Faculty of science

Spatial Statistics with Image Analysis, MASM25

Gaussian Processes for Bayesian Optimization (Project 3)

Authors:
Mohamed Abrash,
Maiya Tebäck

December, 2024

Contents

1 Introduction

Optimization is one of the cornerstones of modern applied mathematics, owing to the widespread need to find the minimum or maximum of functions in various fields. This process often leads to the optimal solution of a task or reduced costs in practical applications. Traditionally, gradient-based algorithms, particularly variants of gradient descent, have been the standard approach to solving optimization problems. While effective, these methods have notable limitations, such as the tendency to get stuck in local minima. To address these shortcomings, many advanced techniques have been proposed, including the incorporation of momentum, adaptive step sizes (e.g., the *Adam* [1] algorithm), and the use of higher-order derivatives.

For problems where gradient information is unavailable or unreliable, gradient-free optimization techniques are employed. These methods only use function evaluations and include algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Simulated Annealing (SA). While these approaches can handle a broader range of problems, they often require more computational effort.

In this project, we explore an alternative approach known as Bayesian Optimization. This method models the objective function using a Gaussian Process (GP) and is particularly suited for optimizing expensive-to-evaluate functions with unknown gradients. The core idea is to construct a surrogate model — the Gaussian Process — that guides the search for the minimum. During the optimization process, the algorithm selects new points to evaluate based on the fitted model and its confidence bounds. This decision-making process relies on an acquisition function, which balances exploration (uncertainty reduction) and exploitation (minimizing the objective function).

Unlike gradient-based algorithms, which take sequential steps dictated by gradients, Bayesian Optimization evaluates points anywhere within the support of the objective function. This flexibility is a key advantage, enabling efficient optimization of complex, non-convex landscapes.

In this project, we outline the fundamental principles of Bayesian Optimization, exploring various acquisition functions and illustrating their differences. Specifically, we examine the Lower Confidence Bound (LCB), Expected Improvement (EI), and Probability of Improvement (PI). After a theoretical overview, we present examples and case studies that demonstrate the method's application. These include the use of different covariance functions, such as the exponential, Matérn, and Gaussian kernels; addressing anisotropy; comparing acquisition functions; and balancing interpolation and exploration. Finally, we showcase results for several test functions in one and two dimensions to illustrate the effectiveness of the approach, and even venture a little into higher dimensions to discuss the new and exacerbated issues encountered there.

2 Theory

2.1 Objectives and Basic Assumptions

Bayesian Optimization aims to locate the global minimum of an objective function f. For simplicity, we assume that the domain of f is a hypercube, although this assumption can be relaxed in more complex treatments. The objective function is considered expensive to evaluate and lacks a specific known structure, classifying it as a *black-box* function. Despite its generality, the function is assumed to exhibit continuity and sufficient regularity to be effectively modelled by a Gaussian Process. Furthermore, this project assumes that gradient information is unavailable, although advanced methods may leverage gradient data to enhance efficiency [2].

2.2 Bayesian Optimization Workflow

Here we provide a high-level overview of the Bayesian Optimization approach. This should aid the reader in understanding subsequent theory sections.

Bayesian Optimization Pseudo-Code

Input: Function f, number of initial evaluations n_0, maximum number of allowed evaluations N.
Output: Estimated global minimum of f and a set of evaluations τ.

1. **Initialize**:
   - Select a Gaussian Process and covariance function as prior for the objective function f. See section 2.3.
   - Evaluate f at an initial set of n_0 points: $\tau = \{x_i, y_i\}_{i=1}^{n_0}$
2. **Iterate**: (while $n \leq N$)
   - Re-estimate model parameters of f by finding the **maximum likelihood estimate** using all available points τ (find the parameters that maximize (11) in section 2.4).
   - Use the current model (posterior) to choose the next evaluation point, x_*, by maximizing/minimizing the **acquisition** function appropriately. Details about possible functions are found in section 2.5.
   - Evaluate the objective function at x_* to obtain $y_* = f(x_*)$ and append $\{x_*, y_*\}$ to τ.
   - Increment $n \leftarrow n + 1$.
3. **Return**: The point which results in the lowest evaluation of the function. I.e. the x_i corresponding to $\min\{y_j\}_{j=1}^{n_0+N}$.

2.2.1 Remarks

Assessing the convergence of this algorithm can be challenging, as it depends on several factors, including the number of iterations, the properties of the objective function, the choice of the acquisition function, and the number of initial evaluations.

The algorithm described above involves two internal optimization steps: One for obtaining the maximum likelihood estimate and another for optimizing the acquisition function. In our implementation, both optimizations are performed using the Nelder-Mead simplex algorithm as implemented in Matlab-R2024b.

As a consequence of using the Nelder-Mead algorithm, a small technicality arises. With a bad initial guess and non-convex objective function, the Nelder-Mead algorithm may converge to a local minimum. When optimizing the acquisition function, this may return a non-optimal evaluation proposal which can negatively affect the performance of the Bayesian optimization.

To address this, we adopt a two-step approach for the optimization of the acquisition function. First, a grid search is conducted over the domain to identify a promising starting point. Then, the grid search result is refined using the Nelder-Mead optimizer, starting from the identified point. It is important to note that this approach is only doable for low-dimension problems because the grid search scales exponentially with dimension.

2.3 Gaussian Process and Covariances

For an unknown function $f(\mathbf{x})$, where $\mathbf{x}$ in general is an n-dimensional vector, we will assume a Gaussian process, such that

$$y_i|f_i \in \mathcal{N}(f_i, \sigma_\varepsilon^2), \tag{1}$$

where f_i is the true function value at point x_i, and y_i is the value which we observe. If the evaluation is noise-free, then the variance σ_ε^2 may be taken to be zero, however, small values for the variance are commonly used to improve numerical stability, for example, $\approx 10^{-6}$. Note that if we are dealing with a stochastic function, this variance would need to be estimated. We further assume that the full vector of true function values at the current points is a zero-mean Gaussian random field, i.e.

$$\mathbf{f} \in \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \tag{2}$$

with $\Sigma_{ij} = r(\mathbf{x}_i - \mathbf{x}_j)$ being the covariance on the $\mathbf{x}$-field. For a function on domains of more than one dimension, we may expect the function to behave differently in different directions. This might indicate the need for an anisotropic covariance function. A simple rescaling of the distance metric can provide an easy implementation of axis anisotropy in the covariance function which adds additional parameters to be estimated in our fit, namely $\boldsymbol{\rho}$:

$$d(\mathbf{x}, \widehat{\mathbf{x}}) = \sqrt{\sum_i \left(\frac{x_i - \widehat{x}_i}{\rho_i} \right)^2}, \tag{3}$$

where the weights in $\boldsymbol{\rho}$ can be different for each coordinate. Meanwhile, for the isotropic case, the anisotropy parameters ρ_i are all equal and are estimated as a single scalar.
For the covariance function itself, a flexible choice is the **Matérn covariance**, which for $\nu = k+1/2,\ k \in \mathbb{Z}_+$ has simple expressions, e.g.

$$r_{1/2}[d(\mathbf{x}, \widehat{\mathbf{x}})] = \sigma^2 \exp(-d), \tag{4}$$

$$r_{3/2}[d(\mathbf{x}, \widehat{\mathbf{x}})] = \sigma^2 \big(1 + \sqrt{3}\big) \exp\big(-\sqrt{3}d\big), \tag{5}$$

$$r_{5/2}[d(\mathbf{x}, \widehat{\mathbf{x}})] = \sigma^2 \Big(1 + \sqrt{5} + \frac{5d^2}{3}\Big) \exp\big(-\sqrt{5}d\big). \tag{6}$$

The ν parameter controls the smoothness of the process. Note that in the function expressions we have used $d = d(\mathbf{x}, \widehat{\mathbf{x}})$ as defined above in (3). We have also let the usually occurring parameter $\kappa = 1$, as including it in the model would simply lead to a rescaling of the $\boldsymbol{\rho}$ vector.
For the particular choice of $\nu = 0.5$, the Matern covariance is also known as the **Exponential covariance** function due to its functional form. We will also be using the **Gaussian covariance** function, which can be seen as the Matérn where we have let $\nu \to +\infty$, i.e.

$$r_{+\infty}[d(\mathbf{x}, \widehat{\mathbf{x}})] = \sigma^2 \exp(-2d^2). \tag{7}$$

2.4 Parameter Estimation & Reconstructions

Given observations $\mathbf{y}$ and a Gaussian model with parameters collected in $\theta = \{\boldsymbol{\rho}, \nu, \sigma^2\}$, the posterior distribution is given by

$$P(\theta|\mathbf{y}) \propto P(\mathbf{y}|\theta) P(\theta) \tag{8}$$

with the likelihood given by a Gaussian probability density. For simplicity, we take flat priors on the model parameters θ and take as our estimate the maximum a posteriori estimate (MAP) which when combined with flat priors is equivalent to the maximum likelihood estimate. Furthermore, taking the log of the likelihood gives

$$\hat{\theta} = \arg\max_\theta \log \left[\frac{1}{|\mathbf{\Sigma}_\theta|^{1/2}} \exp(-\frac{1}{2} \mathbf{y}^T \mathbf{\Sigma}_\theta^{-1} \mathbf{y}) \right] \tag{9}$$

$$= \arg\max_\theta \left(-\frac{1}{2} \ln |\Sigma_\theta| - \frac{1}{2} \mathbf{y}^T \mathbf{\Sigma}_\theta^{-1} \mathbf{y} \right) \tag{10}$$

where $\mathbf{\Sigma}_\theta$ is the Matérn covariance matrix evaluated at the points $\mathbf{x}$ corresponding to $\mathbf{y}$. Taking the Cholesky factor R_θ of the covariance matrix, the maximum likelihood estimate is thus

$$\hat{\theta} = \arg\max_\theta \left(-\sum_i |R_{\theta,ii}| - \frac{1}{2} ||\mathbf{y}^T R^{-1}||_2^2 \right). \tag{11}$$

