Intelligent Sequential Experimentation

Bandit and Bayesian Optimization

Reading Party of Jun Wang Group

Zhongwei Yu
Siyuan Guo
Menglong Zhang
Yifan Zhang

PREFACE

CONTENTS

NOTATIONS

$\mathbb{R}$	Set of real numbers.
$\mathbb{Z}$	Set of integers.
$\mathbb{N}$	Set of natural numbers.
$A, B, \ldots,$ Z	Random events or constants, unless otherwise specified.
$a, b, \ldots,$ z	Variables or functions, unless otherwise specified.
$\mathbf{a}$, $\mathbf{b}$, $\ldots$, $\mathbf{z}$	Vectors or composite variables, unless otherwise specified.
$\mathbf{A}$, $\mathbf{B}$, $\ldots$, $\mathbf{Z}$	Matrix or tensor.
$\mathcal{A}$, $\mathcal{B}$, $\ldots$, $\mathcal{Z}$	Sets or collections.
$\mathbb{E}[X]$	Expectation (mean) of random variable X.
$\mathbf{Var}(X)$	Variance of random variable X.

Part I

Preliminaries

CHAPTER 1

PROBABILISTIC THEORY

Probabilistic theory is built around the concept of a *random experiment*, which is an action or process whose outcome cannot be known in advance. This outline is designed to systematically cover the core concepts, methods, and theorems of probability theory, suitable for beginners.

1 Fundamentals of Probabilistic Theory

This section lays the groundwork by defining probability. It starts with the idea of a random experiment, defines the set of all possible outcomes (the sample space), and introduces events as subsets of this space. Then it establishes the fundamental rules and axioms that govern the calculation of probabilities.

1.1 Random Phenomena

A random phenomenon is any process or experiment where the individual outcome is uncertain, but the overall pattern of the outcomes becomes predictable over a large number of repetitions.

The key feature is this contrast:

- **Short-Term Unpredictability**: You cannot know for certain what the next outcome will be.
- **Long-Term Pattern**: You can describe the distribution of outcomes if the process is repeated many times.

A classic example is flipping a fair coin. Any single flip is a total mystery—it could be heads, it could be a head or a tail. But you know that after thousands of flips, the proportion of heads will be very close to 50

It is helpful to contrast random phenomena with its opposite:

- **Random Phenomenon**: The outcome is not fixed. Even if you try to replicate the conditions exactly (e.g., flip a coin), the result can change.
- **Deterministic Phenomenon**: The outcome is fixed and predictable given a set of initial conditions. If you drop a ball from a specific height (and ignore air resistance), you can calculate exactly when and how fast it will hit the ground and it will do the same thing every time.

1.2 Events and Operations

Random Experiment

The Random Experiment is designed to observe and analyze the inherent randomness of a phenomenon.

In the field of probability and statistics, an **experiment** is a procedure carried out to support, refute, or validate a hypothesis. When the outcome of an experiment cannot be predicted with certainty, even when performed under the same set of conditions, it is called a **Random Experiment** (also known as a Statistical Experiment). The study of such experiments is fundamental to the entire discipline of probability theory.

A procedure is classified as a random experiment if and only if it satisfies three essential conditions:

1. **Repeatability**: The experiment must be repeatable (or capable of being conceived as repeatable) under essentially identical conditions for an unlimited number of times.
2. **Known Outcomes (The Sample Space)**: The set of all possible outcomes of the experiment, known as the **Sample Space** (Ω or S), must be known in advance.
3. **Uncertain Outcome**: While the set of all possible outcomes is known, the exact outcome of any single trial of the experiment cannot be predicted with certainty.

Sample Space (Ω)

The sample space, denoted by Ω (or S), is the set of all possible outcomes of a random experiment.

- **Example:** When tossing a standard six-sided die, the sample space is:

$$\Omega = \{1, 2, 3, 4, 5, 6\}$$

It can be divided into two types:

- **Discrete Sample Space**: A sample space where the outcomes are countable (finite or countably infinite). Most introductory examples use discrete sample spaces.
- **Continuous Sample Space**: A sample space where the outcomes can be any value in a continuous interval (e.g., measuring temperature or time).

Event ($\mathcal{E}$)

An event, usually denoted by capital letters like A, B, or E, is any subset of the sample space. An event is said to occur if the outcome of the experiment is one of the elements contained in that subset.

- **Mathematical Notation:** An event A is a subset of the sample space:

$$A \subseteq \Omega$$

- **Example:** The event A is "rolling an even number" on the die:

$$A = \{2, 4, 6\}$$

There are two special cases of events that maybe require your attention:

- **The Impossible Event ($\emptyset$)** The impossible event is an event that contains no outcomes from the sample space and can therefore never occur.

$$\emptyset \subset \Omega \quad \text{and} \quad P(\emptyset) = 0$$

- **The Certain Event (Ω)** The certain event is the event that contains all outcomes in the sample space. It is guaranteed to occur whenever the experiment is performed.

$$\Omega \subseteq \Omega \quad \text{and} \quad P(\Omega) = 1$$

Operations on Events

Since events are sets, the fundamental set operations can be applied to combine two or more events to form new ones.

- **Complement (A^c or $\bar{A}$)**

  The complement of an event A is the event that A does not occur. It consists of all outcomes in the sample space Ω that are not in A.

  - **Notation:** A^c or $\bar{A}$ (sometimes A').
  - **Definition:**
  $$A^c = \{x \in \Omega \mid x \notin A\}$$
  - **Probability Rule:**
  $$P(A^c) = 1 - P(A)$$

- **Union ($A \cup B$)**

  The union of two events A and B is the event that *at least one* of the events occurs. It consists of all outcomes that are in A, or in B, or in both.

- **Notation:** $A \cup B$.
- **Definition:**
$$A \cup B = \{x \in \Omega \mid x \in A \text{ or } x \in B\}$$
- **Addition Rule:**
$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

- **Intersection ($A \cap B$)**

  The intersection of two events A and B is the event that *both* A and B occur simultaneously. It consists of all outcomes that are common to both A and B.

  - **Notation:** $A \cap B$ (or sometimes AB).
  - **Definition:**
$$A \cap B = \{x \in \Omega \mid x \in A \text{ and } x \in B\}$$

1.3 Definitions and Properties of Probability

Probability is formally defined as a function P that assigns a number (a probability) to every event A in the event space. This function must satisfy a set of rules known as the Axioms of Probability.

The Axioms of Probability

Formulated by Andrey Kolmogorov, these three axioms are the foundation upon which all probability theory is built. For any event A in the sample space Ω:

1. **Non-negativity**: The probability of any event is non-negative.
$$P(A) \geq 0$$

2. **Normalization**: The probability of the certain event (the entire sample space) is 1.
$$P(\Omega) = 1$$

3. **Additivity**: If two events A and B are mutually exclusive (disjoint), meaning they have no outcomes in common ($A \cap B = \emptyset$), then the probability of their union is the sum of their individual probabilities.
$$P(A \cup B) = P(A) + P(B) \quad \text{if } A \cap B = \emptyset$$
This extends to any sequence of disjoint events $A_1, A_2, \ldots$ (countable additivity):
$$P(A_1 \cup A_2 \cup \ldots) = P(A_1) + P(A_2) + \ldots$$

Derived Properties of Probability

From these three axioms, we can derive several other essential properties:

- **Probability of the Complement**: The probability that an event does not occur is 1 minus the probability that it does.

$$P(A^c) = 1 - P(A)$$

- **Probability of the Impossible Event**: The probability of the impossible event is 0.

$$P(\emptyset) = 0$$

- **Monotonicity**: If event A is a subset of event B ($A \subseteq B$), then the probability of A is less than or equal to the probability of B.

$$P(A) \leq P(B) \quad \text{if } A \subseteq B$$

- **Probability Range**: The probability of any event is always between 0 and 1 (inclusive).

$$0 \leq P(A) \leq 1$$

- **Inclusion-Exclusion Principle (for two events)**: As stated earlier, this is the general addition rule for any two events, not just disjoint ones.

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

2 Conditional Probability and Independence

This section explores how the probability of an event can change based on the knowledge that another event has occurred.

2.1 Conditional Probability

The conditional probability of an event A given that event B has occurred is denoted $P(A|B)$. It represents an update to our belief about A in light of new information B.

- **Definition**: The conditional probability is defined as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \quad \text{provided } P(B) > 0$$

- **Interpretation**: We are restricting our sample space to only the outcomes in B. The probability of A is then the proportion of those outcomes that are also in A.

2.2 The Multiplication Rule

By rearranging the definition of conditional probability, we get the multiplication rule, which is useful for calculating the probability of an intersection:

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

2.3 Independence

Two events A and B are **independent** if the occurrence of one does not affect the probability of the other.

- **Definition**: A and B are independent if and only if:

$$P(A \cap B) = P(A)P(B)$$

- **Relation to Conditional Probability**: If A and B are independent (and $P(B) > 0$):

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A)P(B)}{P(B)} = P(A)$$

  This confirms the intuitive idea that knowing B occurred does not change the probability of A.

2.4 The Law of Total Probability

If we partition the sample space Ω into a set of disjoint events $B_1, B_2, \ldots, B_n$ whose union is Ω, then for any event A:

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i) = \sum_{i=1}^{n} P(A|B_i)P(B_i)$$

This law allows us to find the probability of A by breaking it down into conditional "scenarios."

2.5 Bayes' Theorem

Bayes' Theorem, also known as Bayes' Rule, describes how to update probabilities based on new evidence. It is a direct consequence of the definitions of conditional probability.

$$P(B_i|A) = \frac{P(A|B_i)P(B_i)}{P(A)} = \frac{P(A|B_i)P(B_i)}{\sum_{j=1}^{n} P(A|B_j)P(B_j)}$$

- $P(B_i|A)$ is the **posterior probability**: the updated probability of B_i *after* observing A.
- $P(B_i)$ is the **prior probability**: the initial probability of B_i.
- $P(A|B_i)$ is the **likelihood**: the probability of observing A given B_i.

CHAPTER 2

STOCHASTIC PROCESS

CHAPTER 3

NUMERIC OPTIMIZATION

Part II

Bandit

CHAPTER 1

STOCHASTIC BANDIT

In this chapter, we will introduce the most basic bandit setting, *stochastic bandit*. In each round $t \in [T]$, the algorithm should pull an arm a_t from K arms, and then receives a reward r_{a_t} for this arm. The objective of the algorithm is to maximize the accumulated rewards over T rounds. In this setting, we have three key assumptions:

- The environment only returns the reward of the pulled arm, which we call this setting as *bandit feedback*.
- The reward for each arm is independent and identically distributed (IID). For each arm $a \in [K]$, we have a fixed reward distribution $\mathcal{D}_a$. The reward is independently sampled from the reward distribution of the pulled arm, i.e., $r_{a_t} \sim \mathcal{D}_{a_t}$. The algorithm has no knowledge about the reward distribution of any arm at the very beginning.
- The reward is bounded. Without loss of generality, we assume that the reward lies in the interval $[0, 1]$.

As such, if we can accurately estimate the mean value of the reward distribution for each arm, we can derive the optimal policy by always pulling the arm with the maximum mean value.

To measure whether an algorithm is a good online learner, we introduce the definition of *regret*. Formally, denote $\mu^\star$ as the maximum mean value of all the reward distributions (i.e., $\mu^\star = \arg\max_{a \in [K]} \mu_a$), we define the regret at round T as:

$$R(T) = T \cdot \mu^\star - \sum_{t=1}^{T} \mu_{a_t}. \tag{1.1}$$

We typically concern the *expected regret*, i.e., $\mathbb{E}[R(T)]$. We say an algorithm is a good online learner, if it can achieve *no-regret learning*, which means that the averaged regret of the algorithm goes to zero, when the round goes to infinity. Formally, we have:

$$\lim_{T \to \infty} \frac{R(T)}{T} = 0. \tag{1.2}$$

If an algorithm can satisfy the equation above, we also say that it has *sub-linear regret*. Now, we will introduce several naive algorithms to give some quick insights about the stochastic bandit setting.

1 Exploration-First Algorithm

To begin with, we introduce a naive algorithm Exploration-First with a very simple idea. As we only need to accurately estimate the mean value of each reward distribution, we can pull each arm N times to calculate the empirical means of the reward distribution. Then, we always pull the arm with the maximum empirical mean value. We provide the algorithm as below.

Algorithm 1 Exploration-First Algorithm

Require: Exploration budget N

```
for t = 1, ..., T do
    if t ≤ NK then
        randomly pull an arm that has been not pulled N times
    else
        pull the arm with the maximum empirical mean value (break ties arbitrarily)
    end if
end for
```

Now, we try to analyze the regret bound for this naive algorithm.

