Scalable Constrained Bayesian Optimisation using an Augmented Lagrangian

Thomas Christie
Unaffiliated[1]

Carl Henrik Ek
University of Cambridge

Vincent Dutordoir
University of Cambridge

Abstract

Constrained optimisation problems in which the objective and constraint functions are *black-boxes* are ubiquitous in the physical world. Recent work has utilised *trust-regions* to make high-dimensional instances of such problems more tractable to solve with tools taken from the (constrained) Bayesian optimisation literature. However, the acquisition function used in the state-of-the-art approach has diverged from the classical constrained optimisation literature, and is unsuitable for optimisation with a gradient-based optimiser. In this paper we propose the Thompson sampling augmented Lagrangian (TSAL). Our method is grounded in a technique taken from the classical optimisation literature, the *augmented Lagrangian*, resulting in an acquisition function which is amenable to gradient-based optimisation, can easily suggest batches of points to be evaluated in parallel, and bridges the gap between classical and probabilistic constrained optimisation in an elegant manner.

1 Introduction

Countless problems in the physical world involve optimising functions for which the explicit functional form is unknown, but which can be expensively queried throughout their domain. In the domain of science, examples include designing new molecules (Pyzer-Knapp et al., 2015, Griffiths and Hernández-Lobato, 2020), or discovering novel materials with optimised properties (Hachmann et al., 2014). Even within the field of machine learning, the ability to automatically optimise model architectures, whilst minimising the number of expensive training runs required, has garnered considerable recent attention within the field of neural architecture search (Elsken et al., 2019, White et al., 2021).

However, these problems are often made more difficult due to the presence of *constraints*. For instance, whilst engineers may vary the dimensions of certain components of a car in order to minimise its weight, the car is likely to be subject to several constraints, such as collision safety requirements (Kohira et al., 2018).

A great deal of work has been done within the field of Bayesian optimisation (BO) to tackle unconstrained black-box optimisation problems (e.g. Močkus (1975), Thompson (1933)). Whilst historically these methods have largely been applied to low-dimensional problems which require fairly few evaluations to solve, recent work has focused on scaling these techniques to tackle more challenging, high-dimensional problems. For instance, Thompson sampling (Thompson, 1933) has been shown to be a promising candidate for scaling to large evaluation budgets, with the fact that it is trivially parallelisable enabling it to suggest *batches* of points to be queried at a time (Hernández-Lobato et al., 2017, Wang et al., 2018, Vakili et al., 2021). Recent work (Diouane et al., 2023, Eriksson et al., 2019) has also shown that *trust-regions* are a promising tool for mitigating the curse of dimensionality, enabling BO techniques to be used to tackle higher-dimensional black-box optimisation problems, consisting of *hundreds* of parameters to optimise.

At the same time, several methods have been proposed to tackle constrained black-box optimisation problems, often defining acquisition functions which combine elements of unconstrained BO and classical constrained optimisation, in which the explicit functional form of both the objective and constraints is known (Picheny et al., 2016, Gramacy et al., 2016a, Pourmohamad and Lee, 2022). Eriksson and Poloczek (2021) have recently utilised a combination of Thompson sampling

Preliminary work. Under review.

[1]This work was conducted whilst the author was studying at the University of Cambridge.

and trust-regions within the context of constrained BO, and demonstrated the success of their approach on some challenging benchmarks, which were both high-dimensional and consisted of many constraints.

However, the acquisition function used by Eriksson and Poloczek diverges from the classical constrained optimisation literature, and is not amenable to gradient-based optimisation as it is not continuous. In order to obtain a *differentiable* acquisition function, whilst enjoying the benefits of the approach proposed by Eriksson and Poloczek (2021), we propose the *Thompson sampling augmented Lagrangian (TS-AL)*. Our approach satisfies the following desirable properties:

- It is trivially parallelisable, enabling *batches* of points to be queried at a time.
- It utilises trust-regions, enabling it to tackle high-dimensional instances of black-box constrained optimisation problems.
- It scales to large datasets, as it is compatible with sparse Gaussian processes (Snelson and Ghahramani, 2005, Titsias, 2009), enabling it to be utilised for challenging problems which may require *thousands* of iterations to solve.
- The acquisition function defined is amenable to *gradient-based optimisation*, facilitating its maximisation when choosing which point(s) to query next.

2 Related Work

The first attempts to solve the problem of constrained Bayesian optimisation (Schonlau et al., 1998, Gelbart et al., 2014, Gardner et al., 2014), were based on *expected constrained improvement* (`ECI`). For a given point within the domain of interest, this acquisition function is defined as the expected improvement (EI) multiplied by the probability that all of the constraints are satisfied at the given point. However, Gramacy et al. (2016b) showed that this approach can suffer on problems with small feasible regions, as in this case the acquisition function can be zero almost everywhere, making it very difficult to guide the optimiser towards the optimal point. A similar issue arises on problems with many constraints, as the probability of *all* the constraints being satisfied at a given point can become very small in large regions of the search space, making the acquisition function difficult to optimise. Nonetheless, it has demonstrated strong performance on many constrained optimisation problems.

Hernández-Lobato et al. (2015) introduced an information-theoretic acquisition function; *predictive entropy search with constraints*. This utilises several approximations to compute the expected information gain about the global constrained minimum of the black-box objective from observing a given point. A particular strength of this approach is its performance in the *decoupled* setting, where one may independently query the objective and constraint functions, rather than having to evaluate all functions at each queried point. This can be useful in certain classes of problems where the expense of querying the different functions of interest varies significantly.

Gramacy et al. (2016a) were the first to draw on the classical constrained optimisation literature, defining an acquisition function which combined expected improvement with the augmented Lagrangian (Wright et al., 1999), an apparatus commonly used in the classical constrained optimisation literature. Their approach forms an augmented Lagrangian from the posterior distributions of the surrogate models, and then calculates the expected improvement of this augmented Lagrangian. In order to calculate the EI, the authors resort to Monte Carlo approximations. However, in subsequent work (Picheny et al., 2016) they refined their augmented Lagrangian expression through the introduction of slack variables. The authors noted that utilising slack variables resulted in the subsequent augmented Lagrangian following the weighted non-central Chi-square distribution. This meant that the EI could be deterministically calculated, without resorting to Monte Carlo approximations. This determinism enabled gradient-based optimisers, such as L-BFGS-B (Byrd et al., 1995), to be used to further optimise the acquisition function. Several subsequent approaches have also drawn on techniques from the classical constrained optimisation literature. For instance, Ariafar et al. (2019) used the alternating direction method of multipliers for constrained BO in the decoupled setting, and Pourmohamad and Lee (2022) proposed using barrier methods (Wright et al., 1999) within the context of constrained BO.

However, the use of Thompson sampling (TS) for constrained BO is still very under-explored. To the best of our knowledge, the only attempt to incorporate TS into the constrained BO framework has been by Eriksson and Poloczek (2021), which utilises a trust-region approach to fit *local* surrogate models to regions of the domain of interest, and then uses TS to query the regions. They also apply transformations to the objective and constraint functions to emphasise regions of interest in both, and from their N point Thompson samples query the feasible point with the minimal posterior objective function value. If no points satisfy all the constraints, then the point which violates the constraints least is selected. This

approach has demonstrated excellent performance on high-dimensional problems, although the acquisition function deviates from the classical optimisation literature, and is discontinuous at the boundary of the feasible region, preventing it from being optimised with gradient-based optimisers.