For more details on the maximum likelihood estimate of Gaussian processes consult [3].
Once an estimate is established and given all known points, a reconstruction — Kriging — can be performed on any set of points in the domain. This reconstruction is given as the conditional expectation of new unknown values y_u at x_u given the known evaluations $\mathbf{y}_k$ at $\mathbf{x}_k$.

$$\hat{y}_u = \mathbb{E}[y_u|\mathbf{y}_k, \hat{\theta}] = \mathbf{\Sigma}_{uk}\mathbf{\Sigma}_{kk}^{-1}\mathbf{y}_k \tag{12}$$

$$\sigma_u^2 = \mathbb{V}[y_u|\mathbf{y}_k, \hat{\theta}] = \sigma_\epsilon^2 + \hat{\sigma}^2 - \mathbf{\Sigma}_{uk}\mathbf{\Sigma}_{kk}^{-1}\mathbf{\Sigma}_{uk}. \tag{13}$$

Σ_{kk}, Σ_{uk} are the cross-covariance matrices between the known locations and between the unknown and the known locations respectively. An important assumption of Bayesian Optimization is that this reconstruction is much less computationally costly than the objective function. The acquisition function combines the information provided by this reconstruction to create a criterion for selecting the next evaluation point. [4]

2.5 Acquisition function

Acquisition functions play a central role in Bayesian Optimization by guiding the selection of subsequent evaluation points. To understand the necessity of acquisition functions consider the following thought experiment. After evaluating a few points, the Gaussian Process (GP) surrogate model exhibits minimal predictive uncertainty across the domain. In this idealized case, the optimization procedure could terminate, with the global minimum of the GP posterior serving as the solution. However, in realistic settings, the surrogate model's confidence intervals often vary significantly across the domain, potentially concealing the true global minimum. As a result, selecting evaluation points requires a more nuanced strategy that accounts for both the predicted value of the objective function and the associated uncertainties.

Acquisition functions provide a principled criterion for this selection process, balancing exploration (reducing uncertainty by sampling regions with high predictive variance) and exploitation (focusing on regions likely to contain the global minimum). This balance is critical for efficient optimization, especially when function evaluations are expensive.

To compute an acquisition function, the parameters of the GP model (σ and $\boldsymbol{\rho}$) are estimated via the maximum likelihood approach, as detailed in Section 2.4. The resulting posterior distribution allows for the computation of conditional expectations and variances, which serve as inputs to the acquisition function. Below we list a few of the common acquisition functions. Other acquisitions can be found in [2].

2.5.1 Lower Confidence Bound (LCB)

The LCB acquisition function penalizes points with high uncertainty, effectively encouraging exploration. It is defined as:

$$a(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})|\mathbf{y}] - \beta\mathbb{D}[f(\mathbf{x})|\mathbf{y}], \tag{14}$$

where $\beta > 0$ controls the trade-off between exploration and exploitation. A typical choice is $\beta = 3$. By minimizing the LCB we obtain the next point for evaluation x_*.

$$x_* = \arg\min_x a(x) \tag{15}$$

2.5.2 Expected Improvement (EI)

Given evaluated points $\{x_i, f_i\}_{i=1}^n$, we define the best estimate $f_{min}(n) = \min\{f_i\}_{i=1}^n$. If we evaluate yet another point $f(x)$, the improvement to our estimate is given by

$$[f_{min} - f(x)]_+ \tag{16}$$

Ideally, we want to pick the point maximizing this expression. However, we have not yet evaluated $f(x)$, and it is unknown. We thus take the expectation with respect to the posterior distribution:

$$a(\mathbf{x}) = \mathbb{E}\Big\{ \big[f_{min} - f(\mathbf{x}), 0\big]_+ |\mathbf{y}\Big\} \tag{17}$$

This expression is called the expected improvement acquisition function. Given that the posterior is Gaussian with mean and variance $\mu_n(x), \sigma_n^2(x)$, it is possible to derive a closed-form expression for the expected improvement as we will show below:

$$a(x) = \int \left[f_{min} - \mu_n(x) - \sigma_n(x)\xi\right]_+ \phi(\xi)\, d\xi \tag{18}$$

where $\phi(\xi)$ is the density function for the standard normal.
Define $m = [f_{min} - \mu(x)]/\sigma(x)$. Then, we get

$$a(x) = \sigma(x) \int_{\mathbb{R}} [m - \xi]_+ \phi(\xi)\, d\xi \tag{19}$$

$$\frac{a(x)}{\sigma(x)} = \int_{-\infty}^{m} (m - \xi)\phi(\xi)\, d\xi \tag{20}$$

$$= m \int_{-\infty}^{m} \phi(\xi)\, d\xi - \int_{-\infty}^{m} \xi\phi(\xi)\, d\xi. \tag{21}$$

Integrating by parts, we reveal the final expression

$$a(x) = \sigma(x) m \Phi(m) + \sigma(x)\phi(m) \tag{22}$$

$$= [f_{min} - \mu(x)]\Phi\left[\frac{f_{min} - \mu(x)}{\sigma(x)}\right] + \sigma(x)\phi\left[\frac{f_{min} - \mu(x)}{\sigma(x)}\right]. \tag{23}$$

Finally, including an additional exploration parameter β we obtain:

$$a(x) = [f_{min} - \mu(x) - \beta]\Phi\left[\frac{f_{min} - \mu(x) - \beta}{\sigma(x)}\right] + \sigma(x)\phi\left[\frac{f_{min} - \mu(x) - \beta}{\sigma(x)}\right]. \tag{24}$$

2.5.3 Probability of Improvement (PI)

Another approach is to consider the posterior predictive distribution and pick the point maximizing the chance of obtaining a lower point. That is, to maximize

$$a(\mathbf{x}) = \mathbb{P}[f(\mathbf{x}) < f_{min} | \mathbf{y}]. \tag{25}$$

Since the posterior is Gaussian with mean and variance $\mu(x), \sigma^2(x)$, the PI can be evaluated by

$$a(x) = \Phi\left[\frac{f_{\min} - \mu(x) - \beta}{\sigma^2(x)}\right] \tag{26}$$

where Φ is a standard normal accumulative distribution function and a parameter β is also included to balance exploitation and exploitation.

3 Case Studies and Analysis

In this section, we present our implementation of the Bayesian optimization algorithm and explore key aspects of this method. Each case study involves selecting a suitable test function, defining relevant parameters, and focusing on a specific aspect of the algorithm. This is followed by optimization according to the workflow described in Section 2.2.

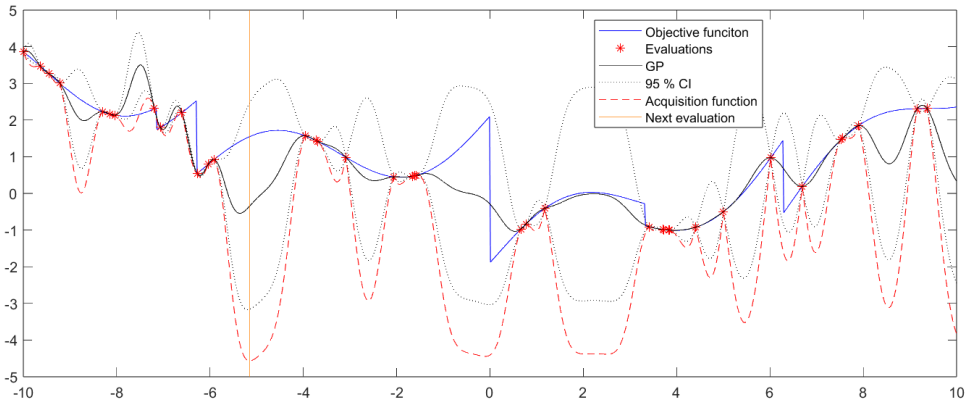
The first case study illustrates the impact of choosing different covariance functions on the optimization process. Next, we compare the performance of various acquisition functions. Subsequently, we examine the influence of the exploration parameter on the optimizer's behaviour. We also highlight the importance of accounting for axis anisotropy. Then we examine a particular test function having multiple global minima and we also study cases in higher dimensions.

We additionally provide visualizations of the true function, the Kriging approximation, and their standard deviations for one- and two-dimensional functions.

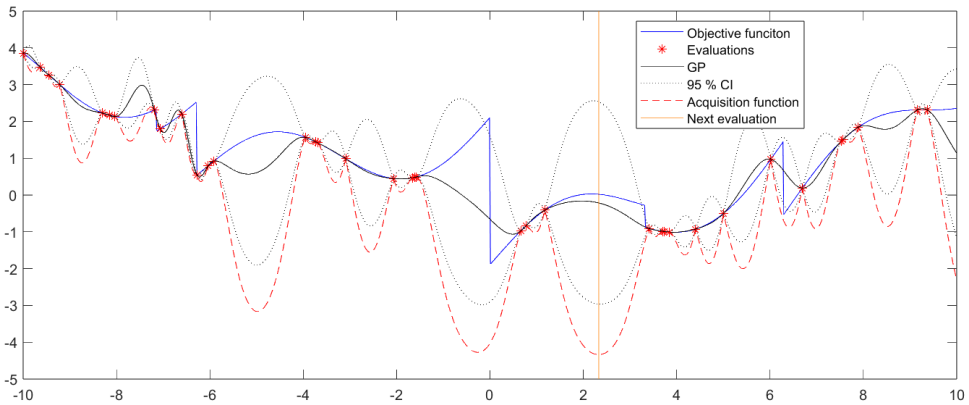
3.1 Covariance Functions

In Bayesian optimization, the covariance function encodes the algorithm's belief about how rapidly the objective function can change. For example, the Gaussian kernel is well-suited for optimizing smooth functions, while the Matérn covariance function can accommodate varying degrees of smoothness based on the parameter ν. Although a perfect fit is not required for the algorithm to locate the global minimum, a well-chosen covariance function can significantly improve efficiency, reducing the number of observations needed for convergence, and enhancing robustness against local minima.

