Introduction to Bayesian Optimization

Drew Gjerstad

Contents

1 Introduction

Bayesian optimization refers to an optimization approach that uses Bayesian inference to guide the optimizer to make "better" decisions under *uncertainty*. In addition, this approach provides a framework that enables us to strategically tackle the uncertainty inherent in all optimization decisions. One particularly attractive property of this framework is its unparalleled *sample efficiency* which refers to its ability to progress towards optima "quicker" than other optimization approaches. We will discuss this property in more detail later.

The goal of these notes is to introduce the notion of Bayesian optimization from a high-level perspective as well as introduce the components (often referred to as *primitives*) involved in the so-called "Bayesian optimization workflow". Each of these components, their importance, and how they fit into the workflow will be discussed in detail in other sets of notes. The list of links below route to sets of notes discussing these components in addition to topics related to Bayesian optimization. Note that these notes as well as examples, tutorials, and from-scratch implementations can be found in the `bayesian-optimization` GitHub repository.

- **Bayesian Decision Theory**
- **Gaussian Processes**
- **Covariance Functions and Kernels**
- **Model Evaluation and Selection**
- **Utility Functions**
- **Acquisition Functions**
- **GP Regression**
- **GP Classification**

Before we jump into formalizing the idea of Bayesian optimization, we first pause to discuss the motivation–both theoretical and practical–behind such an approach.

2 Motivation

2.1 Theoretical Motivation

First, we consider the theoretical motivation for the Bayesian optimization approach. Typically, the theoretical motivation stems from the characteristics of the *objective*–the "function" we are aiming to optimize. Keep in mind that the *objective "function"* does not have to be a boilerplate mathematical expression and could be a score representing how a particular "solution" (i.e., set of parameters) performs. Furthermore, the list below is by no means exhaustive; there exists additional theoretical motivation but these are seemingly the most common, particularly with regard to the characteristics of the objective.

- **Black-box objective functions** are functions that we can only interact with via its inputs and outputs meaning classical, analytical methods (such as gradient-based ones) do not work. In such cases, Bayesian optimization allows us to approximate such an objective while managing the uncertainty.
- **Expensive-to-evaluate objective functions** are functions that require significant computation effort to obtain their output. However, just as with black-box objectives, we can use Bayesian optimization to approximate and model these efficiently.
- More generally, this approach is very useful when the objectives lack analytical evaluation (or, if analytical evaluation is expensive).
- In some spaces such as the discrete or combinatorial ones, the objective may not have efficient gradients (if they exist at all). Thus, classical gradient-based optimization methods are incompatible.

2.2 Applications

The potential applicability of Bayesian optimization can be seen across several critical domains, especially those that aim to accelerate the identification of solutions to real-world scientific and engineering problems. These applications include:

- Drug discovery
- Molecule/protein discovery
- Materials design
- AutoML (i.e., hyperparameter tuning)
- Engineering decisions

The applications presented on the next few pages illustrate how this approach is used for real-world applications. Alongside each graphic are some brief comments explaining why the approach is advantageous. Once again, similar to the list of theoretical motivations, this is not an exhaustive review but rather a concise showcase of where the approach comes in handy.

Application: Drug Discovery

Figure 1 below illustrates an example of a chemical process optimization problem that is common in drug discovery. On the left side, four classical approaches are shown but there is one issue with these approaches: they are expensive. In general, such approaches are expensive due to the cost of preparing and executing experiments in a lab. Ideally, we should use the results from previous experiments to help us design future experiments–exactly where Bayesian optimization comes into play.

Thus, an alternative to the expensive classical approaches, the Bayesian optimization approach shown on the right side uses a model of the objective (where the objective is a *reaction parameter* representing the characteristics of a reaction) to principally select the next chemicals to test while handling the uncertainty in the model. Specifically, the data from previous experiments is used to locate points that may optimize the unknown objective and these located points should be used when planning future experiments. Furthermore, these located points typically represent samples of *high utility*, where high utility refers to the property of a sample that is expected to be useful in optimizing the reaction parameter.