3 Problem Formulation

Mathematically, constrained optimisation problems take the following form:

$$\min_{\mathbf{x}\in X} f(\mathbf{x}) \quad \text{such that} \begin{cases} g_i(\mathbf{x}) \leq 0 & \text{for} \quad i = 1, \ldots, m \\ h_i(\mathbf{x}) = 0 & \text{for} \quad i = 1, \ldots, n \end{cases} \tag{1}$$

with f being the objective function, g_i being a set of *inequality constraints* and h_i being a set of *equality constraints*. Alternatively, *slack variables* can be introduced for each of the inequality constraints, $s_i(\mathbf{x}) \geq 0$, so that the condition enforcing the satisfaction of the inequality constraints becomes $g_i(\mathbf{x}) + s_i(\mathbf{x}) = 0$.

As is commonly the case in the constrained BO literature, we use Gaussian processes (GPs) (Williams and Rasmussen, 2006) as surrogate models for the functions of interest, which we shall denote using capital letters, such that F, G_i and H_i represent surrogate models for the objective, inequality constraints and equality constraints respectively. We shall denote *samples* from the GP posteriors using a tilde so that, for instance, a sample from the GP modelling the objective function will be denoted $\tilde{f} \sim F$. The tilde notation shall be carried over to functions composed of GP samples such that, for instance, whilst an augmented Lagrangian composed of the ground truth functions may be denoted $\mathcal{L}_A(\cdot)$, an augmented Lagrangian formed with samples from the GPs will be denoted $\tilde{\mathcal{L}}_A(\cdot)$.

4 Thompson Sampling Augmented Lagrangian (TS-AL)

We now discuss our proposed algorithm, the *Thompson sampling augmented Lagrangian (TS-AL)*. Given the extensive work done in the field of classical constrained optimisation, our approach aims to frame the problem of black-box constrained optimisation in such a way that it is amenable to being tackled with these techniques. Thompson sampling offers an elegant bridge between these two fields. At each iteration of the BO loop we draw samples from the GP posteriors, resulting in fixed, concrete instances of constrained optimisation problems, which may be tackled with techniques taken from the classical constrained optimisation literature. At the same time, drawing new samples from the GP posteriors at each iteration provides a way to naturally balance exploration and exploitation, enabling one to converge upon solutions in a sample-efficient manner.

We utilise *decoupled sampling* introduced in Wilson et al. (2020) to efficiently draw samples from the GP posteriors. This enables GP posterior samples to be drawn in a manner which scales linearly with the number of points at which the sample is evaluated, $O(N)$, mitigating the cubic complexity incurred by the Cholesky decomposition commonly utilised when drawing *exact* samples from GP posteriors. It should be noted that using *exact* GPs results in a time complexity which scales cubically with the number of observations, $O(n^3)$, although this can be mitigated through the use of sparse GPs (Snelson and Ghahramani, 2005, Titsias, 2009).

In order to scale to high-dimensional problems we follow the *trust-regions* approach employed by Eriksson and Poloczek (2021). This restricts the volume of the search space being explored at a time, enabling the GP surrogates to more accurately model the functions of interest in the region currently being explored, as well as helping to mitigate the tendency of BO methods to *over-explore* in high-dimensional problems. This technique is also highly modular, enabling it to be easily utilised with most existing BO methods.

TS-AL Acquisition Function At the heart of our acquisition function lies the augmented Lagrangian, a commonly used approach for solving constrained optimisation problems when the explicit functional form of both the objective and constraints are known (Wright et al., 1999). The augmented Lagrangian is defined as follows:

$$\mathcal{L}_A(\mathbf{x}, s; \boldsymbol{\mu}, \boldsymbol{\lambda}, \rho) = f(\mathbf{x}) + \sum_{i=1}^{m} \boldsymbol{\mu}_i (g_i(\mathbf{x}) + s_i(\mathbf{x})) + \tag{2}$$
$$\sum_{i=1}^{n} \boldsymbol{\lambda}_i h_i(\mathbf{x}) + \frac{1}{2\rho} \sum_{i=1}^{m} (g_i(\mathbf{x}) + s_i(\mathbf{x}))^2 + \frac{1}{2\rho} \sum_{i=1}^{n} h_i(\mathbf{x})^2$$

with Lagrange multipliers $\boldsymbol{\mu} \in \mathbb{R}^m_{\geq 0}$, $\boldsymbol{\lambda} \in \mathbb{R}^n$, slack variables $s_i(\mathbf{x}) \geq 0$ and penalty parameter $\rho \in \mathbb{R}_{>0}$. The augmented Lagrangian is an *unconstrained* function, and hence may be minimised with a conventional optimiser such as L-BFGS-B (Liu and Nocedal, 1989).

The augmented Lagrangian method is iterative, and over time the penalty parameter ρ is reduced and the Lagrange multipliers $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ are updated. Reducing the penalty parameter ρ results in regions of

space in which the constraints are violated getting penalised more heavily, forcing the unconstrained minimiser of $\mathcal{L}_A(\cdot)$ to lie increasingly close to the valid region. At the same time, as ρ is reduced, the resulting Lagrangian becomes increasingly difficult to optimise with optimisers such as L-BFGS-B, with the gradient at the boundary of the feasible region becoming increasingly large. The introduction of estimates for the Lagrange multipliers in Equation (2) is done to ameliorate this problem, resulting in the unconstrained minimisers of $\mathcal{L}_A(\cdot)$ lying within the valid region for more mild values of ρ, as described by Wright et al. (1999). However, in practise, we found that it was sometimes still difficult to reliably use L-BFGS-B to optimise the acquisition function across all test problems, and instead used a modified version of Adam to perform minimisation of $\mathcal{L}_A(\cdot)$. Full implementation details are given in Appendix A .

While the use of slack variables in Equation (2) may ostensibly complicate matters by increasing the input dimensionality of the problem, Picheny et al. (2016) showed that, given ρ, $\boldsymbol{\lambda}_i$, and $g_i(\mathbf{x})$, the corresponding slack variable which minimises the augmented Lagrangian may be obtained analytically. We subsequently discuss the details in more depth.

Thompson sampling (Thompson, 1933) offers a straightforward path for adapting the augmented Lagrangian presented in Equation (2) to the case where the objective and constraint functions are unknown. Instead of utilising the true functions f, g_i and h_i, at each iteration of the BO loop we may instead draw a sample from the posterior distribution of each of the GPs modelling these functions, and substitute these samples into Equation (2). This forms the acquisition function for our approach, which we denote $\tilde{\mathcal{L}}_A(\cdot)$, which can then be minimised using an optimiser such as L-BGFS-B (Liu and Nocedal, 1989) or Adam (Kingma and Ba, 2014).