Let $a^\star \in [K]$ be an optimal arm with mean $\mu^\star = \mu_{a^\star}$. For each arm a, let the gap be $\Delta_a := \mu^\star - \mu_a \in [0, 1]$, with $\Delta_{a^\star} = 0$ and $\Delta_{\min} := \min_{a \neq a^\star} \Delta_a$. Then, a standard decomposition is

$$\mathbb{E}[R(T)] = \sum_{a=1}^{K} \Delta_a \cdot \mathbb{E}[N_a(T)], \tag{1.3}$$

where $N_a(T)$ is the number of pulls of arm a up to time T.

For Exploration-First with budget N, we first focus on the exploration phase, i.e., $t \leq NK$, where each arm is pulled exactly N times, so this contributes

$$R_{\text{explore}} = \sum_{a \neq a^\star} N\Delta_a. \tag{1.4}$$

Next, we analyze the exploitation phase, i.e., $t > NK$. Let $\widehat{\mu}_a$ be the empirical mean of arm a from its N exploratory pulls, and let $\tilde{a} \in \arg\max_a \widehat{\mu}_a$ be the selected arm for exploitation. Then, we have

$$R_{\text{exploit}} = (T - NK)\mathbb{E}[\Delta_{\tilde{a}}] \tag{1.5}$$

$$= (T - NK) \sum_{a \neq a^\star} \Delta_a \Pr(\tilde{a} = a). \tag{1.6}$$

To further bounding this term, we only need to bound $\Pr(\tilde{a} = a)$. For any suboptimal arm $a \neq a^\star$, we have

$$\Pr(\tilde{a} = a) = \Pr\left(\widehat{\mu}_a \geq \widehat{\mu}_{a^\star}\right) \tag{1.7}$$
$$\leq \Pr\left(\widehat{\mu}_{a^\star} \leq \mu^\star - \tfrac{\Delta_a}{2}\right) + \Pr\left(\widehat{\mu}_a \geq \mu_a + \tfrac{\Delta_a}{2}\right). \tag{1.8}$$

The inequality is due to the event relationship $\{\hat{\mu}_a \geq \hat{\mu}_{a^\star}\} \subseteq \{\hat{\mu}_a \geq \mu^\star - \frac{\Delta_a}{2}\} \cup \{\hat{\mu}_{a^\star} \leq \mu^\star - \frac{\Delta_a}{2}\}$. Then, by Hoeffding's inequality, we have

$$\Pr(\tilde{a} = a) \leq 2\exp\left(-2N\left(\tfrac{\Delta_a}{2}\right)^2\right) \tag{1.9}$$
$$= 2\exp\left(-\tfrac{N\Delta_a^2}{2}\right). \tag{1.10}$$

Putting them together, we have the regret bound as

$$\mathbb{E}[R(T)] \leq N\sum_{a\neq a^\star}\Delta_a + 2(T - NK)\sum_{a\neq a^\star}\Delta_a \exp\left(-\tfrac{N\Delta_a^2}{2}\right) \tag{1.11}$$

We can find that this regret bound leads to a linear regret. However, we can derive a sub-linear regret by choosing an appropriate value of the hyper-parameter N as

$$N = \left\lceil \frac{2}{\Delta_{\min}^2}, \log T \right\rceil. \tag{1.12}$$

As such, for each sub-optimal arm $a \neq a^\star$, we have

$$\exp\left(-\tfrac{N\Delta_a^2}{2}\right) \leq \exp\left(-\log T \cdot \tfrac{\Delta_a^2}{\Delta_{\min}^2}\right) \tag{1.13}$$
$$\leq T^{-1}, \tag{1.14}$$

and the exploitation term is at most $2\sum_{a\neq a^\star}\Delta_a$ (a constant in T). Hence,

$$\mathbb{E}[R(T)] \leq \frac{2\log T}{\Delta_{\min}^2}\sum_{a\neq a^\star}\Delta_a + O\left(\sum_{a\neq a^\star}\Delta_a\right) \tag{1.15}$$
$$= O\left(\frac{K\log T}{\Delta_{\min}^2}\right). \tag{1.16}$$

This is sublinear in T, so the algorithm achieves no-regret learning with an appropriate $N = \Theta(\log T)$.

CHAPTER 2

MULTI-ARM BANDIT

CHAPTER 3

CONTEXTUAL BANDIT

CHAPTER 4

INFINITELY MANY-ARMED BANDIT

CHAPTER 5

ADVERSARIAL BANDIT

CHAPTER 6

COMBINATORIAL BANDIT

CHAPTER 7

CONVEX BANDIT

CHAPTER 8

BAYESIAN BANDIT

Part III

Bayesian Optimization

CHAPTER 1

GENERAL INTRODUCTION TO BAYESIAN OPTIMIZATION

The objective of Bayesian Optimization (BO) is to find the optimum of a black-box function $f : \mathcal{X} \to \mathbb{R}$ using as few evaluations as possible. BO is closely related to stochastic bandits: both face the exploration–exploitation trade-off and use sequential experimentation to discover high-performing inputs. Key practical distinctions include:

- BO explicitly models prior properties about the objective (e.g., smoothness or structure) via a probabilistic prior, improving sample efficiency for expensive evaluations.
- BO typically targets problems where individual evaluations are costly (time, money, computation), whereas some bandit settings focus on very many cheap pulls.
- Performance in BO is often measured by the quality of a final recommended solution (simple regret), while bandit algorithms generally use cumulative regret.

The central idea of BO is to treat the unknown objective f as a random element in a hypothesis space, to infer a posterior distribution over that element from observed data, and to make sequential decisions about where to evaluate next based on the posterior. This chapter gives a concise conceptual foundation for the general workflow and key components of BO. Implementation details and specific algorithms (Gaussian processes, EI/UCB, regret bounds, scalable approximations) are deferred to later chapters.

1 Bayesian Modeling

Bayesian modeling infers a distribution over candidate models rather than a single point estimate. Let f denote a candidate model (a mapping from inputs to outputs) and let $\mathcal{F}$ be the hypothesis space. We observe data

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n, \qquad \mathbf{x} = (x_1, \ldots, x_n), \quad \mathbf{y} = (y_1, \ldots, y_n).$$

The goal of Bayesian modeling is to obtain the posterior distribution $p(f \mid \mathcal{D})$ that quantifies uncertainty about f after seeing $\mathcal{D}$. A careful Bayesian model specifies both a prior $p(f)$ that encodes beliefs about the inherent properties of f and a likelihood model $p(y \mid f, x)$ that describes how observations are produced from the model.

Bayesian modeling infers the posterior distribution of f from observed data. Starting from the joint distribution, Bayes' rule gives

$$p(f \mid \mathcal{D}) = \frac{p(f, \mathcal{D})}{p(\mathcal{D})}.$$

By factorizing the numerator, we write

$$p(f, \mathcal{D}) = p(\mathbf{y} \mid f, \mathbf{x})\, p(f, \mathbf{x}).$$

If $\mathbf{x}$ is independent of f a priori then conditioning on $\mathbf{x}$ gives

$$p(f \mid \mathcal{D}) = \frac{p(\mathbf{y} \mid f, \mathbf{x})\, p(f)}{p(\mathbf{y} \mid \mathbf{x})} \propto p(\mathbf{y} \mid f, \mathbf{x})\, p(f). \tag{1.1}$$

The marginal likelihood (evidence) in the denominator is

$$p(\mathbf{y} \mid \mathbf{x}) = \int_{\mathcal{F}} p(\mathbf{y} \mid f, \mathbf{x})\, p(f)\, \mathrm{d}f,$$

which normalizes the posterior.

In practice we often work with summaries of the posterior (point estimates, moments) or with the posterior predictive distribution for function values at candidate inputs. Exact posterior computation is frequently intractable for complex models; approximations (Laplace, variational inference, sampling) are commonly used and will be discussed in dedicated chapters.

2 Optimization by Sequential Decision Making

Optimization algorithms, in general, operate by iteratively selecting evaluation points, observing outcomes, and updating the dataset. Formally, let the objective be f and let $\mathrm{obs}(f, x)$ denote the observation mechanism that produces data from f. Define the data at time t by $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{t}$. A general sequential framework is given by the following algorithm.

Regret In the context of sequential optimization and bandits, *regret* quantifies the performance loss due to not selecting the global optimum at each step. Two common notions are:

- **Simple regret** r_t: the difference between the optimum value and the value at the recommended point after t evaluations,

  $$r_t = f(x^\star) - f(\hat{x}_t),$$

  where $x^\star = \arg\max_{x \in \mathcal{X}} f(x)$ and $\hat{x}_t$ is the recommended solution at time t. Simple regret measures the quality of the final recommendation and is commonly used in BO.

Algorithm 2 General sequential optimization framework

Require: Domain $\mathcal{X}$, budget T, policy sequence $\pi = (\pi_n, \pi_{n+1}, \dots)$ with $\pi_t : \mathcal{D}_t \to \mathcal{X}$, observation mechanism $\text{obs}(f, x)$

Initialize dataset $\mathcal{D}_n$ (possibly empty or from a space-filling design)
for $t = n,\ n+1,\ \dots,\ T-1$ **do**
  Select next input: $x_{t+1} \leftarrow \pi_t(\mathcal{D}_t)$
  Observe outcome: $y_{t+1} \leftarrow \text{obs}(f, x_{t+1})$
  Augment data: $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup \{(x_{t+1}, y_{t+1})\}$
end for
return Recommended solution $\hat{x}_T$ {e.g., best observed point in $\mathcal{D}_T$ or following π_T}

- **Cumulative regret** R_T: the sum of instantaneous regrets over the evaluation sequence,

$$R_T = \sum_{t=1}^{T} \left(f(x^\star) - f(x_t) \right),$$

  where x_t is the point evaluated at iteration t. Cumulative regret is often the focus in bandit problems, reflecting total loss during exploration.

The distinction reflects differing goals: BO emphasizes finding a single high-quality solution efficiently (minimizing simple regret), while bandit algorithms often aim to maximize reward throughout the process (minimizing cumulative regret).

3 Key Components of Bayesian Optimization

Bayesian Optimization consists of two main components: the surrogate model and the acquisition function. These work together to balance exploration and exploitation in the search for the optimum.

3.1 Surrogate Model

The surrogate model in Bayesian optimization is the core component that approximates the posterior distribution over the unknown objective function f, given observed data $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^{T}$. Here, x_t is the input tested at step t, and $y_t \sim \text{obs}(f, x_t)$ is the corresponding observation at x_t. This is achieved from Bayesian modeling as introduced in Section 1, using a pre-defined prior $p(f)$. A dominant choice is the Gaussian processes, which we will introduce in Chapter 2.

Through the surrogate model, we can efficiently estimates y (including its uncertainty) for any input x, conditioned on the evidence in past observations $\mathcal{D}$. Note that this chapter only provides a conceptual introduction only. Detailed discussion of surrogate models, including Gaussian processes and other approaches, is presented in Chapter 3.

3.2 Acquisition Function

BO introduces the acquisition function (AF) to determine the next evaluation point, which defines the decision policy π in the sequential optimization framework. An acquisition function $\alpha(x \mid \mathcal{D})$ is a computationally tractable scalar criterion, derived from the posterior surrogate, that ranks candidate inputs for the next evaluation. The next query point is selected by maximizing the acquisition function:

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \alpha(x \mid \mathcal{D}_t).$$

Acquisition functions can be motivated as approximations to the posterior expected utility and typically balance exploration (sampling uncertain or less-explored regions) and exploitation (sampling near known optima). Essentially, the AF transforms the unknown expensive objective f into a temporary but cheap and known surrogate objective. Specific acquisition functions, their properties, and optimization methods are treated in later chapters.

4 General BO Algorithm

Here, we present a compact description of the BO loop at a conceptual level. In each iteration of the main BO loop, we update the acquisition function $\alpha(x \mid \mathcal{D}_t)$ based on the current posterior surrogate, select the next evaluation point x_{t+1} by maximizing α, observe the outcome y_{t+1}, augment the dataset $\mathcal{D}_t$ to form $\mathcal{D}_{t+1}$, and update the posterior. When the budget is exhausted, BO returns a recommended solution $\hat{x}_T$ according to a preset rule. The core idea is iteratively refining beliefs (via posteriors) to select informative points, optimizing performance within the evaluation budget.

Initialization At the start of BO, an initialization procedure is often included to create an initial dataset $\mathcal{D}_n$ with preliminary observations. The initial testing points can be chosen randomly or via a specific strategy such as space-filling designs. This "warm-up" stage helps establish a reliable surrogate model before entering the main optimization loop. However, it is also possible to start with an *empty* initial dataset.

5 Summary

This chapter provided a conceptual foundation for Bayesian optimization:

- BO treats the unknown objective as a random element and uses the *surrogate model* to obtain a posterior over functions given observations.
- Acquisition functions are tractable proxies that drive the sequential selection of evaluation points.