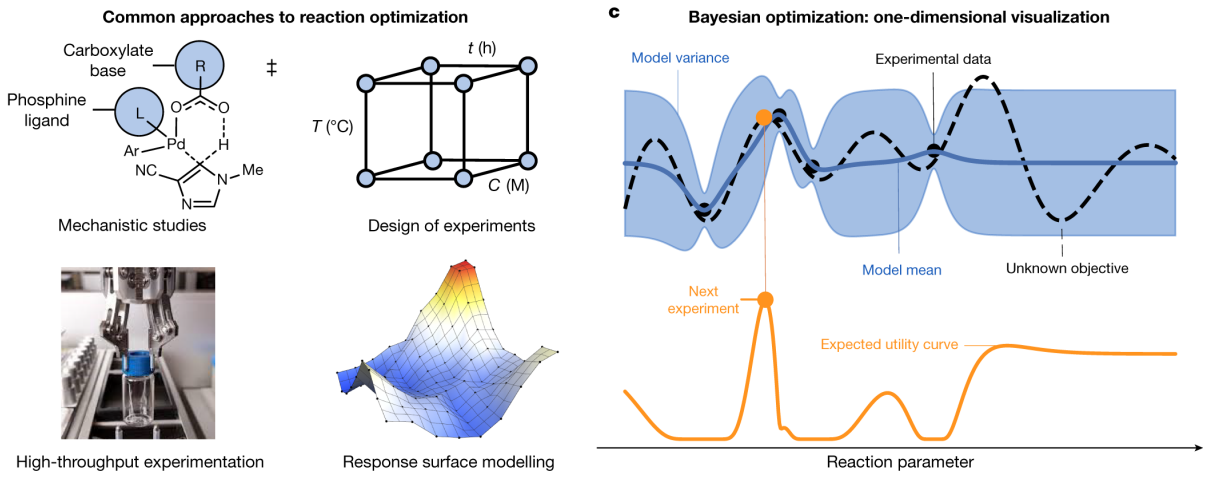


Figure 1: From *Bayesian optimization as a tool for chemical synthesis* by Shields et al. (2021)

Application: Molecule/Protein Discovery

In the field of molecule and protein design, there are similar considerations to the ones in the previous application: experiments are costly. Figure 2 shows the integration of the Bayesian optimization approach with experimentation. In most environments, scientists synthesize and test several different formulations to obtain a dataset. Then, this dataset is used to help model the underlying objective and can be used to suggest new, promising formulations. After making suggestions, the new formulations are synthesized, tested, and added to the dataset so as to inform the model and optimizer of formulations to suggest next.

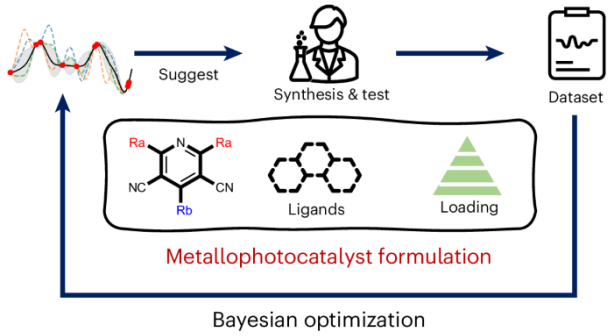


Figure 2: From *Sequential closed-loop Bayesian optimization as a guide for organic molecular metallophotocatalyst formulation discovery* by Li et al. (2024)

Application: Materials Discovery

Figure 3 is similar in style to Figure 2 except now it showcases some additional details specific to the design and discovery of materials. Once again, the initial dataset is used to model the underlying objective and inform design exploration where the results from exploration being used to augment the dataset. Furthermore, in this particular example, the researchers are also using the results of experiments to calibrate a simulation of designs, another application of Bayesian optimization.

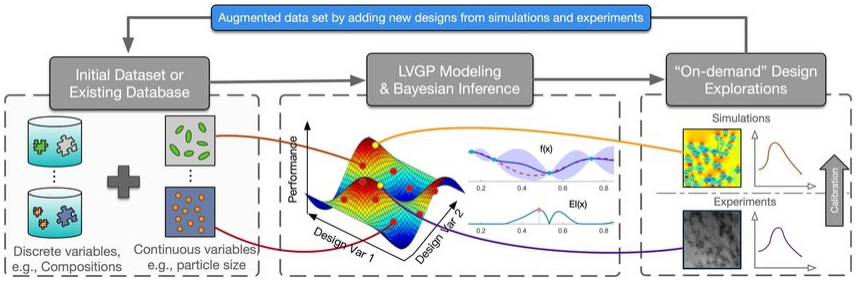


Figure 3: From *Bayesian optimization for Materials Design with Mixed Quantitative and Qualitative Variables* by Zhang et al. (2020)

Application: AutoML

Figure 4 demonstrates a workflow for tuning hyperparameters for a neural network using Bayesian optimization, using a Gaussian process as a surrogate model. For some context, *AutoML* is the process of automating the machine learning workflow with regard to tuning models' hyperparameters to optimize performance. While other methods for this task exist, such as randomized search and grid search, they are often computationally expensive or lack a principled approach to aid in finding the optimal set of hyperparameters.

