

## Inference

In this course, we will focus on **probabilistic inference**. This is the process of inferring unknown properties of a system given observations via the mechanics of probability theory.

For example, suppose I want to understand some aspect of the CMU student population, such as the portion of students who believe that cats are better than dogs. Let's call this unknown value  $\theta$ . How can I gain insight into this value?

One thing I could do is ask people what they believe about  $\theta$ , which they might reasonably compactly communicate via a probability distribution (or equivalently by a drawing curve on a piece of paper). Different people might give very different answers to this question! (What do you think are plausible values of  $\theta$ ?)

Of course, asking individual people to guess about  $\theta$  probably won't give very accurate results. Instead, we could conduct a survey to gain some more information about  $\theta$ . So I reach out to some CMU students and ask them if they think cats are better than dogs and record the results. Let's call the results of this experiment  $\mathcal{D}$  (for "data").

What do we do with this data once we've measured it? The goal of probabilistic inference is to make some statements about  $\theta$  given these observations.

## Probability

There are just two laws of probability you need to know. The first is the **sum rule**, also called the **rule of total probability**. Suppose  $X$  is some event, where an event is simply a collection of things that could be true (e.g., "the total value shown on two dice is less than six"), and suppose  $\{Y_i\}_{i=1}^N$  are some mutually exclusive and exhaustive events (e.g.,  $Y_i$  could be the event "the first rolled die has value  $i$ " for  $i \in \{1, 2, \dots, 6\}$ ). Then:

$$\Pr(X) = \sum_{i=1}^N \Pr(X, Y_i),$$

so the total probability of  $X$  is the sum of the probabilities of  $X$  occurring alongside each of the scenarios described by the  $\{Y_i\}$ .

The second law of probability to know is the **product rule**:

$$\Pr(X, Y) = \Pr(Y | X) \Pr(X).$$

Here  $\Pr(Y | X)$  (read "the probability of  $Y$  given  $X$ ") is the probability of event  $Y$  when we restrict to only cases where  $X$  is true. This is easy to see from a Venn diagram.

By manipulating the product rule, we arrive at **Bayes' theorem**:

$$\begin{aligned} \Pr(Y | X) &= \frac{\Pr(X | Y) \Pr(Y)}{\Pr(X)} \\ &= \frac{\Pr(X | Y) \Pr(Y)}{\sum \Pr(X | Y) \Pr(Y)}. \end{aligned}$$

One possible way to interpret this result is as follows. Suppose we are interested in the truth of  $Y$ . We begin with a prior belief about  $Y$ ,  $\Pr(Y)$ . We then learn that  $X$  is true. We use the formula above to update our belief about  $Y$  by conditioning on this new information, giving  $\Pr(Y | X)$ . Bayes' theorem therefore gives a probabilistically consistent way to update one's beliefs given new

information. In this context the value  $\Pr(Y)$  is called a **prior probability**, as it represents our beliefs prior to observing  $X$ . The output,  $\Pr(Y | X)$  is called the **posterior probability**.

The process of updating beliefs in this way is often called **probability inversion**, because after observing  $X$ , we first calculate  $\Pr(X | Y)$  (“how likely was it to observe  $X$  if  $Y$  were true?”), then invert it using Bayes’ theorem to give the desired probability (“how likely is  $Y$  now that I’ve seen  $X$ ?”). Many classical paradoxes arise because these probabilities can be quite different from each other, depending on the prior  $\Pr(Y)$ .

The above results are only valid for discrete random variables  $X$ . Although this can be useful in machine learning (e.g., if these variables correspond to different hypotheses we wish to compare), in practice, we will often be interested in continuous variables, such as  $\theta$  from the example above. Thankfully, the above formulas are perfectly valid for continuous random variables if we replace probabilities  $\Pr$  with probability density functions  $p$  and replace sums with integrals:

$$p(x) = \int p(x, y) \, dy$$
$$p(y | x) = \frac{p(x | y)p(y)}{\int p(x | y)p(y) \, dy}$$

We will always assume that probability density functions exist, because the cases where they don’t are not typically encountered in machine learning. It also saves us from having to state “when the density exists” all the time.