Algorithm 3 Bayesian Optimization

Require: Domain $\mathcal{X}$, budget T, prior $p(f)$, likelihood $p(y \mid f(x))$, acquisition principle

Initialize: collect initial dataset $\mathcal{D}_n$
for $t = n, \ldots, T-1$ **do**
  Compute surrogate $\hat{f}(x|\mathcal{D}_t)$
  Construct acquisition $\alpha(x \mid \mathcal{D}_t)$ from $\hat{f}$
  Select next point: $x_{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} \alpha(x \mid \mathcal{D}_t)$
  Observe: $y_{t+1} \leftarrow \text{obs}(f, x_{t+1})$
  Augment data: $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup \{(x_{t+1}, y_{t+1})\}$
end for
return Recommended solution $\hat{x}_T$ (per chosen recommendation rule)

- The BO loop alternates between acquisition optimization, function evaluation, and surrogate updating until the evaluation budget is exhausted.

CHAPTER 2

GAUSSIAN PROCESS

Gaussian processes (GPs) are powerful probabilistic tools to model unknown black-box functions and the most widely used surrogate models in Bayesian optimization (BO). In BO, the GP provides a flexible, nonparametric prior over functions, allowing us to capture uncertainty and incorporate prior beliefs such as smoothness or structure. This uncertainty quantification is crucial for guiding the sequential selection of evaluation points via acquisition functions, effectively balancing exploration and exploitation. Compared to other surrogate models, GPs offer analytical tractability: given observed data, the posterior distribution remains a Gaussian process with closed-form expressions for the mean and covariance. This enables efficient updating and prediction, even with limited data, making GPs particularly suitable for expensive-to-evaluate objectives common in BO.

This chapter provides a comprehensive introduction to GPs. We begin with the fundamental definitions and interpretations of GPs, followed by their mathematical characterization both individually and jointly with other random variables or processes. Next, we discuss practical aspects such as sampling from GPs and inference under various observation models, including noisy and non-Gaussian settings. We then explore key properties of GPs, including differentiability and behavior of sample paths. Finally, we cover modeling considerations, focusing on the choice and construction of mean and kernel functions, their properties, and strategies for handling high-dimensional inputs. Together, these topics establish the theoretical and practical foundation necessary to effectively apply Gaussian processes as surrogates within Bayesian optimization frameworks.

1 Basic Concept

We cite a general and concise definition by Rasmussen and Williams [1]:

Definition 1.1: Gaussian Process

A *Gaussian Process* (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution.

1.1 Interpretations

Definition 1.1 provides a general notion of a GP by specifying the property of its finite subsets, without explicitly describing how the random variables are organized (indexed) within this "collection." This vagueness allows the concept of GPs to be highly versatile and applicable across many different fields and problems. Since GPs can model functions, time series, spatial data, or any indexed family of random variables, a more concrete formulation could limit their generality and usefulness in diverse applications.

Set Interpretation In the most basic interpretation, a GP is simply a set of random variables without any assumed ordering or structure on the indices. Here, the set $\mathcal{X}$ is an arbitrary set with no inherent order or topology. The GP is a family $\{X_x : x \in \mathcal{X}\}$ where each X_x is a random variable, and any finite subset $\{X_{x_1}, \ldots, X_{x_n}\}$ has a joint Gaussian distribution. This viewpoint emphasizes the collection aspect but does not assume any further structure such as ordering or geometry.

Stochastic Process Interpretation A classical stochastic process is a collection of random variables indexed by time, often represented as a sequence $\{X_1, X_2, \ldots\}$ or a continuous-time family $\{X_t : t \in \mathbb{R}_+\}$. In this interpretation, the index set is ordered (discrete or continuous), which allows notions such as causality and filtration. A Gaussian process in this context is a stochastic process where any finite collection of variables has a joint Gaussian distribution.

Spatial Interpretation In spatial statistics, the index set $\mathcal{X}$ is taken to be a subset of Euclidean space $\mathbb{R}^d$. The random variables $\{X_x : x \in \mathcal{X}\}$ are associated with spatial locations, and the covariance structure captures spatial correlation or smoothness. This interpretation is common in geostatistics and spatial modeling, where the geometry of the index set is essential.

Functional Interpretation The most general interpretation views a Gaussian process as a *random function* defined on an arbitrary domain $\mathcal{X}$, which may be any set with or without additional structure. Here, the GP defines a distribution over functions $f : \mathcal{X} \to \mathbb{R}$, characterized by a mean function and a covariance kernel. This perspective subsumes all previous ones and is fundamental in modern applications such as machine learning and functional data analysis.

1.2 Characterization

In the context of Bayesian optimization, the GP is interpreted as a random function to model the surrogate of a black-box objective. In this section, we introduce how to mathematically describe a GP—individually or jointly with other variables.

Individual Characterization of a Gaussian Process

Let us consider describing a GP individually, without considering its correlations with other random variables or functions in the system. Naturally, a GP on domain $\mathcal{X}$ is characterized by a mean function $m(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$ and a covariance function $k(\mathbf{x}, \mathbf{x}') : \mathcal{X}^2 \to \mathbb{R}$. The covariance function k is also referred to as the *kernel* function, which is symmetric, i.e., $k(x, x') = k(x', x)$. Then, a random function f drawn from a GP is fully characterized by m and k, written as

$$f \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big). \tag{2.1}$$

For any n points $x_1, \ldots, x_n$ in $\mathcal{X}$, their joint Gaussian distribution is given by:

$$\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(x_1) \\ \vdots \\ m(x_n) \end{pmatrix}, \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{pmatrix} \right). \tag{2.2}$$

Thus, a GP generalizes the notion of a multivariate Gaussian distribution to an infinite-dimensional function space. The kernel function encodes assumptions about smoothness, periodicity, or other structural properties of the unknown function.

Joint Characterization with a Multivariate Gaussian Vector

Let $f \sim \mathcal{GP}(m, k)$ be a Gaussian process on the domain $\mathcal{X}$, and let $\mathbf{z} = (z_1, \ldots, z_n)^\top \in \mathbb{R}^n$ be any random vector satisfying a multivariate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$.

For any finite set of points $\{x_1, \ldots, x_N\} \subset \mathcal{X}$, define the random variables $y_i = f(x_i)$ for $i = 1, \ldots, N$. The joint distribution of $(y_1, \ldots, y_N, z_1, \ldots, z_n)$ is multivariate Gaussian.

To express this, we introduce the cross-covariance function $\kappa : \mathcal{X} \to \mathbb{R}^n$, where $\kappa_i(x) = \mathrm{Cov}(f(x), z_i)$ gives the covariance with the i-th component of $\mathbf{z}$. We use $\kappa^\top : \mathcal{X} \to \mathbb{R}^{1 \times n}$ to denote the transpose of κ, so $\kappa^\top(x) = (\kappa(x))^\top$. This function describes the covariance between the GP $f(x)$ at any point x and the components z_i of the random vector $\mathbf{z}$.

Finally, we can extend the GP notation using block matrix form as the following GP-Normal joint distribution:

$$\begin{bmatrix} f(x) \\ \mathbf{z} \end{bmatrix} \sim \begin{bmatrix} \mathcal{GP} \\ \mathcal{N} \end{bmatrix} \left(\begin{pmatrix} m(x) \\ \mu \end{pmatrix}, \begin{pmatrix} k(x, x') & \kappa^\top(x) \\ \kappa(x) & \Sigma \end{pmatrix} \right). \tag{2.3}$$

For a finite collection $\mathbf{x} = \{x_1, \ldots, x_N\}$, the block $K_{f\mathbf{z}}$ is obtained by evaluating κ at the points in $\mathbf{x}$. That is, the joint distribution of the function values $y_i = f(x_i)$ for $i = 1, \ldots, N$ and the random vector $\mathbf{z} = (z_1, \ldots, z_n)^\top$ can be expressed as a multivariate Gaussian distribution:

$$\begin{pmatrix} \mathbf{y} \\ \mathbf{z} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu} \end{pmatrix}, \begin{pmatrix} \mathbf{K} & \mathbf{K}_\mathbf{z}^\top \\ \mathbf{K}_\mathbf{z} & \Sigma \end{pmatrix} \right), \tag{2.4}$$

where:

- $\mathbf{m} = (m(x_1), \ldots, m(x_N))^\top$ is the mean vector for the function values;

- $\boldsymbol{\mu}$ is the mean vector of the random vector $\mathbf{z}$;
- $\mathbf{K}$ is the kernel matrix of the GP, with $(\mathbf{K})_{ij} = k(x_i, x_j)$;
- $\mathbf{K}_\mathbf{z}$ is the cross-covariance matrix, where the i-th row $(\mathbf{K}_\mathbf{z})_{i*}$ is $\kappa^\top(x_i)$;
- and Σ is the covariance matrix of the random vector $\mathbf{z}$.

Joint Characterization with Another Gaussian Process

Let $f \sim \mathcal{GP}(m_f, k_f)$ be a Gaussian process on $\mathcal{X}$ and $g \sim \mathcal{GP}(m_g, k_g)$ be a Gaussian process on $\mathcal{Y}$. Introduce the cross-covariance functions

$$k_{fg} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}, \qquad k_{fg}(x, y) = \mathrm{Cov}\big(f(x),\, g(y)\big), \tag{2.5}$$

and

$$k_{gf} : \mathcal{Y} \times \mathcal{X} \to \mathbb{R}, \qquad k_{gf}(y, x) = \mathrm{Cov}\big(g(y),\, f(x)\big). \tag{2.6}$$

By properties of covariance, we have $k_{gf}(y, x) = k_{fg}(x, y)$.

Equivalently, we may view the pair (f, g) as a vector-valued Gaussian process:

$$\begin{pmatrix} f(\cdot) \\ g(\cdot) \end{pmatrix} \sim \mathcal{GP}\left(\begin{pmatrix} m_f(\cdot) \\ m_g(\cdot) \end{pmatrix}, \begin{pmatrix} k_f(\cdot,\cdot) & k_{fg}(\cdot,\cdot) \\ k_{gf}(\cdot,\cdot) & k_g(\cdot,\cdot) \end{pmatrix} \right). \tag{2.7}$$

Given any finite collections $\mathbf{x} = \{x_1, \ldots, x_N\} \subset \mathcal{X}$ and $\mathbf{y} = \{y_1, \ldots, y_M\} \subset \mathcal{Y}$, the stacked vector $\mathbf{f} = [f(x_1) \ \cdots \ f(x_N)]^\top$ and $\mathbf{g} = [g(y_1) \ \cdots \ g(y_M)]^\top$ is multivariate Gaussian:

$$\begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mathbf{m}_f \\ \mathbf{m}_g \end{pmatrix}, \begin{pmatrix} \mathbf{K}_f & \mathbf{K}_{fg} \\ \mathbf{K}_{gf} & \mathbf{K}_g \end{pmatrix} \right), \tag{2.8}$$

where:

- $\mathbf{m}_f = (m_f(x_1), \ldots, m_f(x_N))^\top$ is the mean vector for the function values of the Gaussian process f;
- $\mathbf{m}_g = (m_g(y_1), \ldots, m_g(y_M))^\top$ is the mean vector for the function values of the Gaussian process g;
- $\mathbf{K}_f$ is the kernel matrix of the Gaussian process f, with $(\mathbf{K}_f)_{ij} = k_f(x_i, x_j)$;
- $\mathbf{K}_g$ is the kernel matrix of the Gaussian process g, with $(\mathbf{K}_g)_{ij} = k_g(y_i, y_j)$;
- $\mathbf{K}_{fg}$ is the cross-covariance matrix between f and g, with the (i, j)-th entry $k_{fg}(x_i, y_j)$; and $\mathbf{K}_{gf} = \mathbf{K}_{fg}^\top$.

Joint/Multi-output Gaussian Processes*

Above, we have used the *function matrix* to characterize the joint distribution of two Gaussian processes f and g. In this section, we extend this to the joint distribution of any M Gaussian processes, also called a *multi-output Gaussian process* (MOGP). An MOGP describes the

distribution of a *random vector function* $\boldsymbol{f} = (f_1, \ldots, f_M) : \mathcal{X} \to \mathbb{R}^M$ that outputs an M-dimensional vector, where each component f_i is itself a GP.

Similar to a GP, we can characterize an MOGP by specifying its vector-valued mean function $\boldsymbol{m} : \mathcal{X} \to \mathbb{R}^M$ and a matrix-valued kernel (covariance) function $\boldsymbol{K} : \mathcal{X}^2 \to \mathbb{R}^{M\times M}$, and write

$$\boldsymbol{f} \sim \mathcal{GP}\big(\boldsymbol{m}(x), \boldsymbol{K}(x, x')\big), \tag{2.9}$$

where $\boldsymbol{K}(x, x') = \boldsymbol{K}^\top(x, x') = \boldsymbol{K}(x', x)$. Note that we use **bold** symbols for $\boldsymbol{f}$, $\boldsymbol{m}$, and $\boldsymbol{K}$ to highlight that their outputs are composite values (not scalars). However, it is fine to simplify notations as $f \sim \mathcal{GP}(m, K)$ provided the context is clear.