Instead, we can use Bayesian optimization to perform this tuning in a more efficient manner by identifying the next set of promising hyperparameters based on the current model and its uncertainty. One particular reason that using the Bayesian optimization approach for AutoML is advantageous is that continuous hyperparameters only represent a *very* small subset of machine learning hyperparameters. For instance, many hyperparameters are discrete such as the number of hidden layers in a network or are categorical such as the choice of activation function for network layers. Therefore, these cannot be optimized using traditional (i.e., gradient-based) methods.

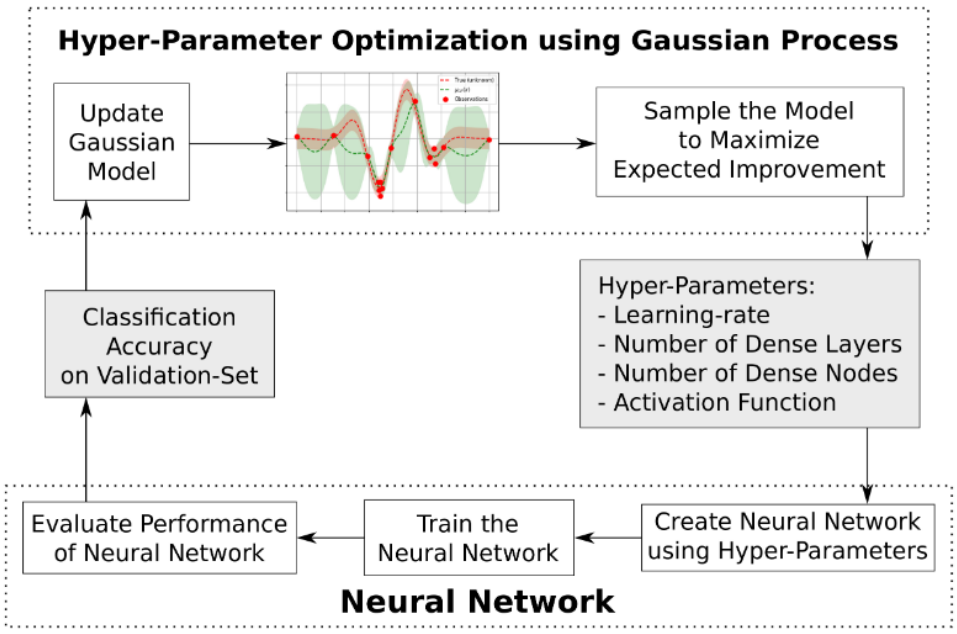


Figure 4: From *Achieve Bayesian optimization for tuning hyperparameters* by Edward Ortiz on *Medium* (2020)

Application: Engineering Decisions

Figure 5 illustrates how Bayesian optimization can be used to calibrate a particle accelerator in a similar manner to the previous applications. The "operator" inputs the target beam parameters while a camera inputs the observed beam parameters. Then, Bayesian optimization determines the changes (i.e., the next set of beam parameters) to ideally improve the calibration of the particle accelerator. This type of task is similar to AutoML: we want to identify a set of parameters that optimize performance (an integral part of engineering design and corresponding decisions). Once again, many of these parameters are discrete or categorical meaning we cannot rely on traditional methods.