Returning to our survey example, we might begin with a prior belief about the value of  $\theta$ , represented by a prior probability distribution  $p(\theta)$ . To couple our observations  $\mathcal{D}$  to the value of interest, we construct a probabilistic model  $\Pr(\mathcal{D} | \theta)$ , which describes how likely we would see a particular survey result  $\mathcal{D}$  given a particular value of  $\theta$ . Note that this model could have any form and we are free to make it as complicated as we’d like: was there some bias in the sampling mechanism that we need to account for? Do we assume that respondents always tell the truth?

Finally, we use these to compute the posterior probability of  $\theta$  given the survey results,  $p(\theta | \mathcal{D})$ . This posterior distribution encapsulates our entire belief about  $\theta$ ! We can use it to answer various questions we might be about  $\theta$ .

## The Bayesian Method

To summarize, there are four main steps to the Bayesian approach to probabilistic inference (these are summarized from Tony O’Hagan and Jonathan Forster’s excellent introduction in *Kendall’s Advanced Theory of Statistics, Volume 2B*):

- **Likelihood.** First, we construct the likelihood (or **model**),  $p(\mathcal{D} | \theta)$ . This describes the mechanism giving rise to our observations  $\mathcal{D}$  given a setting of the parameters of interest  $\theta$ .
- **Prior.** Next, we summarize our prior beliefs about the parameters  $\theta$ , which we encode via a probability distribution  $p(\theta)$ .
- **Posterior.** Given some observations  $\mathcal{D}$ , we obtain the posterior distribution  $p(\theta | \mathcal{D})$  using Bayes’ theorem.
- **Inference.** We now use the posterior distribution to draw further conclusions as required.

The last step is purposely open-ended. For example, we can use it to make predictions about new data (as in supervised learning), we can summarize it in various ways (e.g., point estimation if we must report a single “best guess” of  $\theta$ ), use it to compare alternative models (a.k.a. Bayesian model comparison), determine which data to obtain next (optimal design of experiments or “active learning” as its more commonly referred to in machine learning), and more. We will consider several of these in this course.

## Issues

Bayesian inference is a completely consistent system for probabilistic reasoning. Unfortunately, it is not without its issues, some of which we list below.

### Origin of priors

In contrast to the model  $p(\mathcal{D} \mid \theta)$ , it is not usually clear where the prior  $p(\theta)$  should come from. There is an entire branch of study concerning prior elicitation, but for now we will simply treat it as given. We will see that several “tricks” often encountered in alternative approaches (such as *regularization*) can be interpreted as implicitly placing particular prior beliefs on  $\theta$ .

### The meaning of probability

Another problem is what exactly *probability* means. The dominant statistical practice for many years (known as *classical* or *frequentist* theory) defines probability in terms of the limit of conducting infinitely many random experiments. Therefore it is impossible to consider the “probability” of a statement such as “at least 50% of CMU students prefer cats to dogs.” This statement is either true or false, so its frequentist probability is either zero or one (but we might not know which). In the Bayesian interpretation, we allow probabilities instead to describe *degrees of belief* in such a proposition. In this way, we can treat everything as a random variable and use the tools of probability to carry out all inference. That is, in Bayesian probability, parameters, data, and hypotheses are all treated the same. This viewpoint is not universally accepted, and there is a lot of fascinating philosophical writing on the subject, which we will entirely avoid.

Note that the two interpretations of probability agree on the axioms and theorems of probability theory. No one argues the truth of Bayes’ theorem. The main difference is that a frequentist would not allow a probability distribution to be placed on parameters such as  $\theta$ , so the use of Bayes’ theorem to update beliefs about parameters in light of data is not allowed in that framework.

### Intractable integrals

Unfortunately, the integral in the denominator of Bayes’ theorem:

$$p(x) = \int p(x \mid y)p(y) \, dy$$

is not in general tractable for arbitrary combinations of priors and likelihoods. For this reason, we will spend a lot of time discussing various schemes to approximate the posterior distribution in such cases. Sometimes this can be more of an art than a science.