To further understand the above characterization, let $\mathcal{X}_N = \{x_1, \ldots, x_N\}$ be a finite collection of N input points and define the stacked random vector

$$\mathbf{f}_{1:M,\,1:N} = \big(f_1(x_1), \ldots, f_1(x_N),\ f_2(x_1), \ldots, f_M(x_N)\big)^\top \in \mathbb{R}^{MN}.$$

Then, by the definition of an MOGP, this vector follows a multivariate Gaussian distribution:

$$\mathbf{f}_{1:M,\,1:N} \sim \mathcal{N}\big(\boldsymbol{\mu}_{1:M,\,1:N},\, \mathbf{K}_{1:M,\,1:N}\big)\,, \tag{2.10}$$

where the mean vector is obtained by evaluating $\boldsymbol{m}$ at each input,

$$\boldsymbol{\mu}_{1:M,\,1:N} = \big(\boldsymbol{m}(x_1)^\top, \ldots, \boldsymbol{m}(x_N)^\top\big)^\top,$$

and the covariance matrix $\mathbf{K}_{1:M,\,1:N} \in \mathbb{R}^{MN\times MN}$ has an $M \times M$ block structure:

$$\mathbf{K}_{1:M,\,1:N} = \begin{pmatrix} \boldsymbol{K}(x_1, x_1) & \cdots & \boldsymbol{K}(x_1, x_N) \\ \vdots & \ddots & \vdots \\ \boldsymbol{K}(x_N, x_1) & \cdots & \boldsymbol{K}(x_N, x_N) \end{pmatrix}.$$

Each block $\boldsymbol{K}(x_i, x_j)$ encodes both the *inter-output correlations* between $f_1, \ldots, f_M$ and the *spatial correlations* between input locations x_i and x_j. This structure ensures that any finite subset of outputs and inputs of an MOGP jointly follow a consistent multivariate Gaussian distribution, generalizing the scalar-output GP to the multi-output setting.

2 Sampling

Sampling from a GP allows us to generate realizations (sample functions) consistent with the specified mean and covariance functions, with applications in visualization or implementing acquisition strategies in Bayesian optimization such as Thompson sampling. Depending on the number of points and the desired resolution, sampling can be performed at single points, finite collections of points, or over an entire domain.

2.1 Sampling at Finite Points

Sampling at a Single Point Sampling the GP at a single point $x \in \mathcal{X}$ is straightforward. Since $f(x)$ is a univariate Gaussian random variable with mean $m(x)$ and variance $k(x, x)$, a sample can be drawn as:

$$f(x) \sim \mathcal{N}\big(m(x), k(x, x)\big). \tag{2.11}$$

Sampling at Multiple Points When sampling at multiple points $\mathbf{x} = \{x_1, \ldots, x_n\} \subset \mathcal{X}$, the function values $\mathbf{f} = (f(x_1), \ldots, f(x_n))^\top$ follow a multivariate Gaussian distribution:

$$\mathbf{f} \sim \mathcal{N}\big(\mathbf{m}, \mathbf{K}\big), \tag{2.12}$$

where

$$\mathbf{m} = \big(m(x_1), \ldots, m(x_n)\big)^\top, \quad \mathbf{K} = \big(k(x_i, x_j)\big)_{i,j=1}^{n}. \tag{2.13}$$

Sampling from this distribution can be done using standard methods such as Cholesky decomposition of $\mathbf{K}$ to generate correlated Gaussian vectors.

2.2 Sampling an Entire Function

Sampling an entire function from a GP is conceptually more challenging since it involves an infinite-dimensional object when $\mathcal{X}$ is a continuous subset of Euclidean space. In practice, this is approximated by sampling the function values on a sufficiently dense grid $\{x_1, x_2, \ldots, x_N\}$ covering the domain $\mathcal{X}$. We then draw a multivariate Gaussian sample $\mathbf{f}$ at these points as described above.

To obtain a continuous function approximation from these samples, interpolation methods (e.g., linear interpolation, spline interpolation) or smoothing techniques can be applied to the discrete samples. This results in a smooth function that approximates a sample drawn from the GP.

Note that the accuracy of this approximation depends on the density of the grid and the smoothness properties encoded by the kernel function. Denser grids and smoother kernels yield better approximations of the true GP sample paths.

3 Inference

In this section, we introduce how to infer a GP conditioned on evidence, which may be exact observation data, perturbed observation data, or any random variables correlated with the GP.

3.1 Inference from Gaussian Evidence

We consider a Gaussian process $f \sim \mathcal{GP}(m, k)$ defined on a domain $\mathcal{X}$, and an observed random vector $\mathbf{z} \in \mathbb{R}^n$ that is jointly Gaussian and correlated with f. Following Section 1.2, assume $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, and the joint distribution of f and $\mathbf{z}$ admits the following block structure:

$$\begin{bmatrix} f(x) \\ \mathbf{z} \end{bmatrix} \sim \begin{bmatrix} \mathcal{GP} \\ \mathcal{N} \end{bmatrix} \left(\begin{pmatrix} m(x) \\ \boldsymbol{\mu} \end{pmatrix}, \begin{pmatrix} k(x, x') & \kappa^\top(x) \\ \kappa(x') & \Sigma \end{pmatrix} \right), \tag{2.14}$$

where $\kappa : \mathcal{X} \to \mathbb{R}^n$ is the cross-covariance function defined by $\kappa_i(x) = \mathrm{Cov}(f(x), z_i)$.

Derivation To characterize the posterior process $f \mid \mathbf{z}$, we proceed by finite-dimensional conditioning and then extend to the infinite-dimensional setting.

1. Fix an arbitrary finite collection of points $\mathbf{x} = \{x_1, \ldots, x_N\} \subset \mathcal{X}$, and define the vector of function values
$$\mathbf{f} := \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_N) \end{pmatrix}.$$

2. By evaluating the joint distribution (2.14) at $\mathbf{x}$, the finite-dimensional joint distribution of $(\mathbf{f}, \mathbf{z})$ is Gaussian:
$$\begin{pmatrix} \mathbf{f} \\ \mathbf{z} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mathbf{m} \\ \boldsymbol{\mu} \end{pmatrix}, \begin{pmatrix} \mathbf{K} & \mathbf{K}_{\mathbf{z}}^\top \\ \mathbf{K}_{\mathbf{z}} & \Sigma \end{pmatrix} \right),$$
where
$$\mathbf{m} = \begin{pmatrix} m(x_1) \\ \vdots \\ m(x_N) \end{pmatrix}, \quad \mathbf{K} = \big(k(x_i, x_j)\big)_{i,j=1}^N, \quad \mathbf{K}_{\mathbf{z}} = \begin{pmatrix} \kappa^\top(x_1) \\ \vdots \\ \kappa^\top(x_N) \end{pmatrix} \in \mathbb{R}^{N \times n}.$$

3. Conditioning on $\mathbf{z}$, the conditional distribution of $\mathbf{f}$ is Gaussian with mean and covariance given by the standard formula for conditional Gaussian distributions:
$$\mathbf{m}_{\mathbf{z}} := \mathbf{m} + \mathbf{K}_{\mathbf{z}} \Sigma^{-1} (\mathbf{z} - \boldsymbol{\mu}), \tag{2.15}$$
$$\mathbf{K}_{\mathbf{z}} := \mathbf{K} - \mathbf{K}_{\mathbf{z}} \Sigma^{-1} \mathbf{K}_{\mathbf{z}}^\top. \tag{2.16}$$

4. Since the above holds for any finite subset $\mathbf{x}$, the collection of finite-dimensional conditional distributions is consistent and defines a Gaussian process $f \mid \mathbf{z}$ with mean function and covariance kernel:
$$m_{\mathbf{z}}(x) := m(x) + \kappa^\top(x) \Sigma^{-1} (\mathbf{z} - \boldsymbol{\mu}), \tag{2.17}$$
$$k_{\mathbf{z}}(x, x') := k(x, x') - \kappa^\top(x) \Sigma^{-1} \kappa(x'). \tag{2.18}$$

So far, we have derived the exact characterization of f's distribution conditional on the evidence $\mathbf{z}$. To summarize, we state the following theorem.

Theorem 3.1: GP Conditioned on Gaussian Evidence

Let $f \sim \mathcal{GP}(m, k)$ be a Gaussian process on $\mathcal{X}$, and $\mathbf{z} \in \mathbb{R}^n$ be a Gaussian random vector with distribution $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, jointly distributed with f as in (2.14). Then, the conditional process $f \mid \mathbf{z}$ is a Gaussian process $f \mid \mathbf{z} \sim \mathcal{GP}(m_{\mathbf{z}}, k_{\mathbf{z}})$, where the mean function is
$$m_{\mathbf{z}}(x) = m(x) + \kappa^\top(x) \Sigma^{-1} (\mathbf{z} - \boldsymbol{\mu}),$$
and the covariance function is
$$k_{\mathbf{z}}(x, x') = k(x, x') - \kappa^\top(x) \Sigma^{-1} \kappa(x').$$

3.2 Inference from Observations with Additive Gaussian Noise

Setup Let $f \sim \mathcal{GP}(m, k)$ be a Gaussian process defined on $\mathcal{X}$. Consider a finite collection of input points $\mathbf{x} = \{x_1, \dots, x_N\} \subset \mathcal{X}$. We denote the vector of latent function values at these points by

$$\mathbf{f} := \big(f(x_1), \dots, f(x_N)\big)^\top,$$

with mean vector and covariance matrix

$$\mathbf{m} := \big(m(x_1), \dots, m(x_N)\big)^\top, \quad \mathbf{K} := \big(k(x_i, x_j)\big)_{i,j=1}^N.$$

Suppose we observe noisy measurements $\mathbf{y} \in \mathbb{R}^N$ of the latent function values corrupted by additive Gaussian noise:

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \dots, N,$$

where $\boldsymbol{\epsilon} = (\epsilon_1, \dots, \epsilon_N)^\top$ is a zero-mean Gaussian noise vector with known covariance matrix $\mathbf{N} = \mathrm{Cov}(\boldsymbol{\epsilon})$. We assume $\boldsymbol{\epsilon}$ is independent of f. For instance, the noise ϵ_i may be produced by a random function ϵ such that $\epsilon_i = \epsilon(x_i)$, and ϵ is also a GP that is independent of f.

Joint distribution of $f(x)$ and $\mathbf{y}$. For an arbitrary test point $x \in \mathcal{X}$, define the cross-covariance vector

$$\boldsymbol{\kappa}(x) := \big(k(x, x_1), \dots, k(x, x_N)\big)^\top.$$

Since the noise is independent of f, the covariance between $f(x)$ and $\mathbf{y}$ equals that between $f(x)$ and $\mathbf{f}$:

$$\mathrm{Cov}(f(x), \mathbf{y}) = \mathrm{Cov}(f(x), \mathbf{f}) = \boldsymbol{\kappa}(x).$$

The joint distribution of $f(x)$ and $\mathbf{y}$ is thus Gaussian:

$$\begin{pmatrix} f(x) \\ \mathbf{y} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(x) \\ \mathbf{m} \end{pmatrix}, \begin{pmatrix} k(x,x) & \boldsymbol{\kappa}^\top(x) \\ \boldsymbol{\kappa}(x) & \mathbf{K} + \mathbf{N} \end{pmatrix} \right).$$

Posterior distribution. Conditioning on the observations $\mathbf{y}$, the posterior distribution of $f(x)$ is Gaussian with mean and covariance given by the standard conditional Gaussian formulas:

$$\mathbb{E}[f(x) \mid \mathbf{y}] = m(x) + \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \mathbf{N}\big)^{-1}(\mathbf{y} - \mathbf{m}), \tag{2.19}$$

$$\mathrm{Cov}[f(x), f(x') \mid \mathbf{y}] = k(x, x') - \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \mathbf{N}\big)^{-1}\boldsymbol{\kappa}(x'). \tag{2.20}$$

By applying this to any finite collection of test points, the posterior process $f \mid \mathbf{y}$ is a Gaussian process with mean function and covariance kernel given by (2.19) and (2.20). So far, we have obtained the posterior GP under the *heteroscedastic* Gaussian noise, presented by the following theorem.