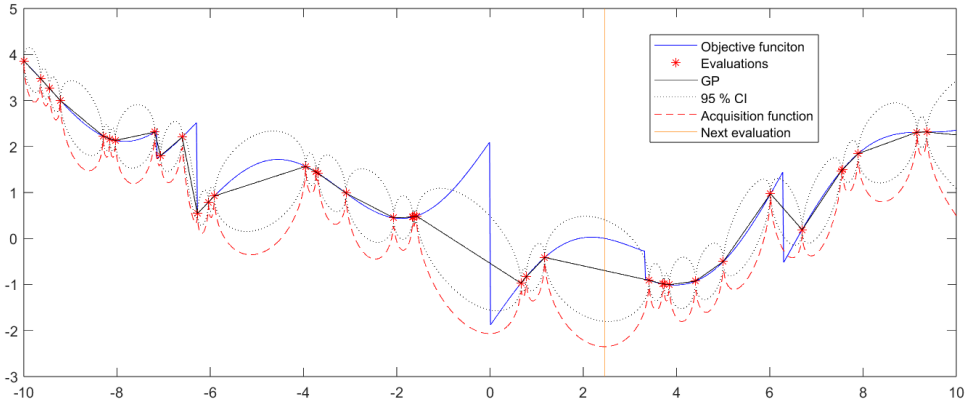
To demonstrate the importance of selecting an appropriate covariance function, we use a test function with a discontinuous derivative and fit different Gaussian processes to 40 randomly selected points within the domain. The fitted covariance functions include the Gaussian kernel, the exponential covariance function, and the Matérn covariance function with $\nu = 2.5$. The evaluation noise variance is set to $\sigma_\epsilon^2 = 10^{-6}$. Additionally, we plot the Lower Confidence Bound (LCB) acquisition function with $\beta = 3$ to illustrate how the choice of covariance function influences the selection of the next evaluation point.



((a)) Gaussian covariance.



((b)) Matérn covariance $\nu = 2.5$



((c)) Exponential covariance (Matérn $\nu = 0.5$)

Figure 1: A Gaussian process is fit to 40 random observations from a test function. The figure shows variable goodness of fit based on the chosen covariance function. Confidence intervals are also shown, as well as an LCB acquisition function in red. The figure illustrates how a better fit results in improved efficiency and choice of subsequent evaluation points.

The test function depicted in Figure 1 is more accurately modelled by a Gaussian process with an exponential covariance compared to the Matérn or Gaussian covariance. The exponential covariance achieves a more faithful reconstruction of the function, accompanied by narrower confidence bounds using the given sample of observations.

The impact of this choice on optimization becomes evident when examining the acquisition functions. For the Gaussian covariance, the acquisition function exhibits multiple regions across the domain where it dips below the true minimum. This behaviour can cause the algorithm to spend additional evaluations in these areas before successfully locating the global minimum. In contrast, with the exponential covariance, the acquisition function dips below the true and reconstructed minima only within a small neighbourhood (approximately from -2 to 4). Because the Gaussian process with exponential covariance better captures the objective function, the algorithm is able to identify the global minimum more efficiently, requiring fewer evaluations and improving the accuracy of the estimate. The Matérn covariance with $\nu = 2.5$ represents an intermediate case, balancing the behaviour observed with the Gaussian and exponential covariances.

Conversely, for smooth functions, the exponential covariance is likely to underperform compared to the Matérn or Gaussian covariances. This observation highlights another critical aspect of Bayesian optimization: While the algorithm is designed to optimize black-box functions, incorporating additional assumptions or prior knowledge about the smoothness of the objective function can significantly enhance performance and efficiency.

3.2 Acquisition Functions

The acquisition function plays a strategic role in Bayesian optimization, guiding the algorithm's search for the global minimum. In this section, we demonstrate how acquisition functions, as introduced in Section 2.5, operate using two test functions in two dimensions: The Goldstein–Price test function and the Ackley test function (Figure 12).

For both test functions, we fix the covariance function to be a Matérn kernel with $\nu = 1.5$. We apply three acquisition functions — Lower Confidence Bound (LCB), Expected Improvement (EI), and Probability of Improvement (PI) — and illustrate the optimization results after 70 iterations in Figures 2 and 4. In these figures:

- The top panels display the reconstructions of the objective function after all iterations, highlighting the randomly-selected initial points (small white dots) and the estimated global minimum (large white dot).
- The bottom panels show the uncertainty in the Gaussian Process (GP) reconstruction and all evaluation points.

Additionally, we plot the improvement paths for each test function. These include:

- The improvement, defined as $I(k) = f_{\min}(k) - f_{\text{true min}}$, at each iteration k.
- The distance to the true minimum in the domain space, $|x_{\min} - \hat{x}_{\min}(k)|$, at each iteration.

These metrics are visualized in Figures 3 and 5, providing a comprehensive view of the optimization process and its convergence.

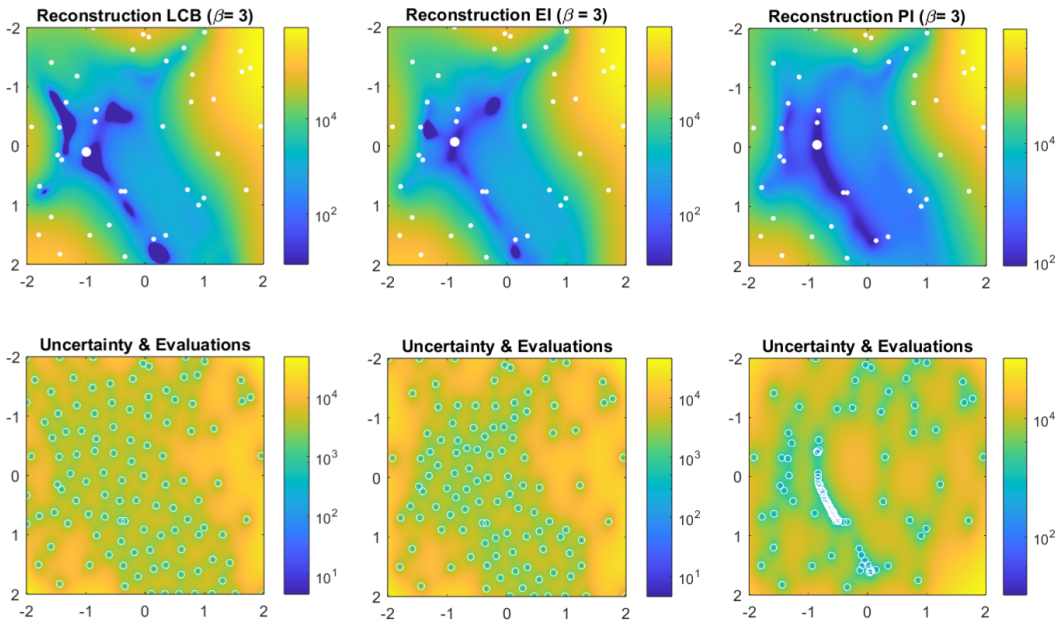


Figure 2: Optimization state of the Goldstein-Price function after 70 iterations. The LCB, EI and PI acquisition functions are used to guide the Bayesian Optimization starting with 40 initial points shown as small white dots in the upper images. The upper images show the Gaussian Process reconstruction using all available points while the lower images show the reconstruction uncertainties over the optimization domain as well as the evaluations. The large white dots in the upper images indicate the location of the estimated minimum.

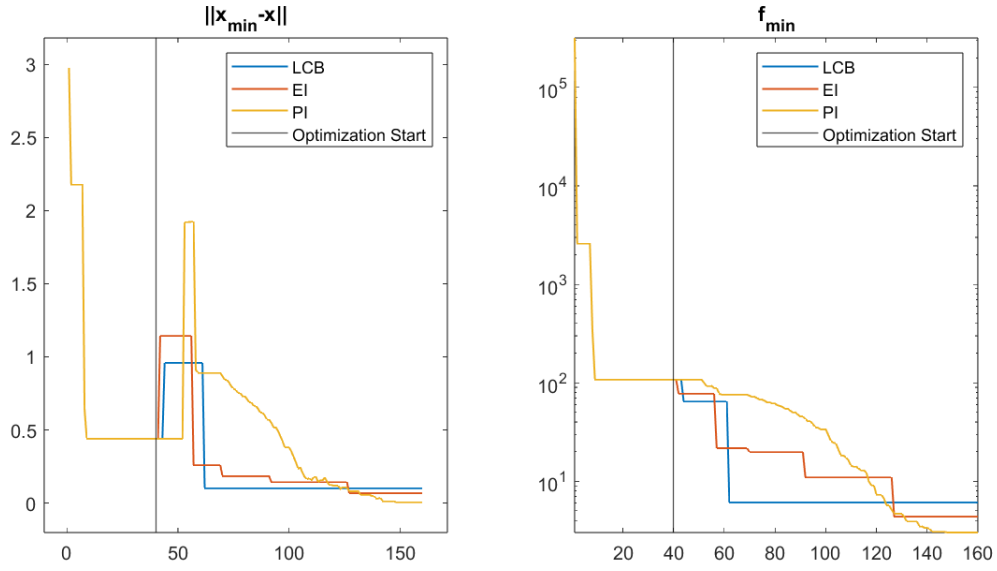


Figure 3: Improvement path showing how far the best estimates at each step are from the true global minimum. The figure to the left shows the distances in the domain of the test function while the figure to the right shows the lowest evaluation point and hence estimate of the global minimum at each iteration (true global minimum = 0 for the Goldstein test function). The plot compares three different acquisitions that are used to select evaluation points, these are LBC, EI and PI.