## Coin flipping

Suppose there is a coin that may be biased – this coin has unknown probability  $\theta$  of giving a “heads.” If we repeatedly flip this coin and observe the outcomes, how can we maintain our belief about  $\theta$ ?

Note that the coin-flipping problem can be seen as a simplification of the survey problem we discussed last time, where we assume that people always tell the truth, are sampled uniformly at random, and whose opinions are generated independently (perhaps by flipping a coin!).

Before we select a prior for  $\theta$ , we write down the likelihood. For a particular problem, it is almost always easier to derive an appropriate likelihood than it is to identify an appropriate prior distribution.

Suppose we flip the coin  $n$  times and observe  $x$  “heads.” Every statistician, regardless of philosophy, would agree that the probability of this observation, given the value of  $\theta$ , comes from a binomial distribution:

$$\Pr(x \mid n, \theta) = \binom{n}{x} \theta^x (1 - \theta)^{n-x}.$$

## Classical method

Before we continue with the Bayesian approach, we pause to discuss how a classical statistician would proceed with this problem. Recall that in the frequentist approach, the value  $\theta$  can only be considered in terms of the frequency of success (“heads”) seen during an infinite number of trials. It is not valid in this framework to represent a “belief” about  $\theta$  in terms of probability.

Rather, the frequentist approach to reasoning about  $\theta$  is to construct an *estimator* for  $\theta$ , which in theory can be any function of the observed data:  $\hat{\theta}(x, n)$ . Estimators are then analyzed in terms of their behavior as the number of observations goes to infinity (for example, we might prove that  $\hat{\theta} \rightarrow \theta$  as  $n \rightarrow \infty$ ). The classical estimator in this case is the empirical frequency  $\hat{\theta} = x/n$ .

## Bayesian method

An interesting thing to note about the frequentist approach is that it ignores all prior information, opting instead to only look at the observed data. To a Bayesian, every such problem is different and should be analyzed contextually given the known information.

With the likelihood decided, we must now choose a prior distribution  $p(\theta)$ . A convenient prior in this case is the **beta distribution**, which has two parameters  $\alpha$  and  $\beta$ :

$$p(\theta \mid \alpha, \beta) = \mathcal{B}(\theta; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1}.$$

Here the normalizing constant  $B(\alpha, \beta)$  is the **beta function**:

$$B(\alpha, \beta) = \int_0^1 \theta^{\alpha-1} (1 - \theta)^{\beta-1} d\theta.$$

The support of the beta distribution is  $\theta \in (0, 1)$ , and by selecting various values of  $\alpha$  and  $\beta$ , we can control its shape to represent a variety of different prior beliefs.

Given our observations  $\mathcal{D} = (x, n)$ , we can now compute the posterior distribution of  $\theta$ :

$$p(\theta \mid x, n, \alpha, \beta) = \frac{\Pr(x \mid n, \theta) p(\theta \mid \alpha, \beta)}{\int \Pr(x \mid n, \theta) p(\theta \mid \alpha, \beta) d\theta}.$$

First we handle the normalization constant  $\Pr(x \mid n, \alpha, \beta)$ :

$$\begin{aligned} \int \Pr(x \mid n, \theta) p(\theta \mid \alpha, \beta) d\theta &= \binom{n}{x} \frac{1}{B(\alpha, \beta)} \int_0^1 \theta^{\alpha+x-1} (1-\theta)^{\beta+n-x-1} d\theta \\ &= \binom{n}{x} \frac{B(\alpha+x, \beta+n-x)}{B(\alpha, \beta)}. \end{aligned}$$

Now we apply Bayes theorem:

$$\begin{aligned} p(\theta \mid x, n, \alpha, \beta) &= \frac{\Pr(x \mid n, \theta) p(\theta \mid \alpha, \beta)}{\int \Pr(x \mid n, \theta) p(\theta \mid \alpha, \beta) d\theta} \\ &= \left[ \binom{n}{x} \frac{B(\alpha+x, \beta+n-x)}{B(\alpha, \beta)} \right]^{-1} \left[ \binom{n}{x} \theta^x (1-\theta)^{n-x} \right] \left[ \frac{\theta^{\alpha-1} (1-\theta)^{\beta-1}}{B(\alpha, \beta)} \right] \\ &= \frac{1}{B(\alpha+x, \beta+n-x)} \theta^{\alpha+x-1} (1-\theta)^{\beta+n-x-1} \\ &= \mathcal{B}(\alpha+x, \beta+n-x). \end{aligned}$$

The posterior is therefore another beta distribution with parameters  $(\alpha+x, \beta+n-x)$ ! Specifically, we have added the number of successes to the first parameter and the number of failures to the second.