Theorem 3.2: Posterior GP under Additive Gaussian Noise

Let $f \sim \mathcal{GP}(m, k)$ be a Gaussian process on $\mathcal{X}$. Suppose we observe noisy data $\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}$ at inputs $\mathbf{x} = \{x_1, \ldots, x_N\}$, where $\mathbf{f} = (f(x_1), \ldots, f(x_N))^\top$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{N})$ is independent Gaussian noise with known covariance $\mathbf{N}$.
Then the posterior process $f \mid \mathbf{y}$ is Gaussian with mean function

$$m_{\mathbf{y}}(x) = m(x) + \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \mathbf{N}\big)^{-1}(\mathbf{y} - \mathbf{m}),$$

and covariance kernel

$$k_{\mathbf{y}}(x, x') = k(x, x') - \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \mathbf{N}\big)^{-1}\boldsymbol{\kappa}(x').$$

Here, $\mathbf{K}$ is the covariance matrix $\big(k(x_i, x_j)\big)_{i,j=1}^N$, $\mathbf{m}$ is the mean vector $\big(m(x_1), \ldots, m(x_N)\big)^\top$, and $\boldsymbol{\kappa}(x) = \big(k(x, x_1), \ldots, k(x, x_N)\big)^\top$.

A commonly encountered special case arises when the observation noise is *homoscedastic*, i.e., independent and identically distributed Gaussian noise with variance $\sigma^2 > 0$. This assumption simplifies the posterior expressions and is widely used in practice.

Corollary 3.3

Posterior GP under i.i.d. Gaussian Noise Under the assumptions of Theorem 3.2, suppose the observation noise covariance is $\mathbf{N} = \sigma^2\mathbf{I}$, where $\sigma^2 > 0$ and $\mathbf{I}$ is the $N \times N$ identity matrix. Then the posterior process $f \mid \mathbf{y}$ is Gaussian with mean and covariance given by

$$m_{\mathbf{y}}(x) = m(x) + \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \sigma^2\mathbf{I}\big)^{-1}(\mathbf{y} - \mathbf{m}),$$

$$k_{\mathbf{y}}(x, x') = k(x, x') - \boldsymbol{\kappa}^\top(x)\big(\mathbf{K} + \sigma^2\mathbf{I}\big)^{-1}\boldsymbol{\kappa}(x').$$

3.3 Approximate Inference

Motivation: Inference from Intractable Evidence

In this section, we discuss the most general case of inference for a GP f. That is, given $f \sim \mathcal{GP}(m, k)$ and evidence $z \in \mathcal{Z}$, where $\mathcal{Z}$ is the evidence space, we consider scenarios where the evidence z is *intractable* for exact inference. This intractability can arise for two main reasons as described in the following.

Non-Gaussian Evidence The evidence z may be non-Gaussian, meaning that the likelihood $p(z|f)$ is not Gaussian. Formally, Bayes' inference reads

$$p(f|z) = \frac{p(z,f)}{p(z)} = \frac{p(z|f)p(f)}{p(z)} \propto p(z|f)p(f). \tag{2.21}$$

However, since f is a random function with an implicit probability measure, the joint distribution $p(z,f)$ is typically intractable or unknown in closed form for arbitrary z. Consequently, the posterior $p(f|z)$ may no longer be a Gaussian process, and the marginal likelihood $p(z)$ is generally unavailable or expensive to compute. This makes exact inference infeasible.

Evidence Scaling Even when the evidence z is Gaussian (or approximately so), exact inference can become computationally prohibitive when scaling the evidence. Consider an Gaussian evidence $\mathbf{z} \in \mathbb{R}^N$ where the evidence scale is represented by N. Then, the covariance matrix Σ in Section 3.1 leads to $\mathcal{O}(N^3)$ time complexity (for inverting Σ) and $\mathcal{O}(N^2)$ space complexity. In the case that the evidence is a dataset of observations with additive noise (Section 3.2, N is the number of observations. This scaling issue renders exact GP inference infeasible for large-scale problems, motivating approximate inference methods to reduce computational costs.

Given such intractable evidence, it is difficult to exactly infer the exact posterior of f, which may not be a GP or is too costly to compute. Instead, we seek to approximate the true posterior $p(f|z)$, by a *surrogate Gaussian process* $q(f)$. This approximate inference framework underpins many practical applications, such as Bayesian optimization and GP-based control, where tractable and scalable inference is essential.

Naïve Approximation: Gaussianization and Data Size Reduction

A simple and widely used heuristic is to assume that evidence z can be transformed via a mapping $g : \mathcal{Z} \to \mathbb{R}^d$ to a *tractable* form, so that the inference framework established in Section 3.1 or Section 3.2 can be applied. The transform g may serve one or both of the following purposes:

- **Gaussianization:** Transform z to approximately Gaussian variables $\tilde{\mathbf{z}} = g(z)$, enabling Gaussian process conditioning using the known formulas.
- **Data size reduction:** Compress or summarize large-scale evidence z into a lower-dimensional representation $\tilde{\mathbf{z}}$, reducing computational cost for inference.

Concretely, we assume that there exists such a transform g and denote $\tilde{\mathbf{z}} = g(z)$. Then, the joint distribution of $g(z)$ and the latent function f can be approximated by a joint Gaussian:

$$\begin{bmatrix} f(x) \\ \tilde{\mathbf{z}} \end{bmatrix} \sim \begin{bmatrix} \mathcal{GP} \\ \mathcal{N} \end{bmatrix} \left(\begin{pmatrix} m(x) \\ \mu \end{pmatrix}, \begin{pmatrix} k(x,x') & \kappa^\top(x) \\ \kappa(x) & \Sigma \end{pmatrix} \right),$$

where μ and Σ are the empirical mean and covariance of the transformed evidence $\tilde{\mathbf{z}}$, and κ approximates the cross-covariance between $g(z)$ and f.

In practice, this "Gaussianization and/or data reduction" approach is often used implicitly in settings such as Bayesian optimization or GP-based control, where the evidence z (e.g., noisy, thresholded, or very large datasets) is first standardized and transformed. During BO iterations, observations may be strategically sifted or summarized to reduce the data size.

Although conceptually naïve, this approach is computationally efficient and can perform surprisingly well when z is not severely non-Gaussian or when the downstream tasks depend primarily on posterior mean estimates rather than full uncertainty quantification. This empirical effectiveness can be attributed to the fact that many real-world non-Gaussian observations are generated as mild perturbations or nonlinear transformations of an underlying latent Gaussian variable.

Laplace Approximation

The Laplace method approximates the posterior process by performing a local second-order Taylor expansion of the log joint distribution $\log p(z, f)$ around its mode $\hat{f} = \arg\max_f p(z|f)p(f)$. Letting

$$\Psi(f) = \log p(z|f) + \log p(f),$$

the posterior is approximated by a Gaussian process centered at $\hat{f}$ with covariance given by the inverse Hessian of $-\Psi(f)$:

$$q(f) = \mathcal{GP}\big(\hat{f}, \big(\nabla_f^2[-\Psi(f)]\big)^{-1}\big). \tag{2.22}$$

This method is primarily suitable for handling *non-Gaussian evidence*, as it approximates a non-Gaussian posterior by a Gaussian around the mode. It is relatively simple and computationally efficient for moderate data sizes. However, Laplace approximation does not inherently address *large-scale* datasets, since it still requires inversion of covariance matrices of size proportional to the number of observations, making it less practical for very large datasets. Furthermore, its accuracy depends on the posterior being well-approximated by a Gaussian near the mode, which may be poor for heavy-tailed or multimodal likelihoods.

Expectation Propagation

Expectation propagation (EP) constructs a tractable Gaussian approximation by iteratively refining local site approximations to the non-Gaussian likelihood terms. If the likelihood factorizes as $p(z|f) = \prod_i p(z_i|f(x_i))$, EP approximates each term by a Gaussian site function

$$p(z_i|f(x_i)) \approx \tilde{Z}_i \, \mathcal{N}\big(f(x_i) \,|\, \tilde{\mu}_i, \tilde{\sigma}_i^2\big),$$

and updates $\{\tilde{\mu}_i, \tilde{\sigma}_i^2\}$ iteratively to match the marginal moments of the true and approximate distributions. EP is well-suited for *non-Gaussian evidence*, often yielding more accurate approximations than Laplace. However, like Laplace, EP typically does not scale well to very large datasets due to the cubic complexity of GP covariance matrix operations, limiting its direct applicability to *large-scale evidence* without additional approximations (e.g., sparse or structured kernels).

Variational Inference

Variational inference formulates posterior approximation as an optimization problem: choose a Gaussian process $q(f)$ (often parameterized by a finite set of inducing variables $\mathbf{u} = f(\mathbf{Z})$) to minimize the Kullback–Leibler divergence $\mathrm{KL}[\, q(f) \,\|\, p(f|z) \,]$. This yields the evidence lower bound (ELBO):

$$\log p(z) \geq \mathbb{E}_{q(f)}[\log p(z|f)] - \mathrm{KL}[q(f) \,\|\, p(f)],$$

which can be maximized w.r.t. the parameters of $q(f)$. By assuming $q(f)$ is a GP with a tractable finite-dimensional marginal $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$, the posterior mean and covariance functions are given in closed form. Sparse variational GPs, such as those proposed by Titsias [2], provide scalable implementations suitable for *large-scale evidence* by reducing computational complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$, where $M \ll N$ is the number of inducing points. Variational inference also naturally accommodates *non-Gaussian evidence* through flexible likelihood models and stochastic optimization of the ELBO. Hence, variational methods are well-suited for both *non-Gaussian and large-scale* inference problems.

Monte Carlo and Sampling-based Methods

Alternatively, posterior samples can be drawn from $p(f|z)$ using Markov chain Monte Carlo (MCMC) methods such as elliptical slice sampling [3] or Hamiltonian Monte Carlo. These methods are capable of handling *non-Gaussian evidence* and provide asymptotically exact inference. However, they are computationally intensive and scale poorly with dataset size, as each iteration requires expensive covariance matrix operations. Consequently, MCMC methods are typically restricted to *small-scale* problems or used for diagnostic purposes rather than large-scale inference.

4 Properties

4.1 Differentiability

A central property of Gaussian processes (GPs) concerns the differentiability of their sample paths and the distribution of their derivatives. Consider a function $f : \mathcal{X} \to \mathbb{R}$ with $\mathcal{X} \subset \mathbb{R}^d$, distributed as $f \sim \mathcal{GP}(m, k)$. For any input dimension i, the partial derivative of f at $\mathbf{x}$ (if it exists) is given by

$$\frac{\partial f}{\partial x_i}(\mathbf{x}) = \lim_{h \to 0} \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h},$$

where $\mathbf{e}_i$ denotes the i-th standard basis vector. Since this expression is a linear transformation of Gaussian random variables for $h > 0$, the derivative remains Gaussian-distributed in the limit as $h \to 0$. Provided that both the mean function μ and the covariance function K are differentiable with respect to their inputs, the distribution of the partial derivative is

$$\frac{\partial f(\mathbf{x})}{\partial x_i} \sim \mathcal{N}\left(\frac{\partial m(\mathbf{x})}{\partial x_i}, \frac{\partial^2 k(\mathbf{x}, \mathbf{x})}{\partial x_i \partial x_i'} \right).$$

If this condition holds for all $i = 1, \ldots, d$, the process is said to be *mean-square differentiable* at $\mathbf{x}$. When f is mean-square differentiable everywhere in its domain, both the function and its gradient jointly follow a Gaussian process:

$$\begin{bmatrix} f \\ \nabla f \end{bmatrix} \sim \mathcal{GP}\left(\begin{pmatrix} m \\ \nabla m \end{pmatrix}, \begin{pmatrix} k(\mathbf{x}, \mathbf{x}') & \nabla_{\mathbf{x}}^{\top} k(\mathbf{x}, \mathbf{x}') \\ \nabla_{\mathbf{x}} k(\mathbf{x}, \mathbf{x}') & \left(\frac{\partial^2 k(\mathbf{x}, \mathbf{x}')}{\partial x_i \partial x_j'} \right)_{i,j=1}^{d} \end{pmatrix} \right). \tag{2.23}$$

This property enables principled gradient-based reasoning under GP models, as the joint distribution captures both function value and derivative uncertainty in a coherent Bayesian framework.

4.2 Global Maximum

Consider a function $f : \mathcal{X} \to \mathbb{R}$ with distribution $\mathcal{GP}(f; \mu, K)$. If f has a global optimum, we denote:

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

where x^* is the location of the global optimum and f^* is its value. In case that f is random and drawn from a GP, x^* and f^* are random variables.

The existence and uniqueness of a global maximum under a Gaussian process can also be established under mild regularity conditions. Let $\mathcal{X}$ be a compact metric space, and suppose $f : \mathcal{X} \to \mathbb{R}$ follows a zero-mean GP with covariance function K, i.e., $f \sim \mathcal{GP}(0, K)$. If f has continuous sample paths, then by the theorem of Kim and Pollard (1990), f almost surely attains a unique global maximum on $\mathcal{X}$, provided that

$$\operatorname{var}[f(x) - f(x')] = K(x, x) - 2K(x, x') + K(x', x') \neq 0 \quad \text{for all } x \neq x'.$$

In other words, randomly sampling f from the GP, the probability that f has a unique maximum is 1. Intuitively, this variance condition prevents the process from being locally constant on any subset of $\mathcal{X}$, ensuring that no two distinct points can achieve the same maximal value with nonzero probability. Under these assumptions, the sample paths of a GP are not only continuous but also exhibit well-defined, isolated extrema—guaranteeing that the global maximum exists and is unique almost surely.