The Goldstein–Price function is a standard test function with multiple local minima. However, with 40 randomly chosen starting points, the three tested algorithms shown in figure 2 are able to locate the global minimum. The acquisition functions chosen in this case result in different approaches. The LCB

and the EI tend to have more spread-out evaluations, effectively exploring large areas of the domain, with the EI being more focused on the neighbourhood of the Goldstein's crease than the LCB. The PI, on the other hand, takes a local approach. It starts from the lowest point among the initiation set and moves locally toward the minimum, maximizing the probability of improvement.

The differences between the algorithms are clearly seen in the improvement path shown in Figure 3. Because of their more global and explorative approach, algorithms equipped with the LCB and the EI acquisitions locate the global minimum at a faster rate than the slower improvement of the PI. However, we also observe that because these approaches prioritize exploration, there are flat plates in the improvement curve during exploratory iterations for the LCB and EI. Meanwhile, the algorithm equipped with the PI catches up in the long run and achieves better estimates of the global minimum with the same number of iterations.

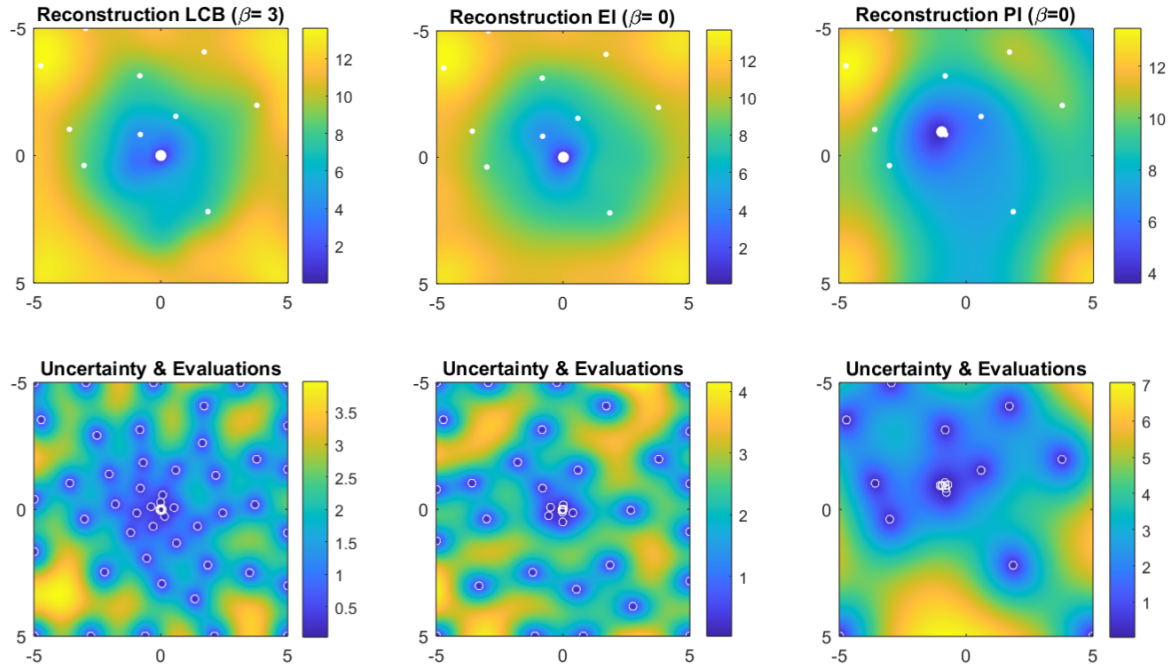


Figure 4: Optimization state of the Ackley function after 70 iterations. The LCB, EI and PI acquisition functions are used to guide the Bayesian Optimization starting with 10 initial points shown as small white dots in the upper images. The upper images show the Gaussian Process reconstruction using all available points while the lower images show the reconstruction uncertainties over the optimization domain as well as the evaluations. The large white dots in the upper images indicate the location of the estimated minimum.

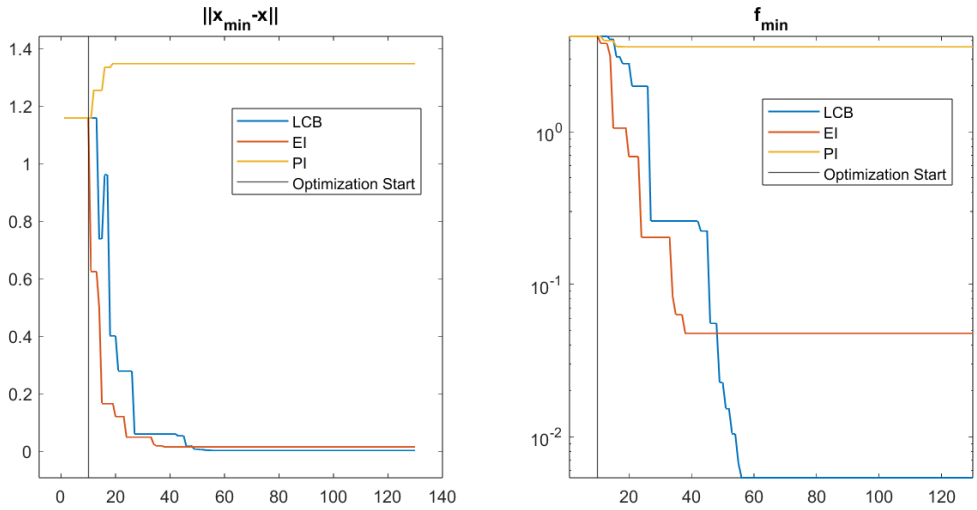


Figure 5: Improvement path showing how far the best estimates at each step are from the true global minimum. The figure to the left shows the distances in the domain of the test function while the figure to the right shows the lowest evaluation point and hence estimate of the global minimum at each iteration (true global minimum = 0 for the Ackley test function). The plot compares three different acquisitions that are used to select evaluation points, these are LBC, EI and PI.

The first example illustrated a case where the exploitative approach of the PI resulted in a better estimate of the global minimum. In the second example, we will show the converse situation and choose a test function with a multitude of local minima, making an explorative approach preferable. To do this, we used the Ackley test function with only 10 initial evaluations and default values for the acquisition parameters $\beta = 3, 0, 0$ for the LCB, EI, and PI, respectively. The results of these optimizations are shown in Figures 4 and 5. In this case, it is clear that the algorithm equipped with the LCB has an advantage over the other approaches as it is better able to navigate the local minima of the Ackley function. It is possible that the PI algorithm can escape the local minima with more iterations, but it is at least clear that it is unlikely to compete with the other algorithms given the multiple other local minima it needs to overcome.

The two examples shown above are intentionally chosen to be extreme opposites to illustrate the differences between the acquisition functions. From these examples, we conclude that an explorative acquisition function is better able to navigate a complex functional form with multiple local minima, while an exploitative acquisition is better suited for refining our estimates once we have found an approximate location of the optimum point. In the next section, we further explore the balance between exploration and exploitation and how they are affected by the β parameter.

3.3 Exploitation-Exploration Balance

The choice of the exploitation-exploration parameter β seen in equations (14),(24) and (25), has an important effect on efficiency and behavior, changing the algorithm's tendencies to be more exploitative or more explorative. In this section, we provide a simple illustration of this fact. We will perform our analysis using the Probability of Improvement acquisition function, but the discussion also applies to the other acquisition functions.

Figure 6 shows the PI plotted as a function of the improvement $(f_{min} - \mu)$ and standard deviation σ. In the subplot to the left where $\beta = 0$, the acquisition function prioritizes points with lower means and high certainty i.e the acquisition priorities points resulting in the most certain improvement. This results in an algorithm that typically "crawls" locally to the closest minimum in a gradient-descent-like approach. This can be an ideal choice in convex problems but can result in faulty convergence to a local minimum as we illustrate in figure 7-a. On the other hand, as we increase β, a more explorative and global approach is seen. In figure 6 with $\beta = 5$, we see that the acquisition function in fact assigns lower weight to points with lower reconstruction than the current minimum even when they have small uncertainty. On the

contrary, it assigns overall higher weights to points with high uncertainty. This results in a more global and explorative approach as can be seen in figure 7-b.

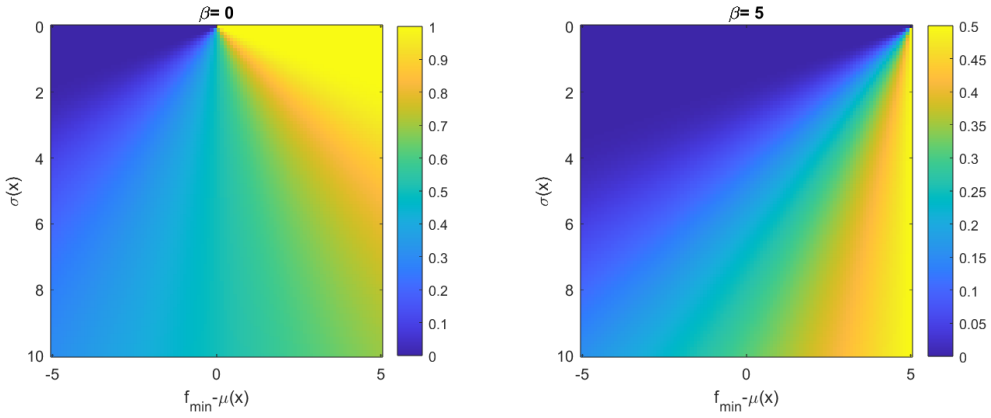
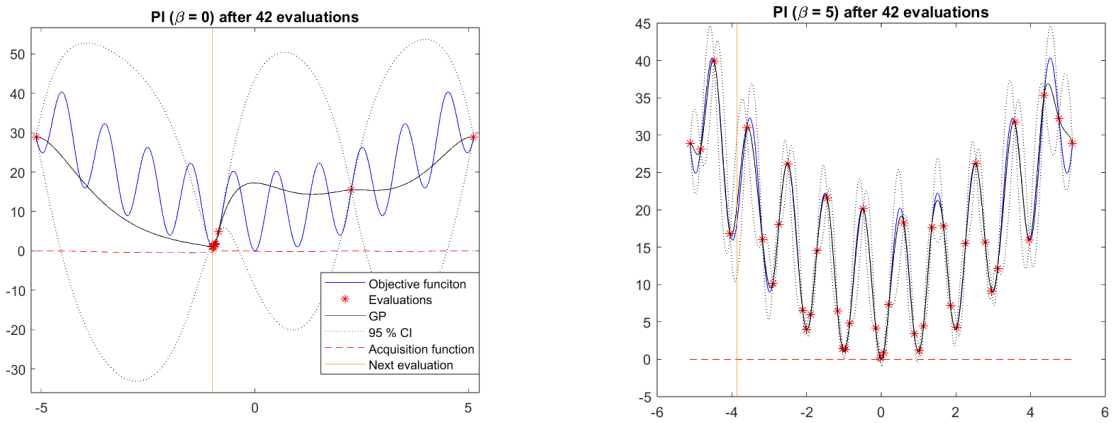