Trust Regions We follow the same setup as Eriksson and Poloczek (2021) when utilising trust-regions. To summarise, at the start of the BO loop, a trust region is initialised as a hypercube with side length $L = L_{\text{init}}$. The center of the trust-region, C, is chosen to be the best point observed so far. This is defined as the best feasible point if any have been observed, and in the case that no feasible points have been observed, it is defined as the point with the minimum total violation of constraints. Each iteration of the BO loop is categorised as a *success* if it discovers a point which improves on the best point observed so far, or a *failure* if not. A *success counter*, n_s, is incremented upon each successful iteration, and is reset to 0 upon an unsuccessful iteration. Similarly, a *failure counter*, n_f, is incremented upon each failure, and is reset to 0 on a successful iteration. Success and failure tolerances, denoted τ_s and τ_f respectively, define when the trust-region is resized. When $n_s = \tau_s$, $L = \max\{2L, L_{\max}\}$, and when $n_f = \tau_f$, $L = L/2$, with n_s and n_f getting reset to 0 when the trust-region is resized. If L reaches a specified minimum, $L_{\min}$, the side length gets reset to L_{init}. Full details of the trust-region procedure are provided in Appendix A.

TS-AL Summary We now provide a summary of TS-AL, with a more technical description provided in Algorithm 1. Initially, the ground truth objective function, f, and constraint functions, g_i, h_i, are queried at k random points, with observations stored in dataset $\mathcal{D}$. These observations are used to fit the initial GP surrogate models F, G_i and H_i, and determine the center of the initial trust-region, the domain of which we denote $\mathcal{B}$. The penalty parameter, ρ^0, and Lagrange multipliers, $\boldsymbol{\mu}^0, \boldsymbol{\lambda}^0$ are also initialised, using defaults we describe in Appendix A. After this, the following steps are repeated until the sampling budget has been exhausted, with the GPs updated and new samples drawn on each iteration:

1. Form the augmented Lagrangian using samples drawn from the GP surrogate models, and let $(\mathbf{x}_i, s_i) = \underset{\mathbf{x}\in\mathcal{B},s}{\operatorname{argmin}}\{\tilde{\mathcal{L}}_A(\mathbf{x}, s; \boldsymbol{\mu}^{i-1}, \boldsymbol{\lambda}^{i-1}, \rho^{i-1})\}$. Picheny et al. (2016) showed that the slack-variables s can be chosen optimally as a function of $\mathbf{x}$: $s_j^*(\mathbf{x}) = \max\{0, -\boldsymbol{\lambda}_j\rho - g_j(\mathbf{x})\}$ for $j = 1, \ldots, m$. Note that $g_j(\mathbf{x})$ is generally unknown, so we instead utilise the value of the corresponding sample from the surrogate model, $\tilde{g}_j(\mathbf{x})$. Minimisation of $\tilde{\mathcal{L}}_A$ is then done using an optimiser, such as L-BFGS-B or Adam. Whilst we assume the objective function and all constraint functions are unknown, this method is easily adapted in the case that some of these functions are known; in the augmented Lagrangian expression one merely has to utilise the known function instead of the sample from the corresponding surrogate model.

2. Evaluate the ground truth objective and constraint functions at the suggested point $\mathbf{x}_i$, adding the observations to the dataset $\mathcal{D}$.

3. Let $(\mathbf{x}^*, s^*) = \underset{\mathbf{x}\in\mathcal{D},s}{\operatorname{argmin}}\{\mathcal{L}_A(\mathbf{x}, s; \boldsymbol{\mu}^{i-1}, \boldsymbol{\lambda}^{i-1}, \rho^{i-1})\}$. Here we use the *true* augmented Lagrangian, since we have access to the true values of the objective and constraint functions at the previously queried locations, stored in $\mathcal{D}$. These values, $\mathbf{x}^*$ and s^*, are used to update the penalty parameter and Lagrange multiplier estimates. In the classical case, this is always done using the

Algorithm 1 Thompson sampling augmented Lagrangian

1: Initialise dataset $\mathcal{D}$ with k initial observations from the objective and constraint functions: $\mathcal{D} = \{(f(\mathbf{x}_i), g_1(\mathbf{x}_i), \dots, g_m(\mathbf{x}_i), h_1(\mathbf{x}_i), \dots, h_n(\mathbf{x}_i))\}_{i=1}^k$.
2: Initialise surrogate models $F, G_1, \dots, G_m, H_1, \dots, H_n$ using dataset $\mathcal{D}$, and draw samples $\tilde{f}, \tilde{g}_1, \dots, \tilde{g}_m, \tilde{h}_1, \dots, \tilde{h}_n$ for use in the augmented Lagrangian.
3: Initialise penalty ρ^0, Lagrange multipliers $\boldsymbol{\lambda}^0, \boldsymbol{\mu}^0$ and equality constraint tolerance ϵ.
4: Initialise trust-region $\mathcal{B}$ with sides of length $L = L_{\text{init}}$ and center C at the best of the initially sampled points. The best point is defined as the minimum of the feasible points or, if no observed points satisfy all constraints, that with the minimum sum of constraint violations.
5: **for** $i = 1, \dots,$ *function_evaluation_budget* **do**
6: Let $(\mathbf{x}_i, s_i) = \underset{\mathbf{x}\in\mathcal{B}, s}{\operatorname{argmin}}\{\tilde{\mathcal{L}}_A(\mathbf{x}, s; \boldsymbol{\mu}^{i-1}, \boldsymbol{\lambda}^{i-1}, \rho^{i-1})\}$, with optimal slack values s_j^* defined analytically as $s_j^*(\mathbf{x}) = \max\{0, -\boldsymbol{\lambda}_j \rho - \tilde{g}_j(\mathbf{x})\}$ for $j = 1, \dots, m$.
7: Evaluate the true objective and constraint functions at $\mathbf{x}_i$ and augment dataset $\mathcal{D}$ with the observations: $\mathcal{D} = \mathcal{D} \cup \{(f(\mathbf{x}_i), g_1(\mathbf{x}_i), \dots, g_m(\mathbf{x}_i), h_1(\mathbf{x}_i), \dots, h_n(\mathbf{x}_i))\}$.
8: Let $(\mathbf{x}^*, s^*) = \underset{\mathbf{x}\in\mathcal{D}, s}{\operatorname{argmin}}\{\mathcal{L}_A(\mathbf{x}, s; \boldsymbol{\mu}^{i-1}, \boldsymbol{\lambda}^{i-1}, \rho^{i-1})\}$.
9: Update Lagrange multipliers $\boldsymbol{\lambda}_j^i = \boldsymbol{\lambda}_j^{i-1} + \frac{1}{\rho^{i-1}} h_j(\mathbf{x}^*)$ for $j = 1, \dots, n$ and $\boldsymbol{\mu}_j^i = \boldsymbol{\mu}_j^{i-1} + \frac{1}{\rho^{i-1}}(g_j(\mathbf{x}^*) + s_j^*)$ for j $= 1, \dots, m$.
10: If $g_{1:m}(\mathbf{x}^*) \leq 0$ and $|h_{1:n}(\mathbf{x})| \leq \epsilon$, set $\rho^i = \rho^{i-1}$; else $\rho^i = \frac{1}{2}\rho^{i-1}$.
11: Update the trust-region by moving the center C, updating the side length L and updating the success and failure counters n_s and n_f using the rules described previously.
12: Update the surrogate models using updated dataset $\mathcal{D}$ and draw new samples $\tilde{f}, \tilde{g}_1, \dots, \tilde{g}_m, \tilde{h}_1, \dots, \tilde{h}_n$ for use in the augmented Lagrangian.
13: **end for**

most recently queried point, $\mathbf{x}_i$, as this is always the closest value to the true minimum of the function. However, in the surrogate model case, the most recently queried point may have been selected due to a misfit model, and may actually turn out to be worse than a previously observed point. This is why we perform the minimisation over $\mathbf{x} \in \mathcal{D}$.