5 Modeling

In Gaussian process (GP) modeling, the choices of prior mean and covariance (kernel) functions are foundational. These choices encode our assumptions about the unknown function and directly shape the behavior of the GP posterior after observing data. The flexibility and power of GPs derive from the ability to select and compose these functions to match domain knowledge, desired smoothness, periodicity, or other structural properties.

5.1 Mean Functions

The mean function $m : \mathcal{X} \to \mathbb{R}$ specifies the expected value of the function at each input before any data is observed. Although the kernel typically plays a more dominant role in GP inference, the mean function can encode important global trends or offsets.

Zero Mean Function

The simplest and most common choice is the zero mean function, $m(x) = 0$ for all x. This reflects a prior belief that the function oscillates around zero, or more generally, that we lack information about its baseline value. The zero mean is often chosen for mathematical convenience and because, in practice, the posterior mean will shift toward the data as observations are incorporated. However, if the true function is systematically nonzero, this choice may slow down learning or bias extrapolations far from observed data.

Constant Mean Function

A constant mean function, $m(x) = c$, introduces a global offset. The constant c can be treated as a hyperparameter and estimated from data. This choice is appropriate when prior knowledge suggests the function is centered around a particular value, but otherwise lacks spatial structure. In regression problems where the outputs are known to be centered away from zero, a constant mean can improve predictive accuracy and extrapolation behavior.

Linear Mean Function

A linear mean function takes the form $m(x) = \mathbf{w}^\top x + b$, where $\mathbf{w}$ and b are parameters. This models a global linear trend, which is useful when the underlying function is expected to increase or decrease steadily in certain directions. Incorporating a linear mean can relieve the kernel of the burden of modeling linear effects, allowing it to focus on nonlinear deviations. Linear mean functions are particularly effective in time series analysis (to capture drift) or spatial modeling (to capture gradients).

Polynomial and Basis Function Means

More generally, the mean function can be expressed as a sum over basis functions,

$$m(x) = \sum_{j=1}^{J} \beta_j \phi_j(x),$$

where $\phi_j(x)$ are chosen basis functions (e.g., polynomials, Fourier modes, wavelets), and β_j are coefficients. This framework allows the incorporation of domain-specific knowledge, such as periodicity or spatial structure. For example, a quadratic mean function can model global curvature, while a periodic basis can encode seasonal effects. The coefficients are typically

learned from data, either jointly with kernel hyperparameters or in a hierarchical Bayesian framework.

5.2 Kernel (Covariance) Functions

The kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defines the covariance between function values at different points, encapsulating assumptions about smoothness, periodicity, stationarity, and more. The choice of kernel fundamentally determines the expressiveness and inductive bias of the GP.

Squared Exponential (RBF) Kernel

The squared exponential or radial basis function (RBF) kernel is given by

$$k_{\text{SE}}(x, x') = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right),$$

where σ_f^2 is the signal variance and ℓ is the lengthscale. This kernel produces sample functions that are infinitely differentiable and extremely smooth. It is both stationary (invariant under translation) and isotropic (invariant under rotation), depending only on the Euclidean distance between points. The RBF kernel is a universal approximator, meaning that with enough data, it can represent any continuous function on a compact domain. However, its strong smoothness assumption can be overly restrictive for rough or discontinuous functions.

Matérn Kernel Family

The Matérn class generalizes the RBF kernel and allows explicit control over smoothness. It is defined as

$$k_{\text{Matérn}}(x, x') = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\|x - x'\|}{\ell}\right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}\|x - x'\|}{\ell}\right),$$

where K_ν is the modified Bessel function of the second kind, and $\nu > 0$ is a smoothness parameter. For $\nu = 1/2$, the kernel reduces to the exponential kernel, producing non-differentiable functions. For $\nu = 3/2$ and $\nu = 5/2$, the sample paths are once and twice differentiable, respectively. As $\nu \to \infty$, the Matérn kernel converges to the RBF kernel. This family is particularly useful in applications where the degree of smoothness is uncertain or where rougher functions are expected.

Periodic Kernel

To model strictly periodic functions, the periodic kernel is used:

$$k_{\text{Per}}(x, x') = \sigma_f^2 \exp\left(-\frac{2\sin^2(\pi\|x - x'\|/p)}{\ell^2}\right),$$

where p is the period. This kernel is stationary with respect to the periodic distance and is suitable for time series with seasonal or cyclic behavior. The lengthscale ℓ controls how rapidly the correlation decays away from perfect periodicity.

Linear and Polynomial Kernels

The linear kernel,

$$k_{\text{Lin}}(x, x') = \sigma_b^2 + \sigma_v^2 x^\top x',$$

models functions with global linear trends. It is not stationary, as the covariance depends on the absolute locations, not just their difference. Polynomial kernels generalize this idea:

$$k_{\text{Poly}}(x, x') = (\alpha x^\top x' + c)^d,$$

where d is the degree. These kernels can capture polynomial relationships but may be prone to overfitting in high dimensions.

Spectral Mixture Kernel

The spectral mixture kernel provides a flexible way to learn arbitrary stationary covariance structures by modeling the kernel's spectral density as a mixture of Gaussians:

$$k_{\text{SM}}(x, x') = \sum_{q=1}^{Q} w_q \prod_{d=1}^{D} \exp(-2\pi^2 (x_d - x'_d)^2 v_{qd}) \cos(2\pi (x_d - x'_d) \mu_{qd}),$$

where w_q, v_{qd}, and μ_{qd} are weights, variances, and means of the mixture components. This kernel can approximate any stationary kernel, as guaranteed by Bochner's theorem.

Quadratic and Anisotropic Kernels

Quadratic kernels take the form

$$k(x, x') = (x - x')^\top \mathbf{B} (x - x'),$$

where $\mathbf{B}$ is positive semidefinite. Such kernels can encode direction-dependent smoothness and are useful in spatial modeling where anisotropy is present.

5.3 Properties of Kernels

Stationarity

A kernel is stationary if it depends only on the difference $x - x'$, not on the absolute locations. Stationary kernels are appropriate when the statistical properties of the underlying process are invariant under translation. This assumption is common in geostatistics and time series

analysis. Stationarity simplifies both theoretical analysis and computational implementation, especially when leveraging spectral methods.

A foundational result in the theory of stationary kernels is Bochner's theorem:

Theorem 5.1: Bochner, 1933

A continuous, shift-invariant kernel $k(x - x')$ on $\mathbb{R}^d$ is positive definite if and only if it is the Fourier transform of a finite non-negative measure μ:

$$k(\tau) = \int_{\mathbb{R}^d} e^{2\pi i \omega^\top \tau} \, d\mu(\omega)$$

Bochner's theorem implies that designing stationary kernels is equivalent to specifying non-negative spectral densities. This result underpins the construction of the spectral mixture kernel and motivates kernel learning in the frequency domain. In practice, by parameterizing the spectral density (for example, as a mixture of Gaussians), one can flexibly learn stationary kernels directly from data. The theorem also highlights that not all covariance structures are stationary, and nonstationary kernels must be constructed via other means.

Isotropy

Isotropy is a stronger condition than stationarity: an isotropic kernel depends only on the Euclidean distance between points, $k(x, x') = \psi(\|x - x'\|)$. Isotropic kernels are suitable when the process is invariant under both translation and rotation, such as in homogeneous physical systems. However, isotropic kernels may be too restrictive in the presence of anisotropy or when certain directions are more important than others.

Universal Approximation

A kernel is universal if the associated reproducing kernel Hilbert space (RKHS) is dense in the space of continuous functions on compact domains. This means that, given enough data, the GP can approximate any continuous function arbitrarily well. The RBF kernel is a classic example of a universal kernel. Universal approximation is a desirable property in nonparametric regression, as it ensures that the GP can adapt to diverse function shapes without strong parametric assumptions.

5.4 Kernel Engineering: Composition and Transformation

The expressive power of Gaussian processes can be greatly enhanced by composing and transforming kernels. Several principled operations are available:

Sum and Product of Kernels

Given kernels k_1 and k_2, their sum $k(x, x') = k_1(x, x') + k_2(x, x')$ corresponds to modeling the function as a sum of independent components, each governed by its own covariance structure. For example, adding a linear kernel to an RBF kernel allows the GP to model both global linear trends and local smooth deviations.

The product $k(x, x') = k_1(x, x') \cdot k_2(x, x')$ captures functions that simultaneously exhibit properties of both kernels. For instance, multiplying a periodic kernel by an RBF kernel yields locally periodic functions, where the periodicity is modulated by smoothness.

Scaling

Kernels can be scaled by positive constants to adjust the overall variance. We may introduce a positive coefficient function $a : \mathcal{X} \mapsto \mathbb{R}_{>0}$, and define

$$k_a(x, x') = a(x)k(x, x')a(x'),$$

which scales k by a. Constant scaling $k_c(x, x') = c^2 k(x, x')$ is a special case where $c \in \mathbb{R}_{>0}$ is a constant.

Input Warping

The inputs to the kernel can be transformed via a mapping $\phi : \mathcal{X} \to \mathbb{R}^d$, yielding a warped kernel $k(\phi(x), \phi(x'))$. Input warping is a powerful technique for learning nonlinear representations, and can be implemented using neural networks (deep kernel learning). This allows the GP to focus its modeling capacity on relevant features or latent structures.

Automatic Relevance Determination (ARD)

ARD is a parameterization in which each input dimension x_d is assigned its own lengthscale ℓ_d. For example, in the RBF kernel,

$$k_{\text{ARD}}(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d - x'_d)^2}{\ell_d^2}\right)$$

This allows the GP to automatically identify and downweight irrelevant dimensions, effectively performing feature selection.

Other Transformations

Kernels can be constructed via convolution, integration over latent variables, or other functional transformations, enabling advanced modeling of nonstationary, hierarchical, or structured dependencies.

5.5 High-Dimensional Modeling

Gaussian processes encounter significant challenges as the input dimension D increases, a phenomenon known as the curse of dimensionality. The number of data points required to adequately cover the input space grows exponentially, and kernel matrices become large and expensive to invert.

Several strategies have been developed to address these issues:

Low-Dimensional Embeddings

One approach is through input warping — map the high-dimensional input $x \in \mathbb{R}^D$ to a lower-dimensional latent space $\phi(x) \in \mathbb{R}^d$, where $d \ll D$. The kernel is then defined on the latent space:

$$k(x, x') = k_{\text{latent}}(\phi(x), \phi(x'))$$

Embeddings can be linear (e.g., principal component analysis), or nonlinear and learned from data (e.g., via neural networks). This reduces the effective dimensionality and focuses modeling power on the most informative features.

Additive Kernel Decomposition

In additive modeling, the function is assumed to decompose as a sum of functions on low-dimensional subspaces:

$$f(x) = \sum_{j=1}^{J} f_j(x_{S_j}),$$

where $S_j \subseteq \{1, \ldots, D\}$ are subsets of input dimensions. The corresponding kernel is

$$k(x, x') = \sum_{j=1}^{J} k_j(x_{S_j}, x'_{S_j})$$

Additive GPs can efficiently model high-dimensional functions when each output depends only on a small subset of inputs, dramatically reducing sample complexity and computational cost.

Feature Selection

In many applications, not all features in the high-dimensional input space are useful to modeling. Therefore, we can carefully select a subset of features to approximate the full GP.

The **explicit feature selection** leads to the sparse GP methods, which introduce a set of inducing points $\{z_m\}_{m=1}^{M}$, with $M \ll N$, and approximate the full GP using only the inducing variables. This reduces computational complexity from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$. Structured

kernel approximations (e.g., Kronecker or Toeplitz matrices) exploit regularities in the input grid to further accelerate inference.

As discussed above, automatic relevance determination can be viewed as an *implicit feature selection*. It enables the GP to ignore irrelevant dimensions by learning large lengthscales for those features. This effectively reduces the dimensionality of the problem and mitigates overfitting.

CHAPTER 3

ALGORITHMIC DESIGN

1 Models in Bayesian Optimization

Bayesian Optimization (BO) is a probabilistic framework for the global optimization of black-box functions. It is particularly effective when direct evaluations of the objective are expensive or time-consuming, motivating the need for sample-efficient search strategies. Throughout this chapter, we use the following notations:

- $f(x)$: the true objective function, unknown and costly to evaluate.
- $\phi = f(x)$: latent function value at input x.
- y: observed output, possibly noisy or transformed.
- $\mathcal{X}$: input (decision) space.
- $\mathcal{Y}$: observation (output) space, typically $\mathbb{R}$.
- $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^{T}$: the dataset of T observed input-output pairs.