Figure 6: Effect of the acquisition parameter β on the balance between exploitation and exploration in the algorithm. The Probability of Improvement is plotted as a function of mean values and standard deviations. The left plot shows acquisition values for $\beta = 0$, leading to a more exploitative behavior, while the right plot corresponds to $\beta = 5$, encouraging more exploration. On the x-axis, $f_{min} - \mu(x)$ refers to the distance between the current global minimum and the reconstruction means while $\sigma(x)$ on the y-axis refers to the reconstruction standard deviation.



(a) Probability of Improvement $\beta = 0$

(b) Probability of Improvement $\beta = 5$

Figure 7: Shows Bayesian optimization results on the Rastrigin test function. The used covariance function is a Matérn with $\nu = 1.5$ equipped with a PI acquisition function. In Figure (a) the exploration parameter β is set to 0 giving an exploitative optimization while in (b) $\beta = 5$ giving an explorative optimization. x-axis, $f_{min} - \mu(x)$ refers to the distance between the current global minimum and the reconstruction means while $\sigma(x)$ on the y-axis refers to the reconstruction standard deviation.

Figure 7 reveals an undesirable aspect of the acquisition parameter β. This simple illustration indicates that this parameter should be tuned for the particular target function. Big values of β can result in many unnecessary explorative evaluations while a small value results in a locally greedy approach and can result in slow convergence and defaulting to local minima. This aspect is undesirable for the reason that Bayesian optimization aims to find the global minimum of a *black-box function* where such parameter tuning is not possible.

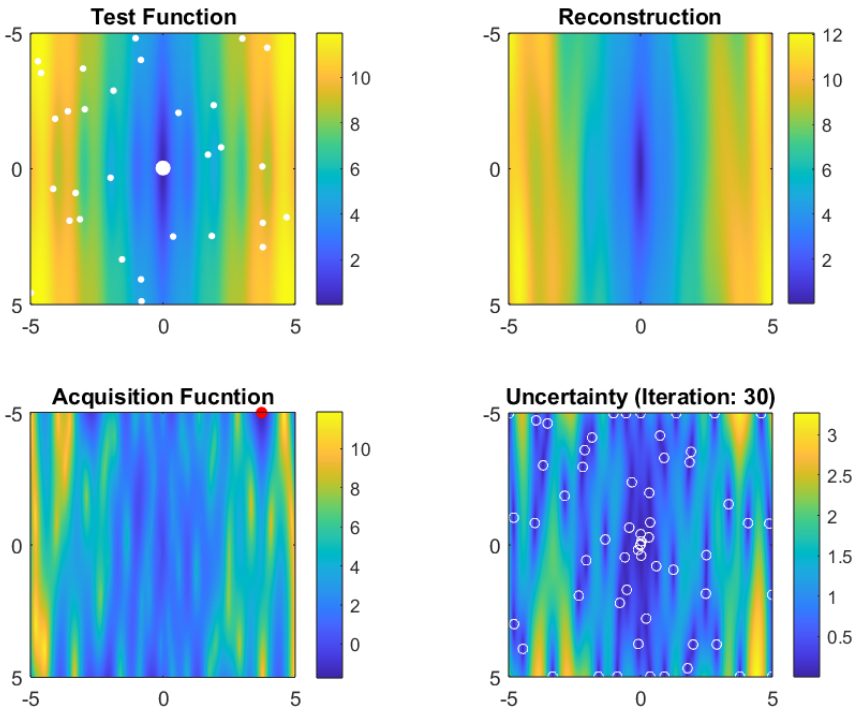
A few different solutions can be proposed for the resolution of such challenges. The first is to perform

batch optimization by first performing an optimization with high β resulting in explorative evaluations and then to perform a second batch where β is small. This method can result in improved efficiency as the first batch serves to locate interesting areas in the domain while the second batch would result in finding a more precise estimate of the minimum's location. Generalizing this approach we may let β be a function of the number of iterations; $\beta(k)$. One example would be to let the parameter be a sigmoid function with appropriate scales $\beta(k) = \frac{5}{\exp(a\frac{k}{N} - a/2)+1}$, where N is the total number of evaluations and a is a scale parameter. Another proposal is to let this parameter adapt dynamically based on the improvement where an improvement less than a given tolerance would increase β.

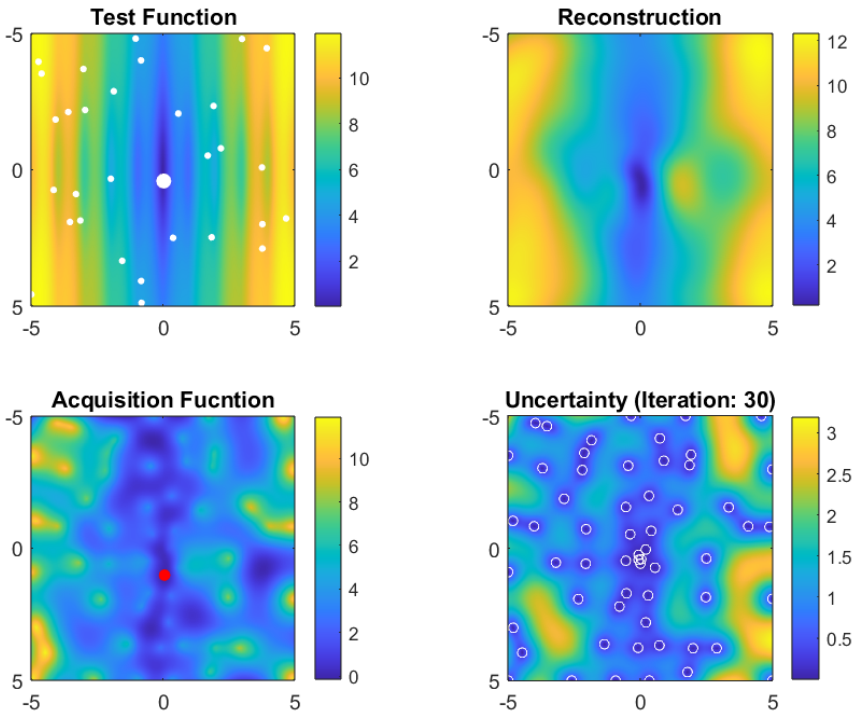
3.4 Axis-anisotropy

In equation (3), we included separate $\boldsymbol{\rho}$ parameters for each of the coordinates which allows the covariance function to have different ranges in different axial directions. In figure 8(a), we investigate the influence these additional parameters have on the Bayesian optimizer by attempting to optimize the axially stretched Ackley function in 2D. Taking 30 initial points and using an LCB with $\beta = 3$ and a Matérn covariance, we allow the optimizer 30 iterations. We test one time when we fit an isotropic Matérn process and one time with axis-anisotropic Matérn.

The true global minimum is found at the origin. We see that when we allow for axis-anisotropy in our Gaussian process, as in sub-figure 8(a), the reconstruction is more faithful to the objective function while if we assume isotropy, as in sub-figure 8(b), then the fit is less optimal. This is also reflected in the estimated minimum. After 30 iterations the algorithm returns the estimate (0.4137, 0.0334) for the isotropic case and (-0.0274, 0.0069) for the anisotropic case, which is an order of magnitude closer to the true minimum when measured with the Euclidean norm.



((a)) Anisotropic Covariance.



((b)) Isotropic Covariance

Figure 8: (a) and (b) illustrate the state of the Bayesian optimizer after 30 iterations. The top figure corresponds to a fit using an anisotropic Matern covariance function with $\nu = 2.5$, while the bottom figure shows the isotropic case. Each figure contains four subplots: (1) the test function along with the initial evaluations and the final estimated minimum (top left), (2) the reconstruction of the function (top right), (3) the acquisition function with a red dot indicating the next evaluation point (bottom left), and (4) the evaluated points along with the reconstruction standard deviations (bottom right). The test function used in this example is the Ackley function, with the x-coordinate scaled by 0.1.

3.5 Multiple Minima

In this section, we will consider a test function with multiple global minima, namely Himmelblau's function (see [5]), which takes the form

$$(x^2 + y - 11)^2 + (x + y^2 - 7)^2 \tag{27}$$

on the domain $(-5, 5) \times (-5, 5)$. It has four global minima, all equal to 0, at the following locations: $(3, 2)$, $(-2.805118, 3.131312)$, $(-3.779310, -3.283186)$, and $(3.584428, -1.848126)$.
This is an interesting function, as we here do not really have to worry so much about getting stuck in a local minima — as these are all equal, and actually also global. Finding *a* minima is thus not really an issue, and basically all configurations of parameter values we tried were successfully able to identify one of the four minima. For example, with 20 initial points, 100 grid points for the Kriging reconstruction per dimension, Matérn covariance with $\nu = 2.5$, and 30 iterations, probability of improvement with $\beta = 0$ readily gave the estimated minima as $(2.993840, 1.984370)$. The optimisation state after this run is shown in figure 9.