4. Update the estimates for the Lagrange multipliers using the observation which minimises the augmented Lagrangian, $\mathbf{x}^*$. Concretely, at iteration i, for inequality constraints g_j with corresponding Lagrange multipliers $\boldsymbol{\mu}_j^i$, this results in the update $\boldsymbol{\mu}_j^i = \boldsymbol{\mu}_j^{i-1} + \frac{1}{\rho^{i-1}}(g_j(\mathbf{x}^*) + s_j^*)$. Similarly, for equality constraints h_j with Lagrange multipliers $\boldsymbol{\lambda}_j^i$, this results in the update $\boldsymbol{\lambda}_j^i = \boldsymbol{\lambda}_j^{i-1} + \frac{1}{\rho^{i-1}} h_j(\mathbf{x}^*)$.

5. Update the penalty parameter using the observation which minimises the augmented Lagrangian, $\mathbf{x}^*$. If this point violates the constraints, then the penalty parameter is halved in order to more heavily penalise areas of constraint violation, and force the minimiser of the augmented Lagrangian to lie within the valid region. If this point is valid, this suggests the penalty parameter is currently sufficiently "aggressive", and so it remains the same.

6. Update the trust region, moving the center to the best observed point $\mathbf{x} \in \mathcal{D}$ as defined previously, and updating the success/failure counters and side lengths accordingly.

Note that for notational clarity our description of the algorithm is for the case where the size of the batch of points queried at each iteration is $B = 1$. For batch size $B > 1$, the only difference is that we simply draw B samples from the GP posteriors at each iteration, and use these to form B augmented Lagrangians, which are then minimised in order to select a batch of B points to query. The same penalty parameter, ρ, and Lagrange multiplier estimates, $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$, are used for each of the augmented Lagrangians in the batch.

5 Empirical Results

We now provide a comparison of TS-AL with several of the state-of-the-art methods for constrained BO. We compare to the following baselines:

- The slack-variable augmented Lagrangian, introduced by Picheny et al. (2016), which we denote `EI-AL` as it uses *expected improvement* as the acquisition function on the augmented Lagrangian formed. We used the authors' original implementation, provided in the `R` package `laGP` (Gramacy, 2016), and made several minor changes to the code in order to fix a few minor bugs we found.

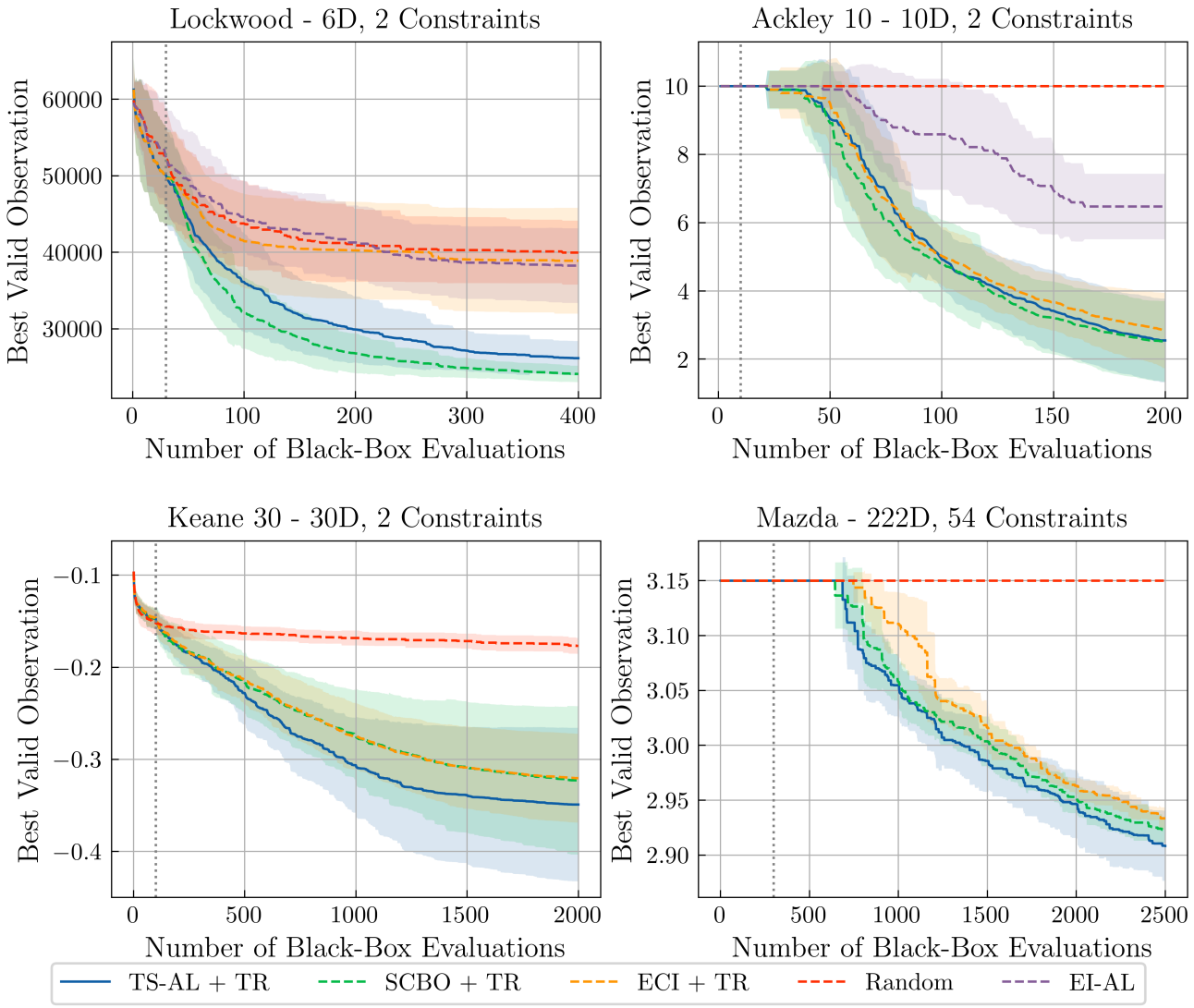

Figure 1: Mean (±1 STD) of the best valid objective value obtained against the number of black-box evaluations for each of the problems. The vertical grey dotted lines denote the start of the BO loop, with prior points selected randomly to initialise the models. All methods were run with trust-regions (`+ TR`) other than `EI-AL`.

The code used to obtain our results is provided in the supplementary material.

- Expected Constrained Improvement (`ECI`) (Gelbart et al., 2014, Gardner et al., 2014), for which we used the implementation provided in the `Trieste` library (Picheny et al., 2023). Note that the original `ECI` was only defined for batch size $B = 1$; we describe how we adapted this for batch sizes $B > 1$ in Appendix A.

- Scalable Constrained Bayesian Optimisation (`SCBO`) (Eriksson and Poloczek, 2021), which we re-implented in the `Trieste` library.

Due to the highly modular nature of trust-regions, we utilised the trust-region approach for *all* acquisition functions described above, other than `EI-AL`. This means that differences in the performance of the resulting algorithms do not arise as a result of (not) utilising trust-regions with the algorithms.