1.1 Observation Model

To accurately capture the data generation process, it is essential to specify how the true function values ϕ relate to the observed outputs y at each input location x. The observation model formalizes this relationship through the conditional probability $p(y \mid \phi, x)$. The choice of observation model reflects our assumptions about the noise and transformation mechanisms in the system, and guides subsequent inference and uncertainty quantification.

Exact Observation In some idealized scenarios, we may assume that observations are noise-free, i.e., we directly observe the true function value. This is modeled as:

$$p(y \mid \phi, x) = \delta(y - \phi) \tag{3.1}$$

where $\delta(\cdot)$ is the Dirac delta function. Under this model, $y = f(x)$ for all x.

Homoskedastic Additive Gaussian Noise In most practical cases, measurements are corrupted by noise. The most common assumption is that the noise is additive and has constant variance (homoskedasticity):

$$p(y \mid \phi, x) = \mathcal{N}(y; \phi, \sigma^2) \tag{3.2}$$

where σ^2 is the fixed noise variance. This setting underpins standard Gaussian Process (GP) regression, where all observations are assumed equally reliable.

Heteroskedastic Additive Noise Sometimes, the reliability of measurements varies across the input domain. To model such situations, we allow the noise variance to depend on the input:

$$p(y \mid \phi, x) = \mathcal{N}(y; \phi, \sigma^2(x)) \tag{3.3}$$

where $\sigma^2(x)$ is a function of x. This heteroskedastic model is suitable when certain regions of the input space yield more uncertain observations than others.

Arbitrary Observation Model In general, the observation process may involve complex noise distributions, censoring, or non-linear transformations. Any model $p(y \mid \phi, x)$ can be considered, provided y remains informative about ϕ. Examples include non-Gaussian noise, censored or truncated data, and outputs subject to unknown transformations. Such generality increases modeling flexibility but often necessitates advanced inference techniques (e.g., variational inference, MCMC), and complicates the computation of predictive uncertainty and acquisition functions.

1.2 Surrogate Model

Bayesian Optimization relies on a surrogate model to approximate the unknown objective function f. The surrogate provides a probabilistic representation of our beliefs about f, updated as new data $\mathcal{D} = \{(x_t, y_t)\}_{t=1}^{T}$ are collected. In practice, the surrogate is constructed by specifying a prior $p(f)$ over functions and inferring the posterior $p(f \mid \mathcal{D})$.

Bayesian Inference of the Surrogate

The surrogate posterior is obtained via Bayes' rule:

$$p(f \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid f)p(f)}{p(\mathcal{D})} \tag{3.4}$$

where $p(f)$ encodes prior assumptions about f, and $p(\mathcal{D} \mid f)$ is the likelihood of the observed data given f. Expanding the likelihood, and assuming that input locations $\mathbf{x}$ are selected independently of f, we have:

$$\begin{aligned} p(f \mid \mathcal{D}) &= \frac{p(\mathbf{y}_\mathcal{D} \mid f, \mathbf{x}_\mathcal{D})p(f)}{p(\mathbf{y}_\mathcal{D} \mid \mathbf{x}_\mathcal{D})} \\ &\propto p(\mathbf{y}_\mathcal{D} \mid f, \mathbf{x}_\mathcal{D})p(f), \end{aligned}$$

where $\mathbf{x}_\mathcal{D} = (x_1, \ldots, x_T) \in \mathcal{X}^T$ and $\mathbf{y}_\mathcal{D} = (y_1, \ldots, y_T) \in \mathcal{Y}^T$.

Given an observation model $p(\mathbf{y} \mid \boldsymbol{\phi}, \mathbf{x})$, the surrogate posterior can be expressed as:

$$p(f \mid \mathcal{D}) \propto p(\mathbf{y}_\mathcal{D} \mid \boldsymbol{\phi}_\mathcal{D} = f(\mathbf{x}_\mathcal{D}), \mathbf{x}_\mathcal{D})\, p(f) \tag{3.5}$$

In general, this inference is challenging, as normalization requires integrating over all possible functions f.

Conjugate Prior

A prior $p(f)$ is said to be **conjugate** to the observation model $p(y|\phi, x)$ if the posterior $p(f|\mathcal{D})$ belongs to the same family as the prior. For example, a Gaussian Process (GP) prior is conjugate to a Gaussian likelihood with additive noise: the posterior remains a GP, and the update equations are analytic (see Chapter 2). Conjugacy simplifies inference, enabling closed-form expressions for the posterior mean and covariance, which are crucial for efficient Bayesian Optimization. In non-conjugate cases (such as non-Gaussian likelihoods), approximate inference methods—like variational inference or Markov Chain Monte Carlo (MCMC)—are required.

Inference Treatment

The choice of inference method depends on the combination of prior and observation model. *Exact inference* is possible when conjugacy holds, allowing for analytic updates. Otherwise, *approximate inference* methods such as Laplace approximation, variational inference, or MCMC must be employed. For large datasets or high-dimensional input spaces, scalable approximations (e.g., inducing points, random features) are often necessary to maintain computational tractability.

2 Model Selection and Hyperparameter Adaptation

A central challenge in Bayesian modeling is determining which surrogate prior and observation model best represent the data. This process, known as model selection, directly impacts the quality of predictions and the effectiveness of optimization. Before proceeding, it is important to clarify that the model's scope is limited to the input-output correlation $p(\mathbf{y} \mid \mathbf{x})$ for any finite set of input points $\mathbf{x}$. This correlation is fully specified by the surrogate prior $p(f)$ and the observation model $p(y \mid \phi, x)$, with the marginal likelihood integrating over latent function values:

$$p(\mathbf{y} \mid \mathbf{x}) = \int p(\mathbf{y} \mid \boldsymbol{\phi}, \mathbf{x}) p(\boldsymbol{\phi} \mid \mathbf{x})\, \mathrm{d}\boldsymbol{\phi}, \tag{3.6}$$

where $p(\boldsymbol{\phi} \mid \mathbf{x})$ is determined by the prior $p(f)$; Given observed data $\mathcal{D}$, the predictive distribution becomes

$$p(\mathbf{y} \mid \mathbf{x}, \mathcal{D}) = \int p(\mathbf{y} \mid \boldsymbol{\phi}, \mathbf{x}) p(\boldsymbol{\phi} \mid \mathbf{x}, \mathcal{D})\, \mathrm{d}\boldsymbol{\phi}, \tag{3.7}$$

2.1 Parameterization of the Hypothesis Space

Both the surrogate prior $p(f \mid \theta)$ and the observation model $p(y \mid \phi, x, \theta)$ may depend on hyperparameters $\theta \in \Theta$. For example, in a Gaussian Process (GP) surrogate, θ might include kernel parameters and noise variances. When the observation process is not fully specified, it is necessary to model $p(y \mid \phi, x, \theta)$ as well, where $\phi = f(x)$.

We define the model family $\mathcal{M}$ as:

$$\mathcal{M} = \{p(f \mid \theta),\ p(y \mid \phi, x, \theta) : \theta \in \Theta\} \tag{3.8}$$

where θ can be any parameterization (scalar, vector, matrix, etc.).

Given θ, the marginal likelihood of the data is:

$$p(\mathbf{y} \mid \mathbf{x}, \theta) = \int p(\mathbf{y} \mid \boldsymbol{\phi}, \mathbf{x}, \theta) p(\boldsymbol{\phi} \mid \mathbf{x}, \theta)\, \mathrm{d}\boldsymbol{\phi} \tag{3.9}$$

where $p(\boldsymbol{\phi} \mid \mathbf{x}, \theta)$ is induced by the surrogate prior $p(f \mid \theta)$.

2.2 Model Selection

Model selection is the process of choosing hyperparameters θ that yield the best fit to the observed data. Two common approaches are Maximum Likelihood Estimation (MLE) and Maximum A Posteriori (MAP) estimation, both motivated by *empirical Bayes* principles — estimating hyperparameters from the observed data, rather than specifying them fully in advance or integrating them out as in hierarchical Bayesian inference.

- **MLE:** The core idea is to select the hyperparameters θ that maximize the probability of the observed data under the model. Formally,

  $$\theta^* = \arg\max_{\theta} p(\mathbf{y} \mid \mathbf{x}, \theta) \tag{3.10}$$

  This approach treats θ as fixed but unknown, and seeks the values that make the observed data most likely.

- **MAP:** If prior knowledge about θ is available, it can be incorporated by maximizing the posterior probability of θ given the data:

  $$\theta^* = \arg\max_{\theta} p(\mathbf{y} \mid \mathbf{x}, \theta) p(\theta) \tag{3.11}$$

  Here, the core idea is to balance data fit with prior beliefs, yielding a regularized estimate.

Both MLE and MAP provide point estimates of θ, which are computationally efficient and easy to interpret. However, they ignore uncertainty in hyperparameter values, which can lead to overconfident predictions and suboptimal optimization performance, especially when data is scarce or noisy.

2.3 Model Averaging

To address the limitations of point estimation, Bayesian model averaging offers a principled alternative. Instead of committing to a single set of hyperparameters, model averaging integrates over the uncertainty in θ, producing more robust predictions and better-calibrated uncertainty estimates. This approach is particularly important when multiple plausible models exist, or when the data does not strongly constrain θ.

Given a belief distribution $q(\theta)$ over hyperparameters, the predictive distributions are computed by averaging over θ:

$$p(f \mid \mathcal{D}) = \int p(f \mid \mathcal{D}, \theta) q(\theta) \, \mathrm{d}\theta, \tag{3.12}$$

$$p(y \mid x, \mathcal{D}) = \iint p(y \mid x, \phi, \theta) p(\phi \mid x, \mathcal{D}, \theta) q(\theta) \, \mathrm{d}\phi \mathrm{d}\theta. \tag{3.13}$$

Hierarchical Bayes

It is natural to extend Bayesian inference to the hyperparameter level, treating the hyperparameters θ themselves as random variables equipped with a prior distribution. That is, we set $q(\theta) = p(\theta \mid \mathcal{D})$, meaning that predictions and inferences are averaged over the posterior distribution of θ conditioned on the observed data $\mathcal{D}$. This approach provides a principled way to account for uncertainty in model selection and parameterization.

Recall the marginal likelihood in Equation (3.9); the posterior distribution over hyperparameters is then given by Bayes' rule:

$$p(\theta \mid \mathcal{D}) \propto p(\theta) \int p(\mathbf{y}_{\mathcal{D}} \mid \boldsymbol{\phi}_{\mathcal{D}}, \mathbf{x}_{\mathcal{D}}, \theta) \, p(\boldsymbol{\phi}_{\mathcal{D}} \mid \mathbf{x}_{\mathcal{D}}, \theta) \, \mathrm{d}\boldsymbol{\phi}_{\mathcal{D}}, \tag{3.14}$$

where $p(\theta)$ is the prior over hyperparameters, encoding any prior beliefs or domain knowledge.

The procedure above exemplifies the *hierarchical Bayesian model*, which extends standard Bayesian inference by introducing an additional layer of uncertainty over hyperparameters. In this framework, both the model parameters and their hyperparameters are treated as random variables with associated priors. Posterior inference is performed jointly, allowing uncertainty to propagate from the hyperparameter level down to predictions. This hierarchical approach enables principled model averaging, regularizes parameter estimates, and yields more robust and calibrated uncertainty quantification, especially when data is limited or model selection is ambiguous.

Monte-Carlo Approximation

In practice, exact integration over θ is rarely feasible, especially for high-dimensional or non-conjugate models. Monte Carlo methods provide a flexible and scalable solution for approximating these integrals. The procedure is as follows:

1. Draw K samples $\{\theta^{(k)}\}_{k=1}^{K}$ from $q(\theta)$ (e.g., from the posterior $p(\theta \mid \mathcal{D})$).
2. For each sampled $\theta^{(k)}$, compute the predictive distribution by integrating over ϕ:

$$p(y \mid x, \mathcal{D}) \approx \frac{1}{K} \sum_{k=1}^{K} \int p(y \mid x, \phi, \theta^{(k)}) p(\phi \mid x, \mathcal{D}, \theta^{(k)}) \, \mathrm{d}\phi \tag{3.15}$$

Monte Carlo approximation enables flexible model averaging, and can be combined with approximate inference for each sampled $\theta^{(k)}$. This approach is widely used in practice, as it scales well and can accommodate complex, non-conjugate models.