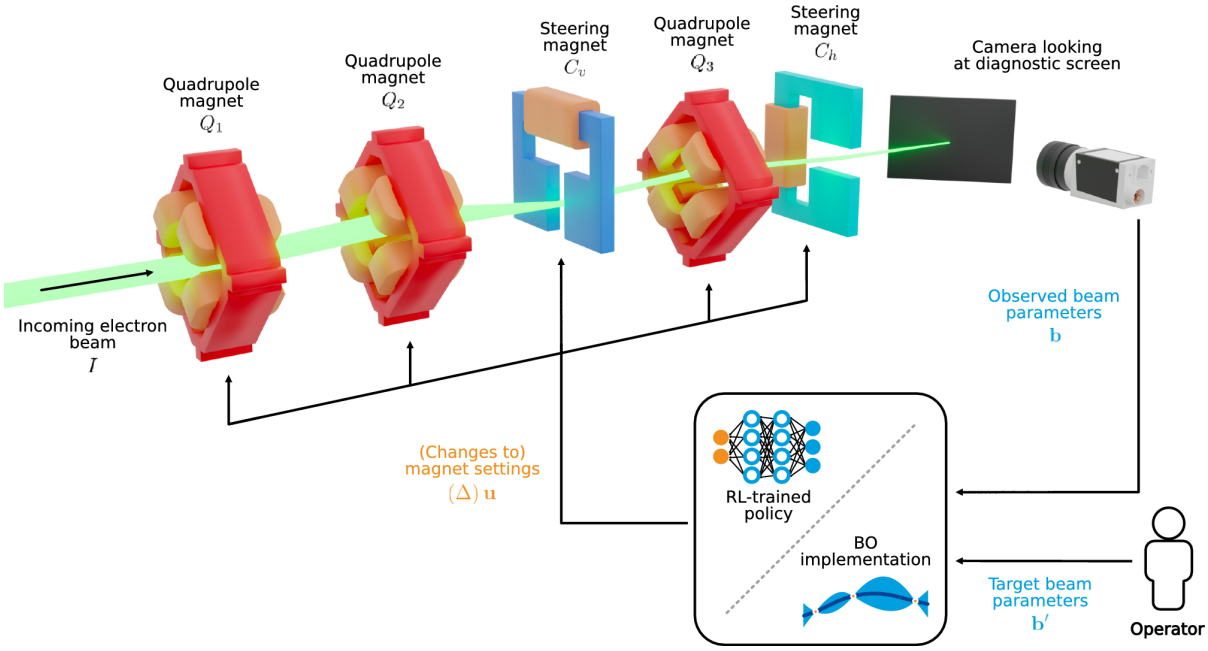


Figure 5: From *Reinforcement learning-trained optimizers and Bayesian optimization for online particle accelerator tuning* by Kaiser et al. (2024)

3 Optimization Foundations

In this section, we introduce the foundations of optimization to help understand the ideas that Bayesian optimization builds on. We will discuss formalizing optimization problems, objective functions, observation models, optimization policies, and termination policies.

- **Optimization** is a process and field of study that aims to efficiently locate the optimal objective value and/or its location from the search domain.

For a more thorough review of optimization, the reader is directed to Nocedal and Wright's *Numerical Optimization* book (in the Springer Series in Operations Research). We begin with an introduction to formalizing an optimization problem.

3.1 Formalization of Optimization

First, let's formalize a typical optimization problem. Notice that the formulation here is a simple and flexible one for global optimization and is not inherently Bayesian. Additionally, while the formulation below does imply *continuous* optimization, it could be modified to support other types of inputs such as discrete, categorical, etc.

$$x^* \in \arg\max_{x \in \mathcal{X}} f(x) \qquad f^* = \max_{x \in \mathcal{X}} f(x) = f(x^*)$$

where $f : \mathcal{X} \to \mathbb{R}$ is a real-valued *objective function* on some domain $\mathcal{X}$, x^* is the point that obtains the global maximum value f^*. Note that the max versus min is arbitrary and depends on the entirely on the problem at hand.

- **Black-box optimization** arises from the fact that we do not need (or do not have access to) an explicit objective function f but rather only some information about the objective at identified points.

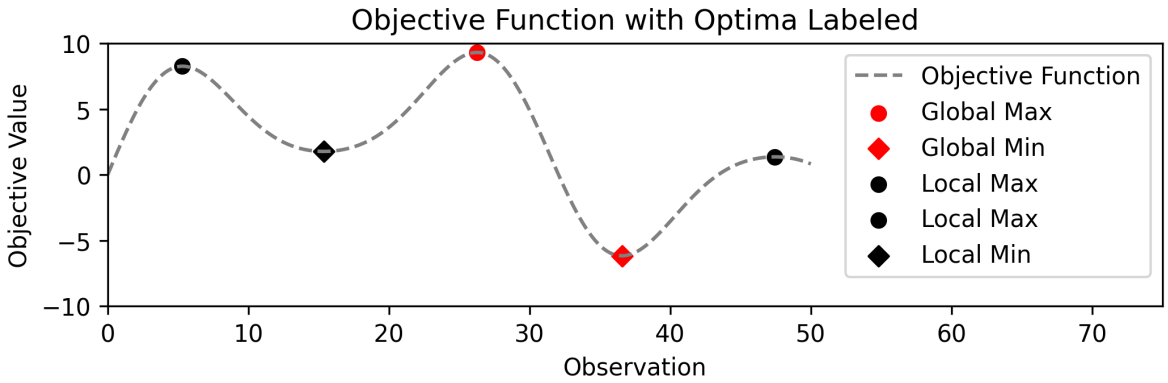
The plot below illustrates an objective function with both its global and local *optima* labeled.

- **Optima** are the points that either maximize or minimize (optimize) the objective function.

The objective function plotted is given by the following expression:

$$f(x) = 10 \sin(0.2x) \cos(0.1x) + \sin(0.5x) + 0.05x$$

Note that the optima for the objective function $f(x)$ were located by obtaining the values of the objective at each point in the *domain*. However, in optimization, we usually want to avoid such computations due to the associated cost of doing so. Specifically, we prefer methods that locate the optima in a more efficient manner.