The rather convenient fact that the posterior remains a beta distribution is because the beta distribution satisfies a property known as **conjugacy** with the binomial likelihood. This fact also leads to a common interpretation of the parameters  $\alpha$  and  $\beta$ : they serve as “pseudocounts,” or fake observations we pretend to have seen before seeing the data.

Figure 1 shows the relevant functions for the coin flipping example for  $(\alpha, \beta) = (3, 5)$  and  $(x, n) = (7, 8)$ . Notice that the likelihood favors higher values of  $\theta$ , whereas the prior had favored lower values of  $\theta$ . The posterior, taking into account both sources of information, lies in between these extremes. Notice also that the posterior has support over a narrower range of plausible  $\theta$  values than the prior; this is because we can draw more confident conclusions from having access to more information.

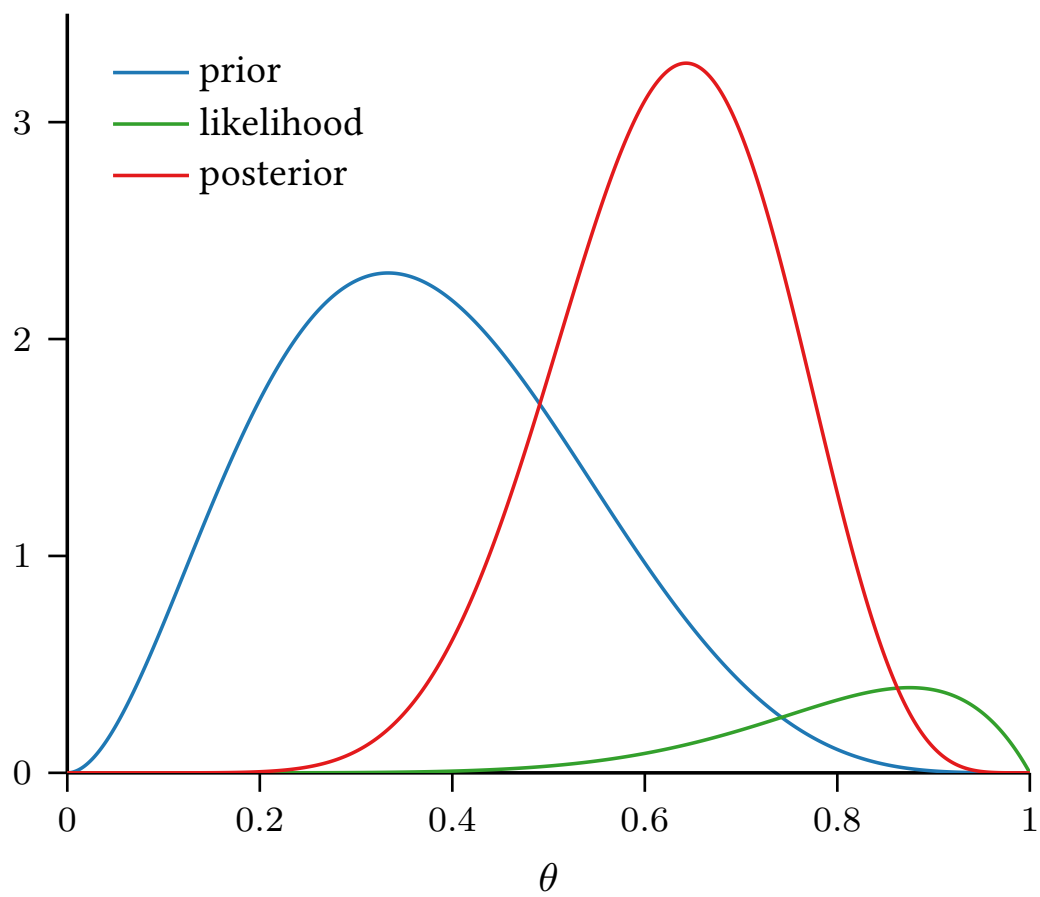


Figure 1: An example of Bayesian updating for coin flipping.