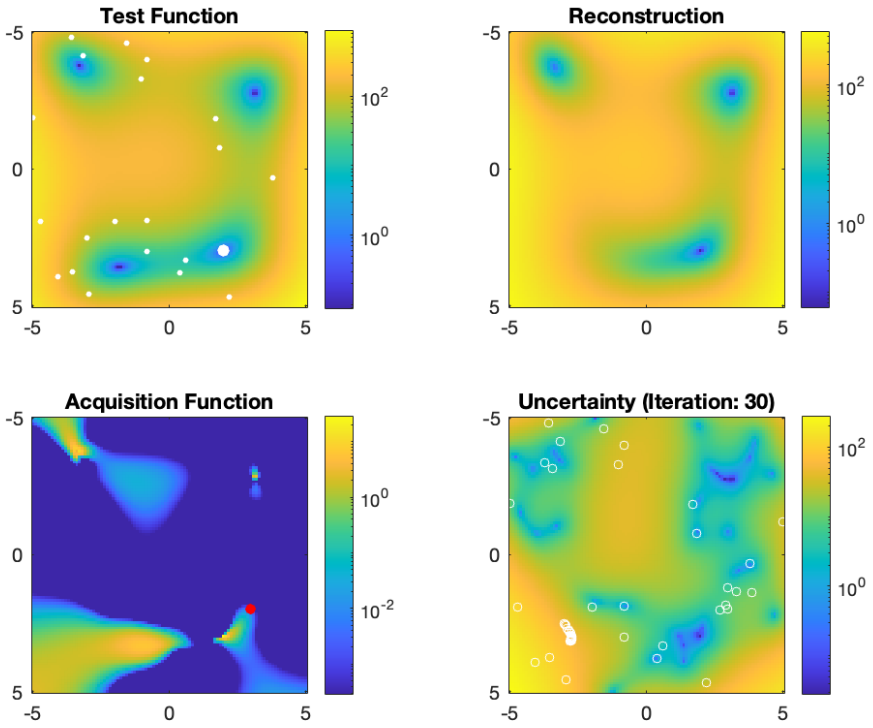


Figure 9: The state of the Bayesian optimizer for Himmelblau's function (see equation (27)) after 30 iterations. The run was made with 20 initial points, 100 grid points for the Kriging reconstruction per dimension, Matérn covariance with $\nu = 2.5$, probability of improvement with $\beta = 0$. The minimum it found was $(2.993840, 1.984370)$.

If finding a minima is all we care about, then there is not much more to say. However, if we wish to potentially locate *all* minima, then we have to give this some more thought. One possibility is to simply vary the parameters a bit. For example, using a different amount of initial points, since we allocate these randomly, allows us to find other minima. All else being equal, probability of improvement then found $(3.802791, -3.308921)$, with the final state being as in figure 10. One could consider other modifications as well, such as different covariance functions (although, in some limited testing, Matérn with $\nu = 2.5$ seems to generally perform a bit better than others for Himmelblau's function in particular), or different magnitude of evaluation noise (although, in testing, it did not in fact seem to make that much of a difference). Changing the β is probably the most obvious choice, as it controls the exploitation–exploration balance, as discussed in subsection 3.3. For example, with $\beta = 1$, probability of improvement

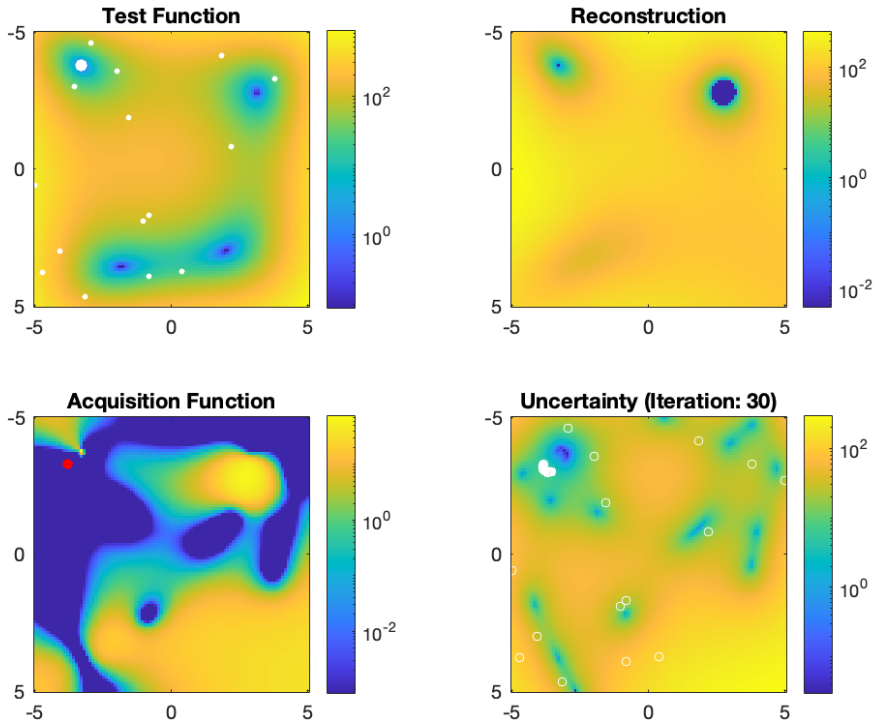
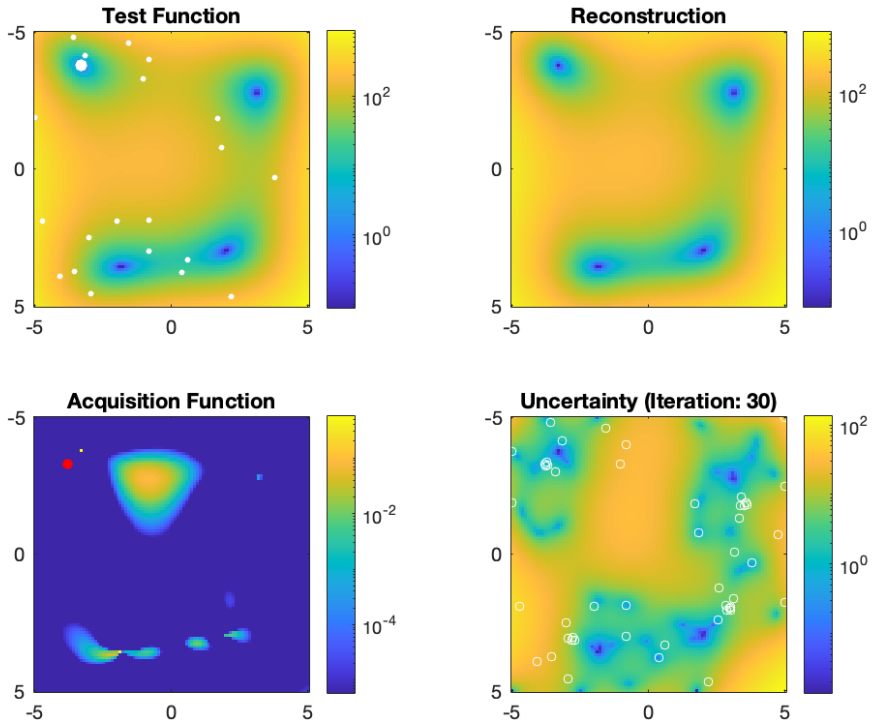


Figure 10: The state of the Bayesian optimizer for Himmelblau's function (see equation (27)) after 30 iterations. The run was made with 15 initial points, 100 grid points for the Kriging reconstruction per dimension, Matérn covariance with $\nu = 2.5$, probability of improvement with $\beta = 0$. The minimum it found was $(-3.802791, -3.308921)$.

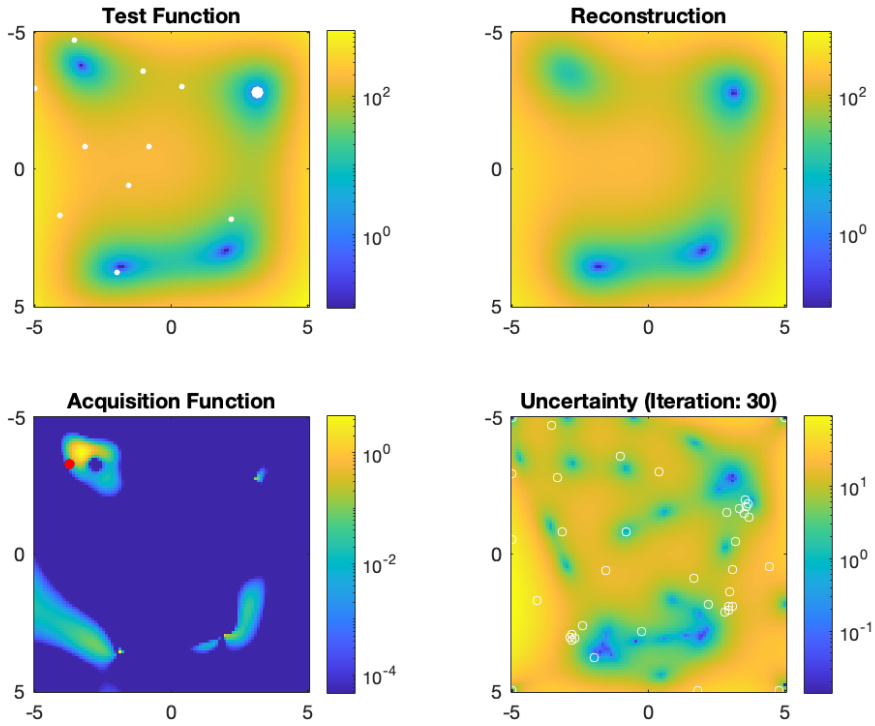
found yet another minima, $(-2.857208, 3.143337)$. As can be seen, though, especially with only 15 initial points (at least with default $\beta = 0$), the function reconstructions using the probability of improvement as acquisition function tends to be quite poor, as it quickly hones in one minima and then stays there.

