*}$ represents the restriction of the matrix to all rows and columns not indexed by $\mathbf{x}_*$. The previous inverses exist if and only if the probability of $\mathbf{x}_*$ appearing is nonzero, as is the case in our context. A second marginalization is later needed to generate samples of size k. Figure 2 shows an example of samples from a k-DPP and a conditional k-DPP with the same kernel.

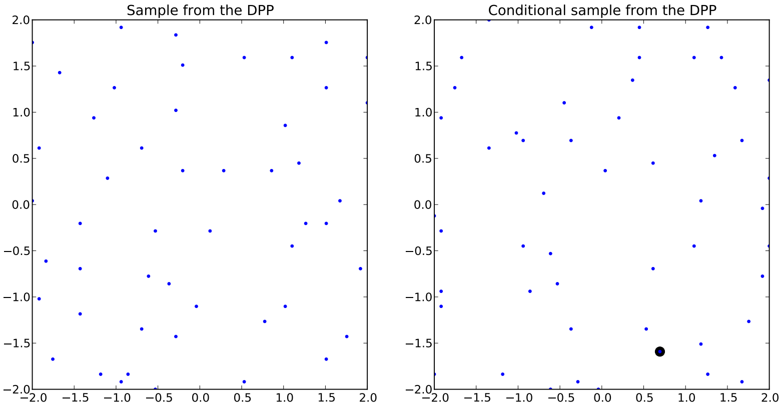

Figure 2: Left: sample from a k-DPP (k=50) for a SE kernel with length-scale 0.5. Right: sample from a k-DPP (k=50) conditional to x_1 (black dot) being in the selected set.