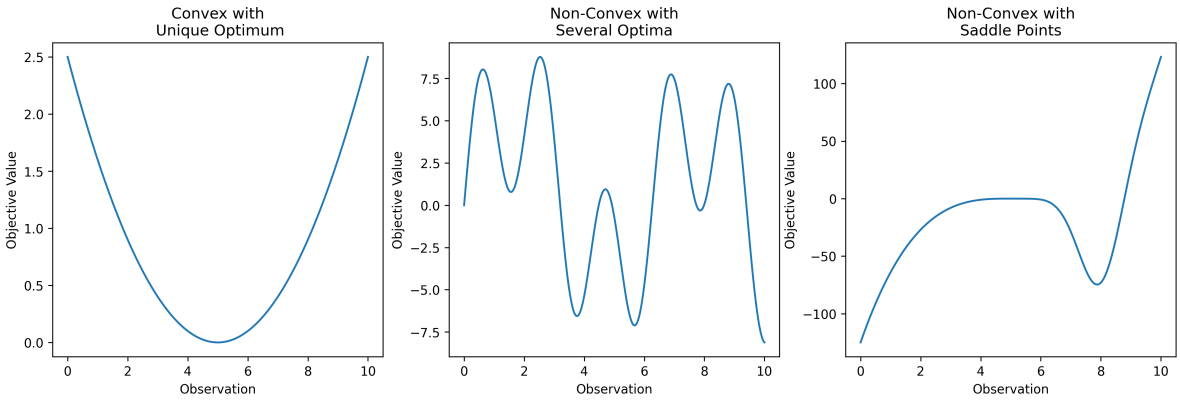


Furthermore, if we face a *black-box objective* then we don't have an explicit mathematical expression for the objective that we can evaluate across the domain. Instead, we have the inputs and their corresponding outputs at a select number of points and must go from there.

3.2 Objective Functions

The **objective function** is the function that we want to optimize. Recall that this does not necessarily have to be a function in the typical mathematical sense. Instead, as is common in many practical cases, the objective function represents a score for a given input such as the reaction parameter in the drug discovery example or the classification accuracy of a neural network in the AutoML example. Furthermore, in many of these real-world applications, there is no analytical or mathematical model to describe the objective–serving as one of the motivations for Bayesian optimization.

In general, objective functions often have different characteristics which impact how various optimization approaches behave. For instance, three different objective functions are plotted below with one being convex and two being non-convex. The left plot is convex and has a unique optimum, the center plot is non-convex with several optima, and the right plot is non-convex with saddle points and a unique optima.



Since the plot on the left is convex, we can apply gradient-based methods to locate the unique optimum. However, the center and right-side plots are more difficult. Consider the center plot, for example. If we use gradient-based methods then we run the risk that we become "trapped" in one of the local optima and not reach the global optimum (global minimum: $x \approx 5.75$, global maximum: $x \approx 2.5$). Another troubling case is depicted in the plot on the right: saddle points. Saddle points are problematic for classical methods since they form a flat region which prohibits the optimizer from reaching the optimum.

Beyond these challenges, there are other characteristics that objective functions may have which lead to issues and are typically addressed via the Bayesian optimization approach. Such characteristics coincide with the theoretical motivation and include:

- The objective is considered to be a *black-box* function meaning we can only interact with the objective via its inputs and outputs.
- The objective's returned value is *corrupted* by some sort of noise and does not represent the exact true objective value at that location.
- The objective has a *high cost* of evaluation and requires a sample efficient method to avoid expensive probing (evaluation).
- There *do not exist gradients* (if there were, efficient gradient-based methods could be employed to locate and evaluate optima).

3.3 Observation Models

Observation models are similar to the idea of surrogate models used in the Bayesian optimization workflow but there are some key differences. The observation model is used to describe how the true objective is *observed*, usually accounting for some form of (additive) noise. On the other hand, the surrogate model is a probabilistic model (i.e., Gaussian process) used to approximate the unknown objective function.

Specifically, the **observation model** is an approach to formalize the relationship between the true objective function, the actual observation, and the noise. This is rather important since the model used to relate the true objective and actual observations must account for uncertainty due to noise. Mathematically, this is the probability distribution of y given the sample location x and true objective function f:

$$p(y|f, x)$$

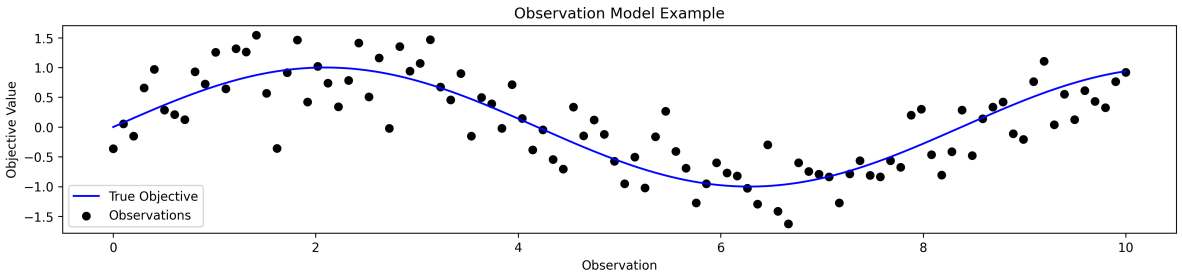
To account for uncertainty, we assume that the observations are *stochastically* dependent on the objective. Mathematically, we assume an additive noise term ε:

$$y = f(x) + \varepsilon$$

Let $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Then, the model becomes:

$$p(y|x, f, \sigma) = \mathcal{N}(y; f, \sigma^2)$$

Thus, the observation y at sample location x is treated as a *random variable* which follows a Gaussian, or normal, distribution with mean f and variance σ^2. This leads to the distribution of y being centered around the true objective value at sample location x, $f(x)$.



In the plot above, the true objective is shown in addition to 100 observations that have an additive term of random (Gaussian, or normal) noise. The observation model will need to take this into account since noise, often called *uncertainty* in Bayesian optimization, is inherent in real-world problems. It can come from measurement errors, environmental factors, or simply the randomness in the system being optimized.

3.4 Optimization Policies

In simple terms, the optimization policy handles the repeated interactions between the "inner-workings" of the policy and the environment, with the environment typically being *noise-corrupted* (hence the need for an observation model).

More definitively, a **policy** is a mapping function that takes in a new input observation plus any existing observations and uses a *principled* sampling approach to output the next observation location. It will also decide if it should perform another iteration (select a new observation) or terminate the optimization process (see *Termination Policies* below).

For most applications, we want the policy to ideally be learning and improving such that it will guide the search toward the global optimum more effectively. Furthermore, the iterative policy improvement should lead to a good policy that retains the (typically limited) sampling budget for more promising candidate points. A policy that does not consider observed data are known as a *non-adaptive* policy and is not ideal for costly observations.

Note that in some literature, there is little distinction between the optimization policy and the termination policy (discussed next). To be clear, the termination policy is one of the several components that make up the optimization policy. This lack of distinction is not unreasonable but rather something to be aware of when reviewing the vast literature on optimization.

3.5 Termination Policies

A **termination policy** is the final decision in the optimization loop. The policy decides whether to terminate immediately or continue with another observation (continue to optimize the objective). Such policies can be *deterministic* or *stochastic*.

- A **deterministic termination policy** is one that will stop the optimization process after reaching a goal or exhausting a pre-defined search budget.
- A **stochastic termination policy** is one that will depend on the observed data and some level of randomness or probability.

Note that this piece of the optimization process can be handled by an external agent or be dynamic (i.e., a deterministic or stochastic termination policy). For the purposes of this notebook, we will not focus on specific termination policies here. This is primarily due to the fact that the termination policy depends heavily on the approach or method and the problem/application.