We compare the algorithms on four problems which we subsequently describe in detail. `EI-AL` was omitted on two of the problems as it was too slow, with the implementation provided in `laGP` (Gramacy, 2016) not scaling well to the problems which were run for over 1000 black-box evaluations. For all problems, we ran each algorithm 30 times, with initial sets of points sampled using a Halton sequence. The only exception was the Mazda problem, for which we only ran 6 repetitions of each algorithm due to the time taken to run each experiment. We plot the mean (± 1 STD) of the best valid objective value obtained by each of the algorithms against the number of black-box evaluations in Figure 1. Where no feasible solution has been found, we assign a default value which is equal to the largest valid objective value found on the problem, as done by Eriksson and Poloczek (2021).

5.1 Lockwood

The Lockwood problem is a 6D problem with 2 constraints. It was used by Gramacy et al. (2016a), and is based on preventing contaminated groundwater from spreading into the Yellowstone river in Lockwood, Montana by operating a system of pumps. The objective is to minimise the cost of running the system of pumps, whilst preventing any contaminated water from escaping beyond some specified boundaries. Evaluation of the objective and constraint functions is done using a simulation, the `C++` source code for which is available online [1]. For this problem we imposed a budget of 400 evaluations, with 30 initial points and a batch size of $B = 1$.

[1] `https://bobby.gramacy.com/surrogates/`

Figure 1 shows that `SCBO` obtained the best results on this problem, with `TS-AL` demonstrating the second-best performance. As previously discussed, the acquisition functions for `SCBO` and `TS-AL` are very similar, but `TS-AL` may be optimised with a gradient based optimiser. As such, we would expect `TS-AL` to obtain superior performance. However, this relies on the assumption that the surrogate models are accurately modelling their corresponding black-box functions. In the case that the surrogate models are misfitting the corresponding black-box functions, it could mean that truly minimising the acquisition function could lead to poorer performance as, for instance, the true minimiser of the acquisition function may lie just outside the valid region. On the other hand, the stochasticity introduced by not using gradient-based optimisation, and instead just choosing the minimum point of a discrete set of candidate points sampled on the acquisition function, may lead to slightly better performance on this problem.

An alternative hypothesis could be that the exploitative nature of trust-regions may have a negative impact on `TS-AL` on this problem. Trust-regions are inherently exploitative, in order to address the tendency to *over-explore* in high dimensional problems, but with them it is possible that sub-optimal early choices may lead to certain regions of the search space becoming difficult to reach due to being ruled out by the current trust-region. It may be the case that this has a more negative impact on `TS-AL` and `ECI` than `SCBO` on this problem, and we are currently running experiments *without* trust-regions to see whether using them can lead to *reduced* performance for some of the algorithms on problems of this low dimensionality.

5.2 Ackley 10

As in Eriksson and Poloczek (2021), we also evaluate the performance of our algorithm on the 10D Ackley function on the domain $[-5, 10]^{10}$, subject to the constraints $c_1(x) = \sum_{i=1}^{10} x_i \leq 0$ and $c_2(x) = ||x||_2 - 5 \leq 0$. This problem has a very small feasible region, with Eriksson and Poloczek (2021) noting that the probability of randomly selecting a feasible point is only 2.2×10^{-5}. For this problem we impose a budget of 200 evaluations, with 10 initial points and a batch size of $B = 1$.

On this problem we observed very similar performance for `ECI`, `TS-AL` and `SCBO`. Interestingly, in the comparison on this problem in Eriksson and Poloczek (2021), there was a notable performance gap between `ECI` and `SCBO`, but trust-regions were not used with `ECI` in their comparison, only with `SCBO`. This suggests that perhaps the trust-regions are responsible for the majority

of performance on this problem, and when paired with any "reasonable" constrained BO acquisition function, one may expect to see similar performance, regardless of the exact acquisition function.

5.3 Keane 30

We also provide an evaluation on the constrained optimisation problem utilising the Keane bump function (Keane, 1994) introduced by Eriksson and Poloczek (2021). This is a 30D problem consisting of 2 constraints, details of which are provided in Appendix B. For this problem we impose a budget of 2000 evaluations, with 100 initial points and a batch size of $B = 50$.

On this problem we can see that `TS-AL` obtained the best performance, with a mean best valid objective value of -0.349 after 2000 black-box evaluations, compared to -0.323 for `SCBO` and -0.320 for `ECI`. We propose that the amenability of `TS-AL` to acquisition with a gradient-based optimiser may be responsible for the improved performance observed.

5.4 Mazda

Finally, we evaluate the performance of the algorithms on the three car Mazda benchmark problem (Kohira et al., 2018). This problem requires tuning the thickness of 222 components used in three different car designs, with 54 black-box output constraints also being imposed. The original problem is a multi-objective problem, with the goal being to minimise the total weight of the three cars whilst maximising the number of components shared between them. We adapted this problem to be a single-objective problem by seeking to solely minimise the combined weight of the three car designs, whilst satisfying all constraints, without focusing on sharing components between the three cars.

Once again, we can see that `TS-AL`, `ECI` and `SCBO` all perform well on this problem, despite the fact that it is 222D, with 54 constraints, making it a very challenging problem by the standards of the constrained BO literature. It is worth noting that `TS-AL` does perform slightly better than the other two algorithms on this problem, which may once again be due to the fact that it's amenable to gradient-based optimisation. Perhaps more surprising is the performance of `ECI` on this problem. Often with problems with many constraints `ECI` would be expected to perform badly, as the probability of *all* constraints being satisfied can become vanishingly small as the number of constraints increases. This makes the acquisition function difficult to optimise, as it can become practically zero in the majority of the search space. Nonetheless, it seems to fare well on this problem with many constraints. It may be the case that the volume of the feasible region in this problem is sufficiently large to not cause `ECI` to suffer, despite the large number of constraints. Alternatively, the relatively good performance of `ECI` may be attributed to the use of trust-regions, leading to more accurate modelling of the underlying black-box functions, which may make it easier to discover the feasible region(s).

6 Concluding Discussion

In this paper we introduced the Thompson sampling augmented Lagrangian (TS-AL) for constrained black-box optimisation. We drew on the recent work of Eriksson and Poloczek (2021) on scaling constrained BO to challenging high-dimensional problems with the use of trust-regions, whilst providing an acquisition function which is grounded in the classical constrained optimisation literature. This results in an acquisition function which is amenable to *gradient-based* optimisation, making it easier to optimise than the acquisition function used in `SCBO`, which contains a discontinuity at the boundary of the feasible region.

We performed an empirical evaluation of `TS-AL` against several of the state-of-the-art constrained BO algorithms, finding that it matched, or outperformed, the state-of-the-art on most problems. Unlike the experiments performed by Eriksson and Poloczek (2021), we utilised trust-regions for *all* acquisition functions due to its highly modular nature, which revealed that *with* trust-regions, the gap in performance of the different acquisition functions was fairly small. This suggests that, in high-dimensional domains, the trust-regions may be doing the majority of the heavy lifting, with the choice of acquisition function itself playing less of a role in the performance of the algorithm. Further exploration of the interplay between acquisition function choice and the use of trust-regions could be useful future work.