## Hypothesis testing

We often wish to use our observed data to draw conclusions about the plausibility of various hypotheses. Returning to our example of coin flipping, we might wish to know whether the parameter  $\theta$  is say less than  $1/2$ . The Bayesian method allows us to compute this value directly from the posterior distribution:

$$\Pr(\theta < 1/2 \mid x, n, \alpha, \beta) = \int_0^{1/2} p(\theta \mid x, n, \alpha, \beta) d\theta.$$

### Classical method

There is a sharp contrast between the simplicity of the approach above and the frequentist method. The classical approach to hypothesis testing uses the likelihood as a way of generating fake datasets of the same size as the observations. The likelihood then serves as a so-called “null hypothesis” that allows us to generate hypothetical datasets under some condition.

From these, we compute *statistics*, which, like estimators, can be any function of the hypothesized data. We then identify some **critical set  $C$**  for this statistic which contains some large portion  $(1 - \alpha)$  of the values corresponding to the datasets generated by our null hypothesis. If the statistic computed from the observed data falls outside this set, we reject the null hypothesis with “**confidence**”  $\alpha$ . Note that the **“rejection” of the null hypothesis in classical hypothesis testing** is purely a statement about the observed data (that it looks “unusual”), and not about the plausibility of alternative hypotheses!

### Bayesian method

Suppose we are reasoning about a parameter  $\theta$  in light of data  $\mathcal{D}$ , and wish to consider a hypothesis of the form  $\theta \in \mathcal{H}$ , where  $\mathcal{H} \subseteq \Theta$  is some set of possible values for this parameter.

We have seen that the Bayesian approach to hypothesis testing is straightforward. We first derive the posterior distribution  $p(\theta \mid \mathcal{D})$  and then may compute the probability of the hypothesis directly:

$$\Pr(\theta \in \mathcal{H} \mid \mathcal{D}) = \int_{\mathcal{H}} p(\theta \mid \mathcal{D}) d\theta.$$

## Coin fairness

Let’s consider an explicit example. Suppose we are interested in the unknown bias of a coin  $\theta \in (0, 1)$ , and begin with the uniform prior on the interval  $(0, 1)$ :

$$p(\theta) = \mathcal{U}(\theta; 0, 1) = \mathcal{B}(\theta; \alpha = 1, \beta = 1).$$

Let’s collect some data to further inform our belief about  $\theta$ . Suppose we flip the coin independently  $n = 50$  times and observe  $x = 30$  heads. After gathering this data, we wish to consider the natural question of whether the coin is fair: that is, whether  $\theta = 1/2$ .

From the results in the previous lecture, we can compute the posterior distribution easily. It is an updated beta distribution:

$$p(\theta \mid \mathcal{D}) = \mathcal{B}(\theta; 31, 21).$$

We may now compute the posterior probability of the hypothesis that the coin is fair:

$$\Pr(\theta = 1/2 \mid \mathcal{D}) = \int_{1/2}^{1/2} p(\theta \mid \mathcal{D}) d\theta = 0.$$

The posterior probability of the coin being *exactly* fair is zero! This should not be surprising, as suggesting that we could possibly know the bias of the coin with infinite precision is unfathomable.

We may however relax the question a bit to get some more insight. One option would be to consider a parameterized family of hypotheses of the form

$$\mathcal{H}(\varepsilon) = (1/2 - \varepsilon, 1/2 + \varepsilon).$$

Thus a high probability of the hypothesis  $\mathcal{H}(\varepsilon)$  corresponds to the notion that the coin is “near fair” with an allowed error of  $\varepsilon$ . We may then compute the posterior probability of these hypotheses and consider how they vary as a function of  $\varepsilon$ . Figure 1 shows the results for the coin-flipping example above. We can see that there’s approximately a 50% posterior probability that the bias of the coin is in the interval  $(0.4, 0.6)$ , corresponding to  $\varepsilon = 0.1$ . We also have evidence to conclude  $\theta \in (0.25, 0.75)$  with near certainty. These probabilities help constrain exactly how “fair” or “not fair” we believe the coin to be in light of our evidence.