3.6 Diagram of the Optimization Process

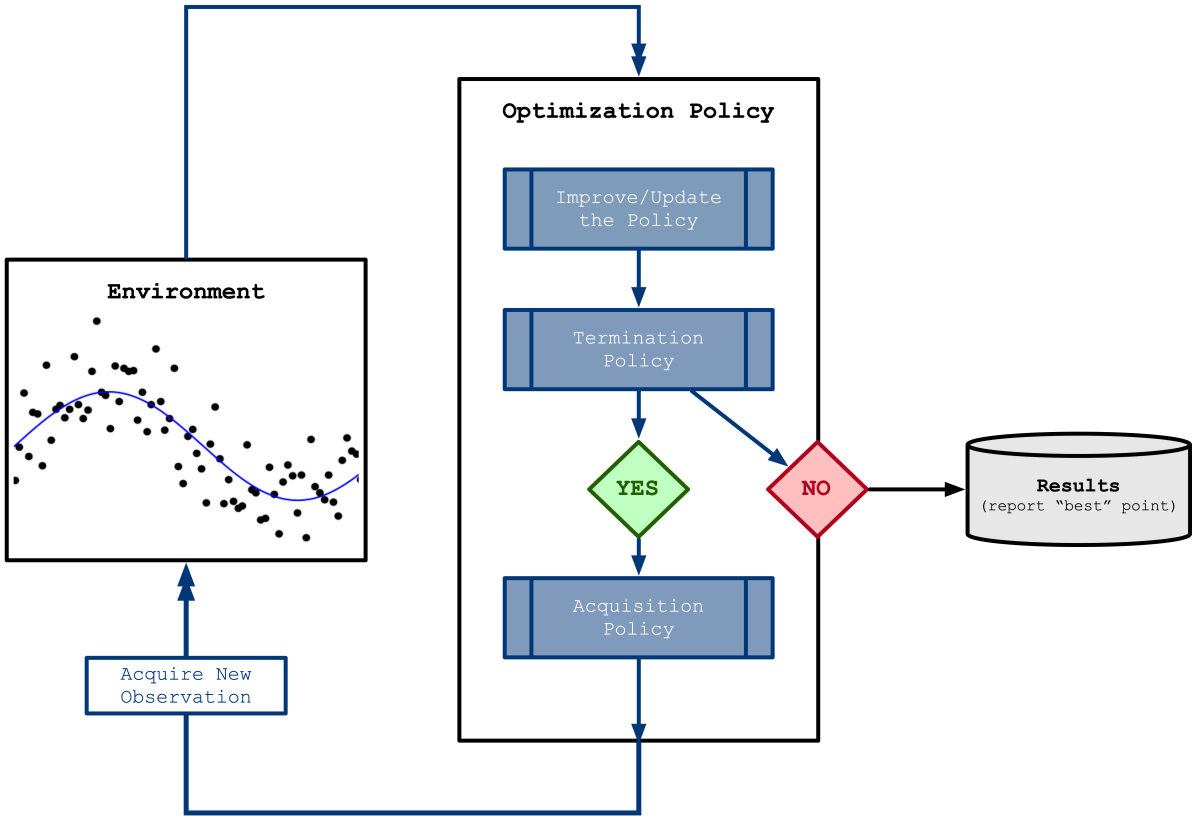


Figure 6: General Optimization Process Diagram

4 Bayesian Foundations

Before we venture into the world of Bayesian optimization, we must first review some foundations of Bayesian statistics. For a more comprehensive examination of Bayesian statistics, the reader is referred to *Mathematical Statistics and Data Analysis* by John A. Rice or *Bayesian Optimization* by Roman Garnett (which provides an optimization-focused review).

4.1 Bayesian Statistics

Bayesian statistics provide us with a systematic and quantitative approach to reason about uncertainty using probabilities. Thus, in *Bayesian* optimization, we use *Bayesian* statistics to reason about uncertainty in the observation (or surrogate) model.

One of the central concepts in Bayesian statistics is **Bayesian inference**. Bayesian inference uses Bayes' theorem to reason about how the prior distribution $p(\theta)$, the likelihood $p(\text{data}|\theta)$, and the posterior distribution $p(\theta|\text{data})$ interact with each other. Note that θ represents the parameter of interest.

Recall **Bayes' theorem**:

$$p(\theta|\text{data}) = \frac{p(\text{data}|\theta)p(\theta)}{p(\text{data})}$$

Let's take a step back and break this down. First, Bayesian inference is a framework that allows us to infer uncertain features of a system of interest from observations using the laws of probability. Thus, within this framework, all unknown quantities are denoted by *random variables*. This is convenient as we can express our beliefs using probability distributions reflecting plausible values.

The **prior distribution**, $p(\theta)$, represents our beliefs before we observe any data. For instance, if we believe that the data is normally distributed then we would likely define the prior distribution to be the Gaussian normal distribution with mean μ and standard deviation σ.

Then, we can refine our initial beliefs once we have observed some data using the **likelihood function**, $p(\text{data}|\theta)$. The likelihood function, or likelihood, provides the distribution of observed values (y) given the location (x), and values of interest (θ).

Finally, using the observed value y, we can derive the **posterior distribution**, $p(\theta|\text{data})$, using Bayes' theorem (defined above) where data $= (x, y)$. This so-called posterior distribution acts as a "compromise" between our initial beliefs from the prior and our observations from the likelihood. It is at the heart of Bayesian optimization and is used to update the surrogate model as we acquire additional observations.

4.2 Bayesian Inference of the Objective Function

The primary use of Bayesian inference in Bayesian optimization is to reason about the uncertainty in the objective function. Specifically, the probabilistic belief we use over the objective function is called a *stochastic process*. A **stochastic process** is a probability distribution over an infinite collection of random variables, for example, the objective function value at each point in the domain.

We will use a **prior process**, $p(f)$, to express our assumptions (beliefs) that we may have about the objective function. Then, we can define a stochastic process using the distribution of the function values $\boldsymbol{\phi}$ given a finite set of points $\boldsymbol{x}$:

$$p(\boldsymbol{\phi}|\boldsymbol{x})$$

The "gold standard" stochastic process used in Bayesian optimization is the **Gaussian process** due to its expressivity and flexibility, in addition to the fact that many of these finite-dimensional distributions are multivariate Gaussian (or approximately so).