3 Acquisition Policy

3.1 Introducing the Acquisition Function

A central component of Bayesian Optimization (BO) is the *acquisition function*, denoted $\alpha(x_{t+1}|\mathcal{D}_t)$, which guides the sequential selection of input points for evaluation. At each iteration t, given the current dataset $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{t}$, the acquisition function quantifies the expected utility or benefit of evaluating the objective at a candidate location x_{t+1}. The next query is chosen by maximizing the acquisition function:

$$x_{t+1}^{\star} = \arg\max_{x_{t+1} \in \mathcal{X}} \alpha(x_{t+1}|\mathcal{D}_t) \tag{3.16}$$

The acquisition function is typically constructed using the surrogate model's predictive distribution, leveraging both its mean and uncertainty estimates. The design of α embodies the trade-off between **exploration** (sampling where the model is uncertain) and **exploitation** (sampling where the model predicts high objective values), which is fundamental to efficient optimization under limited evaluation budgets.

3.2 From Utility to Acquisition Function

The acquisition function can be formally derived from a *utility function*, which encodes the objective of the optimization process. Let $u(x_1, y_1, x_2, y_2, \ldots, x_T, y_T)$ denote the utility of a trajectory of T observations. The acquisition function can be defined as the expected utility of selecting x_{t+1}, given the current data $\mathcal{D}_t$:

$$\alpha(x_{t+1}|\mathcal{D}_t) := \mathbb{E}_{\mathcal{D}_{t+1:T}|x_{t+1},\mathcal{D}_t}\left[u(\mathcal{D}_t, \mathcal{D}_{t+1:T})\right] \tag{3.17}$$

where $\mathcal{D}_{t+1:T}$ denotes the (random) future observations, and T is the total budget.

Note that considering the entire budget T is computationally intractable when T is large or infinite. Alternatively, we may consider a limited horizon h:

$$\alpha(x_{t+1}|\mathcal{D}_t) := \mathbb{E}_{\mathcal{D}_{t+1:t+h}|x_{t+1},\mathcal{D}_t}\left[u(\mathcal{D}_t, \mathcal{D}_{t+1:t+h})\right] \tag{3.18}$$

Where $h = 1$ yields the *one-step look-ahead acquisition w.r.t. u*:

$$\alpha(x_{t+1}|\mathcal{D}_t) = \mathbb{E}[u(\mathcal{D}_t, x_{t+1}, y_{t+1})]. \tag{3.19}$$

In the following, some utilities are introduced.

Cumulative Reward In analogy to reinforcement learning, one may define the utility as the cumulative sum of observed rewards:

$$u(\mathcal{D}_t) = \sum_{s=1}^{t} y_s \tag{3.20}$$

The expected cumulative reward can be used as an acquisition function, often referred to as the *action-value function* in RL, denoted $q(\mathcal{D}_t, x_{t+1})$. Here, the state is the current data $\mathcal{D}_t$, and the action is the selection of x_{t+1}.

Information Gain **TODO:** Expand on utilities based on information gain, such as entropy reduction or mutual information about the location or value of the optimum. Discuss how these lead to acquisition functions like Entropy Search and Predictive Entropy Search.

3.3 Choices of Acquisition Functions

The design of the acquisition function is crucial, as it determines the efficiency and effectiveness of the optimization process. Common considerations include:

- **Exploration**: Sampling in regions of high model uncertainty to improve the surrogate's accuracy.
- **Exploitation**: Sampling in regions where the surrogate predicts high objective values, to quickly find optima.
- **Balance**: Most acquisition functions balance exploration and exploitation, either explicitly or implicitly, to ensure efficient search.
- **Computational Tractability**: Acquisition functions should be amenable to efficient optimization, as they are maximized at each iteration.

Below, we survey the most widely used acquisition functions, their derivations, and their properties.

Expected Improvement (EI)

Expected Improvement (EI) is one of the most popular acquisition functions in BO. Let $y^{\text{best}} = \max\{y_1, \ldots, y_t\}$ denote the best observed value so far. The improvement at x_{t+1} is defined as:

$$\mathrm{I}(x_{t+1}) = \max\{0, f(x_{t+1}) - y^{\text{best}}\} \tag{3.21}$$

The expected improvement is then:

$$\mathrm{EI}(x_{t+1}) = \mathbb{E}_{f(x_{t+1})|\mathcal{D}_t}\left[\mathrm{I}(x_{t+1})\right] \tag{3.22}$$

For a Gaussian surrogate, with predictive mean $\mu(x_{t+1})$ and standard deviation $\sigma(x_{t+1})$, EI admits a closed-form expression:

$$\mathrm{EI}(x_{t+1}) = (\mu(x_{t+1}) - y^{\text{best}})\Phi(z) + \sigma(x_{t+1})\phi(z) \tag{3.23}$$

where $z = \frac{\mu(x_{t+1}) - y^{\text{best}}}{\sigma(x_{t+1})}$, and $\Phi(\cdot)$, $\phi(\cdot)$ are the standard normal CDF and PDF, respectively. EI naturally balances exploration (via $\sigma(x_{t+1})$) and exploitation (via $\mu(x_{t+1})$), and is well-suited for problems where improvement over the current best is the objective.

Probability of Improvement (PI)

Probability of Improvement (PI) quantifies the probability that sampling at x_{t+1} will yield an objective value exceeding a threshold, typically y^{best}:

$$\mathrm{PI}(x_{t+1}) = \mathbb{P}\left(f(x_{t+1}) \geq y^{\text{best}} + \xi \mid \mathcal{D}_t\right) \tag{3.24}$$

where $\xi \geq 0$ is a small positive margin to encourage exploration. For a Gaussian surrogate:

$$\mathrm{PI}(x_{t+1}) = \Phi\left(\frac{\mu(x_{t+1}) - y^{\text{best}} - \xi}{\sigma(x_{t+1})}\right) \tag{3.25}$$

PI is simple and computationally efficient, but may focus too heavily on exploitation when $\sigma(x_{t+1})$ is small.

Upper Confidence Bound (UCB)

The Upper Confidence Bound (UCB) acquisition function selects points according to an optimistic estimate of the objective:

$$\mathrm{UCB}(x_{t+1}) = \mu(x_{t+1}) + \beta_t \sigma(x_{t+1}) \tag{3.26}$$

where $\beta_t > 0$ is a parameter controlling the exploration-exploitation trade-off. Larger β_t encourages exploration, while smaller β_t emphasizes exploitation. UCB is motivated by regret bounds in bandit theory and is widely used due to its simplicity and strong theoretical guarantees.

Knowledge Gradient (KG)

The Knowledge Gradient (KG) acquisition function seeks to maximize the expected value of information gained from each evaluation, specifically the expected improvement in the optimal decision after observing y_{t+1}:

$$\mathrm{KG}(x_{t+1}) = \mathbb{E}_{y_{t+1}}\left[\max_x \mathbb{E}[f(x)|\mathcal{D}_t \cup \{(x_{t+1}, y_{t+1})\}]\right] - \max_x \mathbb{E}[f(x)|\mathcal{D}_t] \tag{3.27}$$

KG is particularly useful when the total evaluation budget is small, as it explicitly considers the value of learning for future decisions. However, KG can be computationally intensive, as it requires simulating the impact of each potential observation.

Mutual Information (MI) / Entropy Search

Acquisition functions based on information theory aim to reduce uncertainty about the location or value of the global optimum. One prominent example is *Entropy Search* (ES), which seeks to maximize the expected reduction in entropy of the posterior over the maximizer x^*:

$$\mathrm{ES}(x_{t+1}) = \mathbb{E}_{y_{t+1}}\left[H(p(x^*|\mathcal{D}_t)) - H(p(x^*|\mathcal{D}_t \cup \{(x_{t+1}, y_{t+1})\}))\right] \tag{3.28}$$

where $H(\cdot)$ denotes entropy. Similarly, the *Predictive Entropy Search* (PES) acquisition function selects points that maximize the mutual information between y_{t+1} and the global optimum. These methods are powerful in settings where learning the location of the optimum is paramount, but typically require expensive approximations.

Thompson Sampling

Thompson Sampling is a randomized acquisition strategy that selects the next evaluation point by sampling from the surrogate model's posterior distribution over the objective function. At each iteration, the procedure is as follows:

1. **Sample a function:** Draw a sample $\tilde{f}$ from the posterior distribution $p(f \mid \mathcal{D}_t)$ provided by the surrogate model (e.g., a Gaussian Process).
2. **Select the maximizer:** Choose the next query point as the maximizer of the sampled function:
$$x^\star_{t+1} = \arg\max_{x \in \mathcal{X}} \tilde{f}(x) \tag{3.29}$$

This approach implicitly balances exploration and exploitation: the randomness in the posterior sample encourages exploration in regions of high uncertainty, while the maximization step favors exploitation of promising areas. Over repeated iterations, Thompson Sampling efficiently explores the input space and converges to the optimum.

Thompson Sampling is especially attractive in Bayesian Optimization due to its simplicity, scalability, and strong theoretical guarantees in the context of multi-armed bandit problems. It is particularly well-suited for parallel BO, where multiple samples can be drawn and evaluated simultaneously. Furthermore, it readily accommodates complex surrogate models and non-Gaussian posteriors, as long as sampling from the posterior is feasible.

Despite its strengths, practical implementation of Thompson Sampling requires efficient posterior sampling and global optimization of the sampled function. For Gaussian Processes, this can be achieved using random feature approximations or spectral sampling. In high-dimensional spaces or with complex priors, approximate sampling methods (e.g., Monte Carlo, variational inference) may be necessary.

Overall, Thompson Sampling provides a principled and flexible acquisition policy for Bayesian Optimization, and serves as a probabilistic counterpart to deterministic strategies like UCB and Expected Improvement.

Bandit Algorithms and Connections to BO

Bayesian Optimization is closely related to the multi-armed bandit (MAB) problem, where an agent sequentially chooses actions to maximize cumulative reward under uncertainty. Many bandit algorithms, such as Thompson Sampling and UCB, can be interpreted as specific acquisition policies within BO. For instance, Thompson Sampling and UCB (as described above) are direct analogues of bandit strategies, adapted to the probabilistic framework of BO.

TODO: Expand on the connections between BO and bandit algorithms, including regret analysis, theoretical guarantees, and practical implications. Discuss how BO generalizes classical bandit strategies, and highlight scenarios where bandit-inspired acquisition functions are particularly beneficial.

3.4 Optimizing the Acquisition Function

At each iteration, BO requires solving the optimization problem

$$x_{t+1}^{\star} = \arg\max_{x \in \mathcal{X}} \alpha(x|\mathcal{D}_t) \tag{3.30}$$

Unlike the original black-box objective, the acquisition function is typically cheap to evaluate and differentiable, but may be multimodal and non-convex. Several strategies are commonly used to optimize $\alpha(x)$:

- **Grid Search**: Evaluate $\alpha(x)$ on a discrete grid over $\mathcal{X}$ and select the maximizer. Simple but inefficient in high dimensions.
- **Random Search**: Sample candidate points at random and select the best. Effective for low-dimensional or irregular spaces.
- **Gradient-Based Optimization**: Use gradient ascent (e.g., L-BFGS, Adam) to locally maximize $\alpha(x)$. Multiple restarts from different initializations help avoid local maxima.
- **Evolutionary Algorithms**: Apply population-based methods (e.g., CMA-ES, genetic algorithms) to explore complex or high-dimensional domains.
- **Bayesian Optimization of the Acquisition Function**: In some cases, a secondary BO loop can be used to optimize the acquisition function itself, especially for expensive or high-dimensional $\alpha(x)$.

The choice of optimization method depends on the properties of $\alpha(x)$, the dimensionality of $\mathcal{X}$, and computational constraints. Efficient optimization of the acquisition function is critical for the overall performance of BO, as it directly determines the quality of the next query point.

CHAPTER 4

ANALYSIS OF BO ALGORITHMS

CHAPTER 5

PARALLEL AND BATCHED BO

CHAPTER 6

CONTEXTUAL BAYESIAN OPTIMIZATION

CHAPTER 7

MULTI-TASK BO

Part IV

Advanced Topics

CHAPTER 1

HIGH-DIMENSIONAL OPTIMIZATION

CHAPTER 2

ROBUSTNESS

CHAPTER 3

MULTI-OBJECTIVE LEARNING

CHAPTER 4

MULTI-TASK AND TRANSFER LEARNING

CHAPTER 5

INTEGRATION WITH DOMAIN KNOWLEDGE

CHAPTER 6

INTEGRATION WITH LARGE LANGUAGE MODELS

Part V

Practices in Science

BIBLIOGRAPHY

[1] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning), 3. print. Cambridge, Mass.: MIT Press, 2008, ISBN: 978-0-262-18253-9.

[2] M. Titsias, "Variational Learning of Inducing Variables in Sparse Gaussian Processes", in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, PMLR, Apr. 2009, pp. 567–574. (visited on 10/26/2025).

[3] I. Murray, R. Adams, and D. MacKay, "Elliptical slice sampling", in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, Mar. 2010, pp. 541–548. (visited on 10/26/2025).