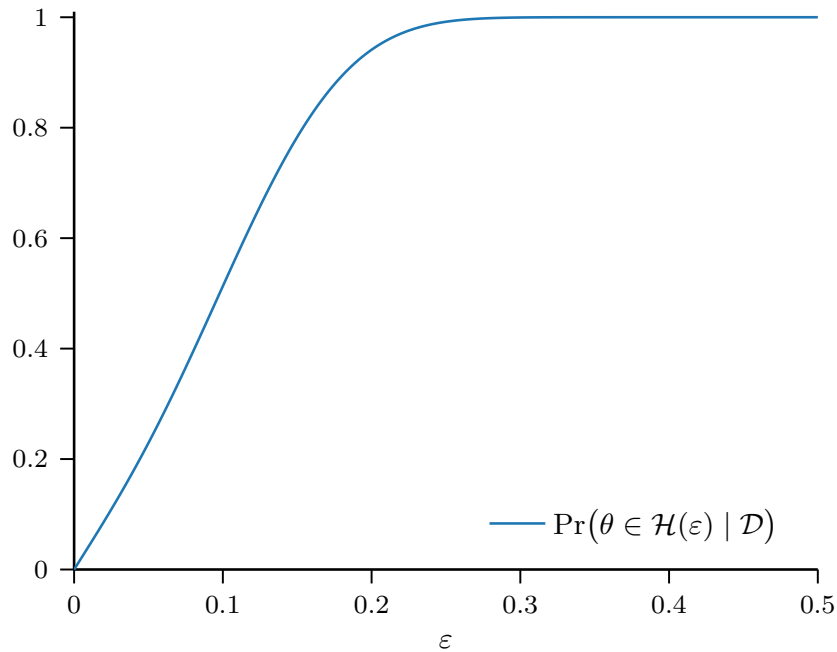


Figure 1: The posterior probability of the hypotheses  $\mathcal{H}(\varepsilon)$  for  $0 < \varepsilon < 1/2$ .

The classical approach to testing this hypothesis is to create a so-called “null hypothesis”  $\mathcal{H}_0$  that serves to define what “typical” data may look like assuming that hypothesis. For example, for reasoning about the fairness of a coin, we may choose the natural null hypothesis  $\mathcal{H}_0: \theta = 1/2$ . Now we can use the likelihood

$$\Pr(x \mid n, \theta = 1/2)$$

to reason about what observed data would look like if this hypothesis were true. This is a critical point: the null hypothesis exists to define what sort of data we would expect to see under an assumed value of  $\theta$ .



The classical procedure would then define a statistic summarizing a given dataset  $s(\mathcal{D})$  in some way. An example for coin flipping would be the sample mean  $s(\mathcal{D}) = \hat{\theta} = x/n$ . This happens to be a common estimator for  $\theta$  as well, but this is a coincidence. We now compute a so-called *critical set*  $C(\alpha)$  with the property

$$\Pr(s(\mathcal{D}) \in C(\alpha) \mid \mathcal{H}_0) = 1 - \alpha,$$


where  $\alpha$  is called the *significance level* of the test. The interpretation of the critical set is that the statistic computed from datasets generated assuming the null hypothesis “usually” have values in this range.

Finally, we compute the statistic for a particular set of observed data and determine whether it falls inside the critical set  $C(\alpha)$  we have defined. If so, the dataset appears, according to the statistic, typical for datasets generated from the null hypothesis. If not, the dataset appears unusual, in the sense that data generated assuming the null hypothesis would have such extreme values of the statistic only a small portion of the time ( $100\alpha\%$ ). In this case, you “reject” the null hypothesis with significance  $1 - \alpha$ .