From a computational complexity viewpoint, whilst our use of decoupled sampling scales linearly with the number of points at which the samples are evaluated, the use of *exact* GPs means that it still scales cubically with the number of observations used to fit the GPs. As such, in order to further improve the scalability of the algorithm, it would be useful to consider utilising sparse variational GPs (Titsias, 2009) for modelling the objective and constraint functions. In addition to this, the use of multi-output GPs may also be a promising avenue for further improving the sample-efficiency of our algorithm if there is correlation between some of the constraints/objective.

Acknowledgements

We would like to thank Henry Moss for useful discussions during this project.

References

Setareh Ariafar, Jaume Coll-Font, Dana H Brooks, and Jennifer G Dy. Admmbo: Bayesian optimization with unknown constraints using admm. *J. Mach. Learn. Res.*, 20(123):1–26, 2019.

Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing*, 16(5):1190–1208, 1995.

Youssef Diouane, Victor Picheny, Rodolophe Le Riche, and Alexandre Scotto Di Perrotolo. Trego: a trust-region framework for efficient global optimization. *Journal of Global Optimization*, 86(1):1–23, 2023.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

David Eriksson and Matthias Poloczek. Scalable constrained bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 730–738. PMLR, 2021.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local bayesian optimization. *Advances in neural information processing systems*, 32, 2019.

Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. Bayesian optimization with inequality constraints. In *ICML*, volume 2014, pages 937–945, 2014.

Michael A Gelbart, Jasper Snoek, and Ryan P Adams. Bayesian optimization with unknown constraints. *arXiv preprint arXiv:1403.5607*, 2014.

David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. Kriging is well-suited to parallelize optimization. In *Computational intelligence in expensive optimization problems*, pages 131–162. Springer, 2010.

Robert B Gramacy. lagp: large-scale spatial modeling via local approximate gaussian processes in r. *Journal of Statistical Software*, 72:1–46, 2016.

Robert B Gramacy, Genetha A Gray, Sébastien Le Digabel, Herbert KH Lee, Pritam Ranjan, Garth Wells, and Stefan M Wild. Modeling an augmented lagrangian for blackbox constrained optimization. *Technometrics*, 58(1):1–11, 2016a.

Robert B Gramacy, Genetha A Gray, Sébastien Le Digabel, Herbert KH Lee, Pritam Ranjan, Garth Wells, and Stefan M Wild. Rejoinder to modeling an augmented lagrangian for blackbox constrained optimization. *Technometrics*, 58(1):26–29, 2016b.

Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 11(2):577–586, 2020.

Johannes Hachmann, Roberto Olivares-Amaya, Adrian Jinich, Anthony L Appleton, Martin A Blood-Forsythe, László R Seress, Carolina Román-Salgado, Kai Trepte, Sule Atahan-Evrenk, Süleyman Er, et al. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry–the harvard clean energy project. *Energy & Environmental Science*, 7(2):698–704, 2014.

José Miguel Hernández-Lobato, Michael Gelbart, Matthew Hoffman, Ryan Adams, and Zoubin Ghahramani. Predictive entropy search for bayesian optimization with unknown constraints. In *International conference on machine learning*, pages 1699–1707. PMLR, 2015.

José Miguel Hernández-Lobato, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik. Parallel and distributed thompson sampling for large-scale accelerated exploration of chemical space. In *International conference on machine learning*, pages 1470–1479. PMLR, 2017.

AJ Keane. Experiences with optimizers in structural design. In *Proceedings of the conference on adaptive computing in engineering design and control*, volume 94, pages 14–27, 1994.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Takehisa Kohira, Hiromasa Kemmotsu, Oyama Akira, and Tomoaki Tatsukawa. Proposal of benchmark problem based on real-world car structure design optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 183–184, 2018.

Rodolphe Le Riche and Victor Picheny. Revisiting bayesian optimization in the light of the coco benchmark. *Structural and Multidisciplinary Optimization*, 64(5):3063–3087, 2021.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

Alexander G de G Matthews, Mark Van Der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas,

Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. Gpflow: A gaussian process library using tensorflow. *J. Mach. Learn. Res.*, 18 (40):1–6, 2017.

Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference: Novosibirsk, July 1–7, 1974*, pages 400–404. Springer, 1975.

Victor Picheny, Robert B Gramacy, Stefan Wild, and Sebastien Le Digabel. Bayesian optimization under mixed constraints with a slack-variable augmented lagrangian. *Advances in neural information processing systems*, 29, 2016.

Victor Picheny, Joel Berkeley, Henry B Moss, Hrvoje Stojic, Uri Granta, Sebastian W Ober, Artem Artemev, Khurram Ghani, Alexander Goodall, Andrei Paleyes, et al. Trieste: Efficiently exploring the depths of black-box functions with tensorflow. *arXiv preprint arXiv:2302.08436*, 2023.

Tony Pourmohamad and Herbert KH Lee. Bayesian optimization via barrier functions. *Journal of Computational and Graphical Statistics*, 31(1):74–83, 2022.

Edward O Pyzer-Knapp, Changwon Suh, Rafael Gómez-Bombarelli, Jorge Aguilera-Iparraguirre, and Alán Aspuru-Guzik. What is high-throughput virtual screening? a perspective from organic materials discovery. *Annual Review of Materials Research*, 45:195–216, 2015.

Matthias Schonlau, William J Welch, and Donald R Jones. Global versus local search in constrained optimization of computer models. *Lecture notes-monograph series*, pages 11–25, 1998.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18, 2005.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

Sattar Vakili, Henry Moss, Artem Artemev, Vincent Dutordoir, and Victor Picheny. Scalable thompson sampling using sparse gaussian process models. *Advances in neural information processing systems*, 34: 5631–5643, 2021.

Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale bayesian optimization in high-dimensional spaces. In *International Conference on Artificial Intelligence and Statistics*, pages 745–754. PMLR, 2018.

Colin White, Willie Neiswanger, and Yash Savani. Bananas: Bayesian optimization with neural architectures for neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10293–10301, 2021.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. Efficiently sampling functions from gaussian process posteriors. In *International Conference on Machine Learning*, pages 10292–10302. PMLR, 2020.

Stephen Wright, Jorge Nocedal, et al. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

In Appendix A we provide additional implementation details for the algorithms used in the paper, and provide full specifications of the test problems in Appendix B.

A Implementation Details

We now discuss the details of the implementation used to produce the results presented in the main body of the paper. For any additional clarification required, the code used to obtain the results is available at: `https://anonymous.4open.science/r/5D85`. Our code used to obtain the results for the `EI-AL` algorithm (Picheny et al., 2016) is available at `https://anonymous.4open.science/r/D0DB`. This uses the code provided in the `laGP R` package (Gramacy, 2016), with some edits to the original code in order to fix several minor bugs we found.

All experiments were performed on a 2018 Macbook Pro, with a 2.3GHz Quad-Core Intel Core i5 processor and 16GB of RAM.