Let's now return to discussing how Bayesian inference is applied over the objective. Suppose we make a set of observations at locations $\boldsymbol{x}$ with corresponding values $\boldsymbol{y}$. Let $\mathcal{D} = (\boldsymbol{x}, \boldsymbol{y})$ be the dataset of aggregated observations. Bayesian inference will account for these observations via the formation of the **posterior process**, akin to the posterior predictive distribution:

$$p(f|\mathcal{D}) = \int p(f|\boldsymbol{x}, \boldsymbol{\phi}) p(\boldsymbol{\phi}|\mathcal{D}) \, \mathrm{d}\boldsymbol{\phi}$$

5 The Bayesian Approach

The "Bayesian approach", particularly in the context of optimization, refers to a philosophical approach that uses Bayesian inference to reason about uncertainty. Specifically, the Bayesian approach enables us to tackle the inherent uncertainty in optimization decisions which is crucial since our decisions will determine our success. This is accomplished through a systematic reliance on probability laws and Bayesian inference during optimization.

Recall that the objective function is viewed as a random variable that will be informed by our prior expectations and posterior observations. The Bayesian approach will play an active role in the optimization process to guide the optimization policy by evaluating the merit of a candidate observation. This results in **uncertainty-aware optimization policies**.

5.1 Uncertainty-Aware Optimization Policies

The optimization policy determines the decisions made during the optimization process. To reasonably handle the uncertainty of the objective function, the policy should use the available data to determine the successive observation locations optimally. There is only one requirement from ourselves: we need to establish our preferences for what data or what "kind" of data we want to acquire. Then, we design the policy to maximize such preferences.

Clearly, this will require some sort of framework to make decisions in this way, especially in the face of uncertainty. A natural choice is **Bayesian decision theory** which will be discussed in more detail in another set of notes (see the *Additional Notes* section).

As we continue to explore Bayesian optimization and uncertainty-aware policies, a common theme will begin to emerge: all Bayesian optimization policies handle the uncertainty in the objective function in a uniform manner, a property that is defined implicitly within an optimal *acquisition function*.

6 Bayesian Optimization Workflow

The Bayesian optimization workflow consists of two main *primitives*: the surrogate model and the acquisition function. In this section, we will briefly discuss these two primitives to understand how they fit into this workflow. Note that these topics will be discussed in-depth in other notes (see the *Additional Notes* section).

6.1 Surrogate Models

Surrogate models are the models we use in Bayesian optimization to express our beliefs and knowledge about the objective function. While in certain literature the term *observation model* is used interchangeably with *surrogate model*, they are distinguishable. For instance, if we have a mathematical formulation of the objective then we refer to the model that relates the observations and the objective as an *observation model*. However, if we do not know the underlying structure of the objective then we refer to the model as a *surrogate model* since it acts as a surrogate and "takes the place" of the objective function. The surrogate model is used within the posterior process to quantify the probabilities of observations in conjunction with utility functions. Therefore, it is a powerful tool when we are reasoning about both the uncertainty and utility of candidate observations.

As mentioned before, we use a *stochastic process* to characterize the objective function and the gold standard in Bayesian optimization is the *Gaussian process*. In fact, one of the most common choices for surrogate models is the Gaussian process due to its flexibility, expressivity, and sufficient uncertainty quantification properties. We will discuss Gaussian processes in greater detail in other notes (see *Additional Notes* section).

6.2 Acquisition Functions

Acquisition functions assign a score to candidate observations where the score represents an observation's potential benefit during optimization. Ideally, acquisition functions are cheap to evaluate and address the *exploitation-exploration tradeoff*. In the context of optimization, *exploitation* refers to sampling where the objective value is expected to be large whereas *exploration* refers to sampling where we are more uncertain about the objective value. Another useful property of an acquisition function is it vanishes at pre-existing observations as there is no sense in sampling twice.

There are two main types of acquisition functions: myopic and look-ahead. Myopic acquisition functions consider only the immediate utility while look-ahead acquisition functions consider longer-term utility. This will be discussed further in the notes covering Bayesian decision theory. Regardless of the type, the optimization policy should be designed to maximize the acquisition function such that the candidate observation with the most potential benefit is selected.

7 References

[1] Roman Garnett. 2023. *Bayesian optimization*. Cambridge University Press, Cambridge, United Kingdom ;

[2] Peng Liu. 2023. *Bayesian optimization : Theory and practice using python*. Apress, New York, NY.

[3] Carl Edward Rasmussen and Christopher K. I Williams. 2019. *Gaussian processes for machine learning*. The MIT Press.