What is a *p-value*? It must be the probability that the null hypothesis is true, right? Absolutely not! The null hypothesis cannot be associated with a probability in the classical interpretation of probability. A *p-value* is actually the minimum  $\alpha$  for which you would reject the null hypothesis using this procedure. That is, a *p-value* is not the probability that the null hypothesis is true, but rather the probability that we would observe results as extreme as those in our dataset, as measured by the chosen statistic, *if the null hypothesis were true!* The *p-value* is thus only a probability that is well-defined when already assuming the null hypothesis to be true. A *p-value* does *not* say how extreme our results would appear under alternative hypotheses.

Bayesian model selection will eventually allow us to explicitly quantify the plausibility of a collection of models having generated the observed data.

To interpret the above procedure in the frequency interpretation of probability, the critical sets are constructed by reasoning about the following experiment:

- 
- generate  $\mathcal{D}$  assuming  $\mathcal{H}_0$ ;
  - compute  $s(\mathcal{D})$ ;
  - state  $s(\mathcal{D}) \in C(\alpha)$ .

In the limit of infinitely many repetitions of this experiment, the final claim will be true exactly  $100(1 - \alpha)\%$  of the time. Recall this is the definition of probability in this context: the frequency of occurrence in the limit of infinitely many trials. Note that the experiment we repeat here *includes generating data from the null hypothesis* as its first step! This is not the experiment we are conducting, since we have a dataset in front of us that we want to analyze, which may have been generated in any number of ways.

## Summarizing Distributions

In the Bayesian method, the posterior distribution  $p(\theta \mid \mathcal{D})$  is the main object of interest and contains all relevant information about  $\theta$  in light of the observations  $\mathcal{D}$ . A natural task is to provide a summary of the posterior distribution, for example to efficiently convey its relevant properties.

In the next lecture we will consider point estimation, which is one common summarization method. Another commonly considered problem is **interval summarization**, where we provide an interval  $(\ell, u)$  indicating plausible values of the parameter  $\theta$  in light of the observed data. Classical interval estimates are known as *confidence intervals*, and we will discuss them in more detail shortly.

The Bayesian approach to interval estimation is straightforward. Again we use the posterior distribution  $p(\theta \mid \mathcal{D})$  to guide the construction of an interval summary. If we can find an interval  $(\ell, u)$  such that the posterior probability that  $\theta \in (\ell, u)$  is “large” (say, has probability  $\alpha$ ):


$$\Pr(\theta \in (\ell, u) \mid \mathcal{D}) = \int_{\ell}^u p(\theta \mid \mathcal{D}) \, d\theta = \alpha,$$

then we call  $(\ell, u)$  an  **$\alpha$ -credible interval** for  $\theta$ . Note the parallel in this definition to our treatment of hypothesis testing above! Effectively, an  $\alpha$ -credible interval is simply a hypothesis that has posterior probability equal to  $\alpha$  and happens to take the form of an interval.

Examining our coin flipping example from before, we can construct some credible intervals immediately from the data in Figure 1. We have that  $\mathcal{H}(\varepsilon = 0.1) = (0.4, 0.6)$  is a 50%-credible interval for the bias of the coin, and  $\mathcal{H}(\varepsilon = 0.2) = (0.3, 0.7)$  is a 95%-credible interval. The slightly wider interval  $\mathcal{H}(\varepsilon = 0.25) = (0.25, 0.75)$  represents a very high probability credible interval, corresponding to  $\alpha > 99\%$ .

It is clear from the definition that multiple intervals (in fact, often uncountably many) can serve as a credible interval for a particular value of  $\alpha$ . Exactly which interval should we construct to summarize a given distribution? This is a question for which we will need to develop Bayesian decision theory before we can continue, which we will discuss in the next lecture. In short, we will first need to quantify how “desirable” a given credible interval is in some way, then select the one maximizing this measure. For example, we may want to construct the narrowest possible interval, or we may wish it to be centered on a particular point (such as the posterior mean, median, or mode), or we may wish the interval to have some other property.