A.1 Gaussian Process Regression

Our implementation is built on top of the `Trieste` (Picheny et al., 2023) library, which leverages `GPFlow` (Matthews et al., 2017) for fitting GPs. We used the Matérn-52 kernel with ARD and a zero mean function, and kept the noise variance fixed at 10^{-6}. The lengthscales and signal variance of the kernel were fitted through optimising the log-marginal likelihood. Lengthscales were initialised at $0.2\sqrt{d}$, with d being the dimensionality of the problem, with a log-normal prior placed on the lengthscales, with mean of $\ln(0.2\sqrt{d})$ and standard deviation 1. Lengthscales were also bounded between $\sqrt{d}/100$ and $\sqrt{d}$, as recommended by Le Riche and Picheny (2021). The signal variance was initialised as the empirical variance of the randomly sampled initial values, with a log-normal prior placed on it with mean of the natural logarithm of the empirical variance of the initial values, and standard deviation 1. For the decoupled sampling (Wilson et al., 2020) used in our implementations of `SCBO` and `TS-AL` we used 1000 random Fourier features for all problems other than the Mazda problem, for which we used 500 random Fourier features.

A.2 Thompson Sampling Augmented Lagrangian (`TS-AL`) Details

We provide the main details of the `TS-AL` algorithm in the main body of the paper, but include additional details here about the initialisation and updates of the Lagrange multipliers and penalty parameters.

The Lagrange multipliers $\boldsymbol{\mu} \in \mathbb{R}^m_{\geq 0}$ and $\boldsymbol{\lambda} \in \mathbb{R}^n$ are initially set to $\mathbf{0}$ as in Gramacy et al. (2016a). The penalty term ρ is initialised in a similar manner to Gramacy et al. (2016b), Picheny et al. (2016), with a minor refinement made in order to handle inequality constraints correctly. Adopting the same notation as Picheny et al. (2016), we use $\mathbf{v}(\mathbf{x})$ to denote a Boolean vector of length $m+n$ recording which constraints are valid at $\mathbf{x}$. In other words, $\mathbf{v}_j(\mathbf{x}) = 1$ if $g_j(\mathbf{x}) \leq 0$ for $j = 1, \ldots, m$ and $\mathbf{v}_j(\mathbf{x}) = 1$ if $|h_j(\mathbf{x})| \leq \epsilon$ for $j = m+1, \ldots, m+n$. Then, given k randomly sampled initial points $\{\mathbf{x}_i\}_{i=1}^k$ our amended expression for initialising ρ is:

$$\rho^0 = \frac{\min_{i=1,\ldots,k}\{\sum_{j=1}^m \max(g_j(\mathbf{x}_i), 0)^2 + \sum_{j=m+1}^{m+n} h_j(\mathbf{x}_i)^2 : \exists j, \mathbf{v}_j(\mathbf{x}_i) = 0\}}{2\min_{i=1,\ldots,k}\{f(\mathbf{x}_i) : \forall j, \mathbf{v}_j(\mathbf{x}_i) = 1\}}$$

In the case that that all k initial samples are valid, ρ^0 is set to 1, and in the case that no initial samples are valid, the denominator is set to the median value of $f(\mathbf{x}_i)$, as in Picheny et al. (2016).

As mentioned in the main body of the paper, whilst the description given for the `TS-AL` algorithm in Algorithm 1 is for batch size $B = 1$, the extension to $B > 1$ is straightforward. At each iteration of the BO loop, instead of drawing a *single* sample from each of the GP surrogate posteriors at each iteration, we instead draw B samples, and use them to form B augmented Lagrangians, which are then minimised in order to select a batch of B points to query. The Lagrange multipliers $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ are updated in exactly the same manner as with batch size $B = 1$. The penalty update is altered slightly for the case $B > 1$. As detailed in step 10 of Algorithm 1, at each step of the BO loop the penalty term ρ is updated. At iteration i, the best point, $\mathbf{x}^*$, which has been previously observed is defined as the one which minimises the augmented Lagrangian, i.e.

$$(\mathbf{x}^*, s^*) = \underset{\mathbf{x}\in\mathcal{D},s}{\operatorname{argmin}}\{\mathcal{L}_A(\mathbf{x}, s; \boldsymbol{\mu}^{i-1}, \boldsymbol{\lambda}^{i-1}, \rho^{i-1})\}$$

If this point is valid, then it suggests that the penalisation of constraint violation, controlled by ρ, is sufficiently high, causing the minimiser of $\tilde{\mathcal{L}}_A(\cdot)$ to lie within the valid region. In this case, ρ is deemed to be sufficiently low, and hence isn't reduced for the next iteration i.e. $\rho^i = \rho^{i-1}$. However, if this point is invalid, then it suggests that ρ isn't yet sufficiently penalising constraint violation, causing the minimiser of $\tilde{\mathcal{L}}_A(\cdot)$ to lie outside the valid region. In this case, ρ is reduced in order to more heavily penalise constraint violation. For batch size $B = 1$, we half ρ, i.e. $\rho^i = \frac{1}{2}\rho^{i-1}$, as was done by Gramacy et al. (2016a). However, for batch size $B > 1$, it makes sense to more aggressively reduce ρ, as the corresponding single batch approach may have reduced ρ by up to a factor of $\frac{1}{2^B}$ over the course of B iterations. However, we did find that reducing ρ too aggressively did make the resulting augmented Lagrangian more difficult to optimise, due to the gradient at the boundary of the feasible region becoming extremely large. Therefore, for batch size B, in step 10 of Algorithm 1, if ρ is deemed to be too large, we use the update rule $\rho^i = 2^{-\min(B,10)} \times \rho^{i-1}$.

A.3 Expected Constrained Improvement (`ECI`) Details

As discussed in the main body of the paper, for `ECI` (Gelbart et al., 2014, Gardner et al., 2014) we used the implementation provided in the `Trieste` library (Picheny et al., 2023). The original `ECI` was only defined for batch size $B = 1$; for batch sizes $B > 1$, we adapted the `BatchExpectedConstrainedImprovement` acquisition function implemented in the `Trieste` documentation [2]. This approximates the `ECI` for batches of points by using Monte Carlo estimation with the reparameterisation trick, originally introduced by Ginsbourger et al. (2010) for *unconstrained* EI. We used 50 samples for the Monte Carlo approximation when using `BatchExpectedConstrainedImprovement`.

A.4 Trust Region Details

We utilise largely the same trust region approach as that introduced by Eriksson and Poloczek (2021), which we describe here for the sake of completeness. For the hyperparameters associated with trust regions discussed in the body of the paper, we set the following values: $\tau_s = \max(3, \lceil d/10 \rceil)$, $\tau_f = \lceil d/B \rceil$, $L_{\min} = 2^{-7}$, $L_{\max} = 1.6$ and $L_{\text{init}} = 0.8$, for problem dimensionality d and batch size B. These values are used after scaling the domain of the problem to the unit hypercube, $[0, 1]^d$.

Each iteration of the BO loop is categorised as a success if it discovers a point which improves on the best point observed so far, or a failure if not. When a success occurs, the success counter n_s is incremented by one and the failure counter n_f is reset to zero. When a failure occurs, the failure counter is incremented by one and the success counter is reset to zero. If $n_s = \tau_s$, the side length of the trust region, L, is set to $\min(2L, L_{\max})$, and n_s and n_f are reset to zero. If $n_f = \tau_f$, then the side length L is set to $L/2$. If $L < L_{\min}$, we reset the side length to L_{init} and reset n_s and n_f to zero. Note that in this case we do not discard all data collected by previous trust regions, as done by Eriksson and Poloczek (2021).