In contrast, using expected improvement or the lower confidence bound, they will tend to give more of a complete picture of the function even with default values, tending to explore the full domain more, as discussed in subsection 2.5. As such, one could conceivably first use one of these acquisition functions to get a more complete view of what the function looks like in the reconstruction, and then restrict the domain, say, to pinpoint the other minima we can guess are there, perhaps then using the probability of improvement to get as close as we can. Or, if one allowed for a more direct selection of initial points, rather than randomizing them, this could conceivably also lead to different minima being found.

All other conditions being the same, expected improvement and the lower confidence bound also find different minima, so even just for that reason it is worthwhile to change the acquisition function around. For example, with all other initial conditions being the same as for the first probability of improvement run shown in figure 9, we end up with $(-3.776631, -3.280980)$ using expected improvement. The final state for this optimization is shown in figure 11(a). Also, even with only 10 initial points, expected improvement gives a fairly good final reconstruction, as shown in figure 11(b), which could then be used for further pinpointing of the minima suggested by the reconstruction, and not just the absolute minima found.
One could also consider, especially for expected improvement or the lower confidence bound, to search among the function values calculated not just for *the* minima found, but for coordinates giving values close to the minima. After all, we might in just one run jump around to spots close to all the minima in the function.



((a)) The state of the Bayesian optimizer for Himmelblau's function (see equation (27)) after 30 iterations. The run was made with 20 initial points, 100 grid points for the Kriging reconstruction per dimension, Matérn covariance with $\nu = 2.5$, expected improvement with $\beta = 0$. The minimum it found was $(-3.776631, -3.280980)$.



((b)) The state of the Bayesian optimizer for Himmelblau's function (see equation (27)) after 30 iterations. The run was made with 10 initial points, 100 grid points for the Kriging reconstruction per dimension, Matérn covariance with $\nu = 2.5$, expected improvement with $\beta = 0$. The minimum it found was $(-2.799573, 3.142727)$.

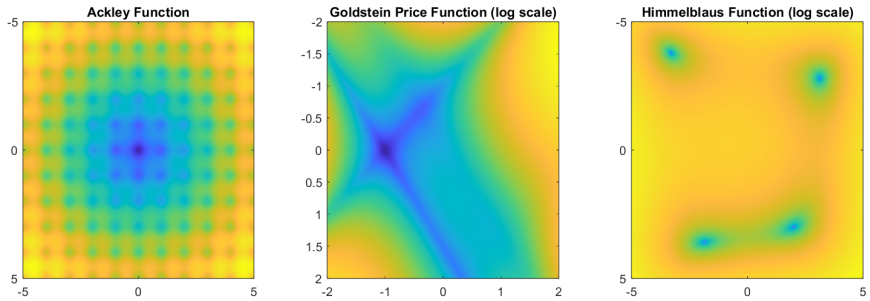


Figure 12: Two dimensional test functions. Ackley, Goldstein-Price and Himmelblau's test functions

3.6 Higher-Dimensional Optimization

In this section, we will consider a test function that we can evaluate in higher dimensions than 2. Specifically, we will consider the Styblinski–Tang function (see [5]), taking the form

$$\frac{\sum_{i=1}^{d}(x_i^4 - 16 * x_i^2 + 5 * x_i)}{2}, \tag{28}$$

with domain $\overbrace{(-5,5) \times \cdots \times (-5,5)}^{d \text{ times}}$, and minima found at $\overbrace{(-2.903534, \ldots, -2.903534)}^{d \text{ times}}$. The value of this minima lies between $-39.16617d$ and $-39.16616d$. Note that we use d as the number of dimensions, which for the Styblinksi–Tang function can take any positive integer value, i.e. $d \in \mathbb{Z}_+$.

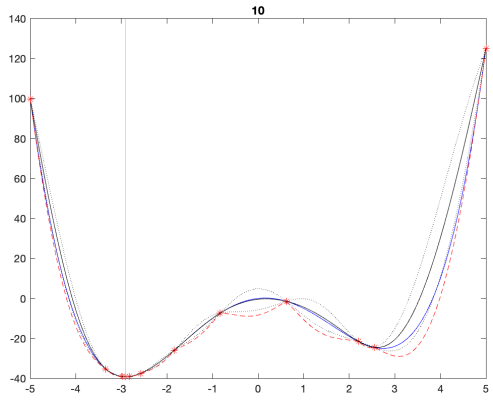
In higher dimensions, there are several things to consider, both entirely new and those with greater importance. The initial number of points turns out to be crucial, as a too-low value now very quickly leads to singular matrices in the process of the optimization. Conversely, we now cannot choose the number of grid points, which we use to evaluate the acquisition function, too large. In our implementation, the grid will in total have n_{grid}^d points, which scales extremely fast. While $n_{grid} = 100$ works well in two dimensions, using even half of that in four, for example, makes the optimization prohibitively slow.

Both these points of interest are of course subject to other conditions, limiting our choice of them at the other end. Since our assumption is that the functions we want to find the minima of are expensive to evaluate (indeed, the whole point of making the Bayesian optimization algorithm in the first place), we want to keep the number of initial points low. As far as the number of grid points, we are using the grid to evaluate the acquisition function and get a minimum which we then feed to an optimizer for refinement. With a less fine grid, as we are forced to use in higher dimensions, we run a higher risk of the optimization falling into local minima. While not implemented in our optimizer, one solution to this trade-off would be to do batch evaluation. Instead of doing one evaluation per round, one would perform multiple evaluations, to reduce the number of times that we have to construct the grid, thus allowing for a slightly finer grid while keeping the optimization less slow.

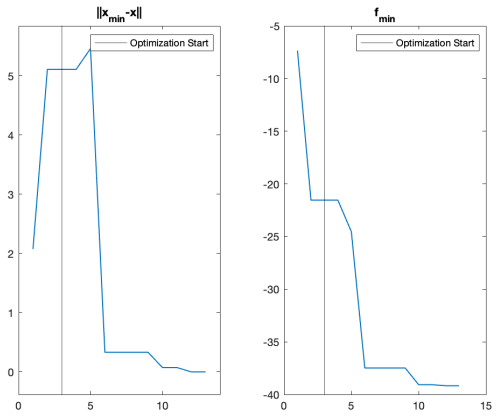
Further, in dimensions higher than two we cannot (easily, at least) visualise the function reconstructions. To get a feel of the Styblinski–Tang function, then, we first show the results after optimizing in one and two dimensions, before moving on to three and more.

We note that the Styblinski–Tang function has local minima at points where one (or more) of the parameter values is exchanged with approximately 2.75. As probability of improvement is especially vulnerable to getting stuck in local minima, we thus opt for one of the other acquisition functions. The lower confidence bound seems to do well in one dimension. With a Matérn covariance with $\nu = 2.5$, $\beta = 3$, 3 initial points, and a grid of 100 points, we quite readily find the minima in 10 iterations. The state of the optimizer after 10 iterations, as well as the path of improvement, with these initial conditions, are shown in figure 13. For a more thorough explanation of paths of improvement, see subsection 2.5.

Changing only the number of initial points and iterations, we achieve quite good results with the lower confidence bound also in two dimensions. With 10 initial points and 30 iterations, we get an



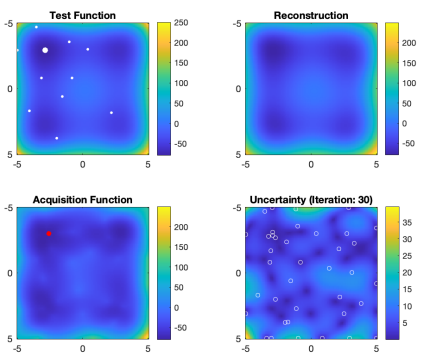
((a)) The state of the Bayesian optimizer after 10 iterations.



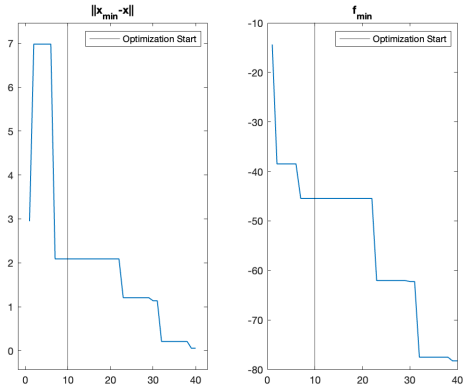
((b)) The path of improvement.

Figure 13: Bayesian optimization of the Styblinski–Tang function in one dimension (see equation (28)). The run was made with 3 initial points, 100 grid points for the Kriging reconstruction, Matérn covariance with $\nu = 2.5$, lower confidence bound with $\beta = 3$. The approximation of the minima was then -2.903317.

approximation of the minimum to $(-2.950457, -2.869730)$. The state of the optimizer, and the path of improvement, are shown in figure 14.



((a)) The state of the Bayesian optimizer after 30 iterations.



((b)) The path of improvement.

Figure 14: Bayesian optimization of the Styblinski–Tang function in two dimensions (see equation (28)). The run was made with 10 initial points, 100 grid points per dimension for the Kriging reconstruction, Matérn covariance with $\nu = 2.5$, lower confidence bound with $\beta = 3$. The approximation of the minima after 30 iterations was $(-2.950457, -2.869730)$.

Having now gotten a sense of the function, we turn to optimizing in more dimensions. In three dimensions, to not have the optimization take a lot of time and computing power, we need to decrease the grid size. 50 seems to be small enough when working in three dimensions. To compensate both for the decrease in grid size, and in general, the fact that higher-dimensional optimization is harder, we need to start with more initial points, and also increase the amount of iterations used. This was seen already comparing one to two dimensions — in one dimension we could get away with a lot fewer initial points, and also fewer iterations. For this specific case, we get a decent estimate after 100 iterations with 20 initial points, namely $(-2.906086, -2.903996, -2.903481)$. The path of improvement is shown in figure 15. As can be seen, we got stuck in a local minima for a while between around iteration 30 and 70 or so, so it is only with more patience that we arrive at a decent estimate of the global minimum.