A.5 Acquisition Function Optimisation Details

When performing minimisation of the acquisition functions, they are first evaluated at a set of $\min(500d, 5000)$ random locations (within the trust region). We follow the logic of Eriksson et al. (2019), Eriksson and Poloczek (2021), who found that it is useful not to perturb all dimensions of candidate points at once. Therefore, for each dimension of the randomly generated locations, we select the randomly generated value along that dimension with *perturbation probability* $p_{\text{perturb}} = \min\{1, 20/d\}$, and set it equal to the value of the best point observed so far (i.e. the point at the center of the trust region) otherwise.

For `SCBO`, the best of these randomly sampled locations is chosen to be queried next, in accordance with Eriksson and Poloczek (2021). For `BatchExpectedConstrainedImprovement`, we also adopted the same approach. For `ECI` with batch size $B = 1$ we then ran L-BFGS-B (Liu and Nocedal, 1989) from the best of the initial randomly sampled points in order to minimise the acquisition function further. This approach is also utilised for optimising the `EI-AL` acquisition function in the implementation provided in the `laGP` package used (Gramacy, 2016).

[2] `https://secondmind-labs.github.io/trieste/1.2.0/notebooks/inequality_constraints.html`

As mentioned in the main body of the paper, we found that it was difficult to reliably use L-BFGS-B to optimise the `TS-AL` acquisition function across all test problems, and instead used a modified version of Adam (Kingma and Ba, 2014) to perform minimisation of $\tilde{\mathcal{L}}_A(\cdot)$. We found that the main difficulty facing optimisers when performing minimisation of $\tilde{\mathcal{L}}_A(\cdot)$ was that at the boundary of the feasible region the gradient of $\tilde{\mathcal{L}}_A(\cdot)$ would get extremely large, often making it several orders of magnitude larger than the gradient within the feasible region. Even when using Adam (Kingma and Ba, 2014) this posed a challenge, as it maintains an exponential moving average of the gradient, which could become large if the optimiser ever strayed into the invalid region, and no longer an accurate estimate of the gradient within the valid region if the optimiser moved back into this region in subsequent iterations. This meant that after straying into the invalid region, sometimes the optimiser would actually take steps which resulted in an *increase* in $\tilde{\mathcal{L}}_A(\cdot)$. As such, when running Adam we kept track of each of the points queried by the optimiser over the course of the 150 iterations it was run for, and returned the best of these points when performing minimisation. Input constraints imposed by the trust region are handled naturally by L-BFGS-B; in the case of Adam we added an additional penalty term to the augmented Lagrangian being optimised which multiplied the distance by which a point was *outside* the trust region in each dimension by a heavy penalty term (10^{10}). Finally, Adam was run for 150 iterations, with a learning rate of 0.001 and with gradient-clipping applied when the norm of the gradient exceeded 1000.

B Problem Specifications

Full specifications of the problems used in the paper are given below. Note that some of the functions were re-scaled in order to fit the GPs; we provide the constants used to re-scale below. The domains of all functions were also re-scaled to $[0, 1]^d$ for the purpose of fitting the GPs.

B.1 Lockwood

As described in the body of the paper, the Lockwood problem is a 6D problem with 2 constraints, and was used by Gramacy et al. (2016a). It is based on preventing contaminated groundwater from spreading into the Yellowstone river in Lockwood, Montana, by operating a system of pumps. The objective is to minimise the cost of running the system of pumps, whilst preventing any contaminated water from escaping beyond some specified boundaries. Mathematically, the problem is defined as:

$$\min_{\mathbf{x}\in[0,2\cdot 10^4]^6} f(\mathbf{x}) = \sum_{i=1}^{6} \mathbf{x}_i \quad \text{such that} \quad c_1(\mathbf{x}) \leq 0 \quad \text{and} \quad c_2(\mathbf{x}) \leq 0$$

Evaluation of the objective f and constraint functions c_1, c_2 is done using a simulation, the C++ source code for which is provided by Robert Gramacy on his website [3].

In order to re-scale the objective and constraint functions for facilitating the fitting of the GPs, we divided the objective by 10000 and the constraints by 1000. For this problem, we imposed a budget of 400 evaluations, with 30 initial points sampled randomly using a Halton sequence and a batch size of $B = 1$.

B.2 Ackley 10

We used the constrained 10D Ackley function problem given by Eriksson and Poloczek (2021). In this problem, the goal is to minimise the 10D Ackley function:

$$f(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}\mathbf{x}_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi\mathbf{x}_i)\right) + 20 + \exp(1)$$

subject to:

[3] `https://bobby.gramacy.com/surrogates/`

$$c_1(\mathbf{x}) = \sum_{i=1}^{10} \mathbf{x}_i \leq 0 \quad \text{and} \quad c_2(\mathbf{x}) = ||\mathbf{x}||_2 - 5 \leq 0$$

for $\mathbf{x} \in [-5, 10]^{10}$.

In order to re-scale the objective and constraint functions for facilitating the fitting of the GPs, we divided the objective and constraints by 10. For this problem, we imposed a budget of 200 evaluations, with 10 initial points sampled randomly using a Halton sequence and a batch size of $B = 1$.

B.3 Keane 30

We also used the constrained 30D Keane function problem used by Eriksson and Poloczek (2021) and introduced by Keane (1994). For this problem, the goal is to minimise the Keane bump function:

$$f(\mathbf{x}) = -\left| \frac{\sum_{i=1}^{d} \cos^4(\mathbf{x}_i) - 2\prod_{i=1}^{d} \cos^2(\mathbf{x}_i)}{\sqrt{\sum_{i=1}^{d} i\mathbf{x}_i^2}} \right|$$

subject to:

$$c_1(\mathbf{x}) = 0.75 - \prod_{i=1}^{d} \mathbf{x}_i \leq 0 \quad \text{and} \quad c_2(\mathbf{x}) = \sum_{i=1}^{d} \mathbf{x}_i - 225 \leq 0$$

for $\mathbf{x} \in [0, 10]^{30}$.

In order to re-scale the objective function for facilitating the fitting of the GP surrogate, we multipled it by 10. Due to the highly varied nature of the constraints, we adopted the approach taken by Eriksson and Poloczek (2021) and applied the *bilog* transformation: $\text{bilog}(y) = \text{sign}(y)\ln(1 + |y|)$. For facilitating the fitting of the GP surrogates, we then divided $\text{bilog}(c_1(\mathbf{x}))$ by 10, leaving $\text{bilog}(c_2(\mathbf{x}))$ unaltered. We imposed a budget of 2000 evaluations, with 100 initial points sampled randomly using a Halton sequence, and a batch size of $B = 1$.

B.4 Mazda

The Mazda three vehicle benchmark used was introduced by Kohira et al. (2018). This is a 222D problem, requiring the tuning of the thickness of 222 parts across 3 vehicles whilst satisfying 54 constraints. The original problem was multi-objective, with the objectives being to minimise the total weight of the 3 vehicles whilst maximising the number of parts shared between them. As we were focused on the single-objective case, we solely focused on minimising the total weight of the 3 vehicles whilst satisfying all 54 constraints.

The objective and constraint functions were evaluated using the `C++` simulation introduced by Kohira et al. (2018), which is available online [4]. We didn't scale the objective or constraints before fitting the GPs. We imposed a budget of 2500 evaluations, with 300 initial points sampled randomly using a Halton sequence, and a batch size of $B = 100$.

[4] `https://ladse.eng.isas.jaxa.jp/benchmark/`