For four dimensions, we get an (indeed) exponential increase in difficulty. The total number of grid

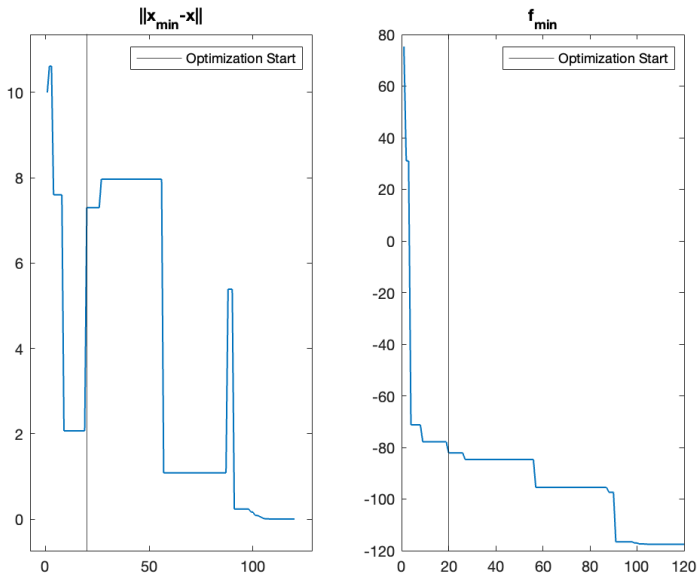


Figure 15: Path of improvement for the Bayesian optimization of the Styblinski–Tang function in three dimensions (see equation (28)). The run was made with 20 initial points, 50 grid points per dimension for the Kriging reconstruction, Matérn covariance with $\nu = 2.5$, lower confidence bound with $\beta = 3$. The approximation of the minima after 100 iterations was $(-2.906086, -2.903996, -2.903481)$.

points used in three dimensions was $50^3 = 125000$, but even with only 20 grid points per dimension, this amounts to $20^4 = 160000$ in total, and as such, each iteration requires a lot more time, even with this relatively low resolution. Further, we need more initial points, and more iterations to find the true minimum — we found an approximation of $(-2.903477, -2.904126, -2.903986, -2.904618)$ only after close to 200 iterations with 40 initial points. As can be seen from the path of improvement in figure 16, we once again run into several local minima on the way, until we finally make our way toward the global minimum close to iteration 160 (i.e. after 200 evaluation points in total).

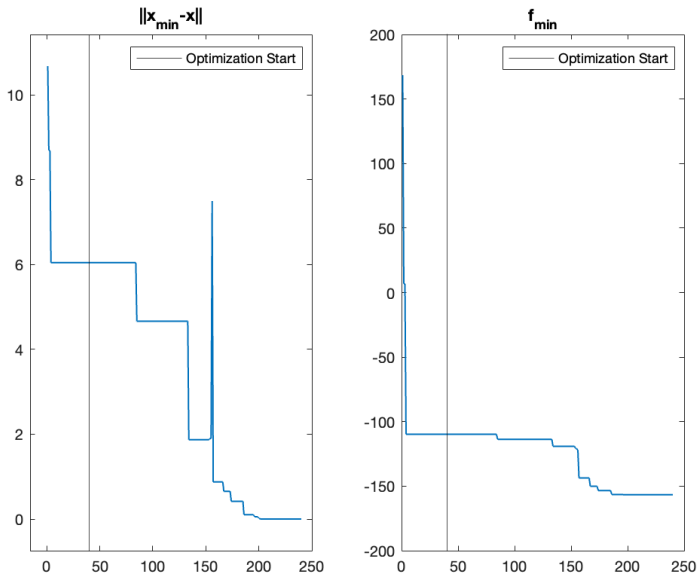


Figure 16: Path of improvement for the Bayesian optimization of the Styblinski–Tang function in four dimensions (see equation (28)). The run was made with 40 initial points, 20 grid points per dimension for the Kriging reconstruction, Matérn covariance with $\nu = 2.5$, and lower confidence bound with $\beta = 3$. The approximation of the minima after 200 iterations was $(-2.903477, -2.904126, -2.903986, -2.904618)$.

Finally, we try to see if we can get the optimization to work in five dimensions. We note that for a comparable amount of grid points as in three or four dimensions, we need to go down all the way to around 10 grid points per dimension (resulting in a total of $10^5 = 100000$). As noted before, we then run even more of a risk of getting stuck in local minima. After failing to find the global minimum after 200 iterations and 50 initial points with the lower confidence bound with $\beta = 3$, we first try to increase the β, but find that the minima then found are even worse, maybe getting too inaccurate. Instead, we find that the probability of improvement, on its own less explorative, but in general more precise, is able to give decent results with a non-zero β, in our case using a value of 5. We also increased the number of initial points further to 60, but could on the other hand, with the change of acquisition function and β, lower the number of iterations to 150. We then are able to approximate the minimum at least fairly closely, to a value of $(-3.035499, -2.985286, -2.761779, -2.943341, -2.969109)$. The path of improvement is shown in figure 17. Note that it seems like we could have stopped the optimization earlier, after only about 100 iterations, and gotten the same result.

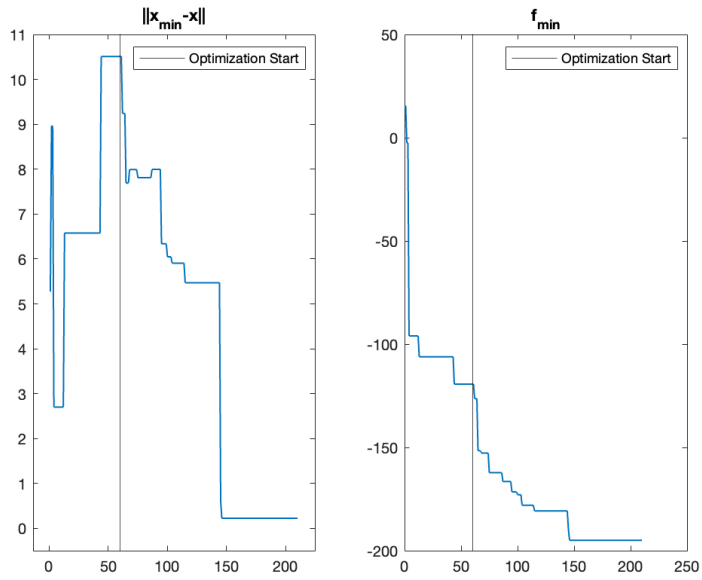


Figure 17: Path of improvement for the Bayesian optimization of the Styblinski–Tang function in five dimensions (see equation (28)). The run was made with 60 initial points, 10 grid points per dimension for the Kriging reconstruction, Matérn covariance with $\nu = 2.5$, and probability of improvement with $\beta = 5$. The approximation of the minima after 200 iterations was $(-3.035499, -2.985286, -2.761779, -2.943341, -2.969109)$.

Due to the exponential increase in the number of evaluation points per dimension, our optimizer is not easily able to scale well into higher dimensions. With some clever tricks, though, it is possible to get decent results in not all too much time. Aside from changing the acquisition function and/or β parameter, and increasing the number of initial points, one could also as mentioned previously do more evaluations per iteration. Of course, with more powerful computers, one could conceivably also simply allow for a relatively fine grid, although even that has of course limited use. Another strategy would be to search over a smaller part of the domain if one has some prior knowledge of around where the minimum is likely to be, although this will probably not be the case a lot of the time.

We also note that in higher dimensions, it is generally more effective to increase the amount of iterations, rather than to increase the grid size. If we double the number of iterations, that leads simply to a total doubling of the number of evaluations we need to make. Increasing the number of grid points by 50% per dimension, on the other hand, leads in 5 dimensions, say, to a $1.5^5 \approx 7.59$-fold increase.

4 Conclusions

Bayesian optimization presents a powerful approach for solving optimization problems, particularly in scenarios where the objective function is expensive to evaluate or its analytical form is unknown. By leveraging surrogate models, such as Gaussian processes with various covariance functions, and selecting appropriate acquisition functions, Bayesian optimization provides a flexible framework capable of achieving efficient results.

However, for black-box functions, the versatility of Bayesian optimization can also pose challenges. The abundance of options for surrogate models and acquisition functions may complicate the process of identifying the most suitable configuration, especially in the absence of prior knowledge about the objective function.

5 Outlook

Several challenges and limitations were encountered during this project. One significant issue is the exponential computational cost associated with the implementation, particularly for the internal optimization of the acquisition function. Future work should explore methods to reduce these computational costs, thereby making Bayesian optimization more practical for problems with higher-dimensional input spaces.

Another key challenge has been managing the trade-off between exploration and exploitation. To address this, we propose the development of adaptive criteria or dynamic acquisition functions that can adjust over the course of the optimization process. For example, the exploration parameter β of the acquisition functions could be made iteration-dependent, prioritizing exploration during the early stages and shifting toward exploitation in later stages. Such an approach could enhance the efficiency and effectiveness of Bayesian optimization in diverse applications.

References

[1] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[2] P. I. Frazier, "A tutorial on bayesian optimization.," 2018.

[3] *Handbook of spatial statistics.*, pp. 46–49. Chapman amp; Hall/CRC handbooks of modern statistical methods, CRC Press, 2010.

[4] *Handbook of spatial statistics.*, pp. 41–43. Chapman amp; Hall/CRC handbooks of modern statistical methods, CRC Press, 2010.

[5] "Test functions for optimization." https://en.wikipedia.org/wiki/Test_functions_for_optimization. Last retrieved 2024-12-13.