The classical approach to interval summarization is to construct a so-called **confidence interval** for the parameter of interest  $\theta$ . Again a confidence interval is described in terms of repeating a particular experiment infinitely many times. The experiment we consider will proceed as follows. First we are going to define a function  $\text{CI}(\mathcal{D})$  that will map a given dataset  $\mathcal{D}$  to an interval  $(\ell, u) = \text{CI}(\mathcal{D})$ . Now we consider repeating the following experiment:

- 
- collect data  $\mathcal{D}$
  - compute the interval  $(\ell, u) = \text{CI}(\mathcal{D})$
  - state  $\theta \in (\ell, u)$ .

In the limit of infinitely many repetitions of this experiment, if the final statement is true with probability  $\alpha$ , then the procedure  $\text{CI}(\mathcal{D})$  is called an  $\alpha$ -confidence interval procedure, and we will write  $\text{CI}(\mathcal{D}; \alpha)$  to indicate the confidence level  $\alpha$  when required.

This might sound like exactly the same definition as a Bayesian credible interval. For example, if we have an  $\alpha$ -confidence interval procedure available, then when we plug in a given dataset  $\mathcal{D}$ , we must have

$$\Pr(\theta \in \text{CI}(\mathcal{D}; \alpha) \mid \mathcal{D}) = \alpha, \quad (\star)$$

right? **No!** This interpretation is widespread, but it is wrong. This conclusion is sometimes known as the **fundamental confidence fallacy**,<sup>1</sup> and confuses the nature of prior information with that of posterior information. Namely, note that the experiment we consider when defining the confidence interval procedure *includes gathering a random dataset* as its first step. All we know is that if we repeat the confidence interval procedure on *infinitely many datasets*, that it will succeed with probability  $\alpha$ . However, we usually have only one particular dataset in front of us to analyze that we care about, and we cannot say anything about the interval produced for this dataset in isolation.

Here is a simple example that shows how this erroneous conclusion can go wrong. Suppose we are going to observe two values  $x_1, x_2 \in \mathbb{R}$  generated independently from some unknown distribution  $p(x)$  and wish to construct a confidence interval for the mean of the distribution generating the data,  $\theta = \mathbb{E}[x]$ . Consider the following procedure:

$$\text{CI}(\mathcal{D}) = \begin{cases} (-\infty, \infty) & x_1 < x_2 \\ \emptyset & x_2 \geq x_1. \end{cases}$$

Obviously this trivial map is a 50%-confidence interval procedure! Because the values are generated independently,  $x_1$  will be the lesser value exactly 50% of the time. In this case, the absurdly large interval produced will *definitely* contain  $\theta$ . The other 50% of the time, the interval will be empty, and *definitely will not* contain  $\theta$ . Therefore the procedure succeeds exactly 50% of the time. However, in half the cases, the posterior probability that  $\theta$  is inside the interval produced is 100%, and otherwise this probability is 0%. In no case is this probability equal to the confidence level.

Another fallacy in the interpretation of confidence intervals is the so called **precision fallacy**, that shorter confidence intervals indicate the data provide more precise information about  $\theta$ . A striking illustration of this fallacy is provided by the “lost submarine” example provided by Morey, et al. in the reference given below. I encourage you to read this paper and reflect!

---

<sup>1</sup>See the following reference for some excellent extended discussion on confidence intervals: Richard D. Morey, et al. (2015). The fallacy of placing confidence in confidence intervals. *Psychonomic Bulletin & Review* 23(1): 103–123

## Point estimation

Suppose we are interested in the value of a parameter  $\theta$ , for example the unknown bias of a coin. We have already seen how one might use the Bayesian method to reason about  $\theta$ ; namely, we select a likelihood function  $p(\mathcal{D} \mid \theta)$ , explaining how observed data  $\mathcal{D}$  are expected to be generated given the value of  $\theta$ . Then we select a prior distribution  $p(\theta)$  reflecting our initial beliefs about  $\theta$ . Finally, we conduct an experiment to gather data and use Bayes' theorem to derive the posterior  $p(\theta \mid \mathcal{D})$ .

In a sense, the posterior contains all information about  $\theta$  that we care about. However, the process of inference will often require us to use this posterior to answer various questions. For example, we might be compelled to choose a single value  $\hat{\theta}$  to serve as a **point estimate** of  $\theta$ . To a Bayesian, the selection of  $\hat{\theta}$  is a **decision**, and in different contexts we might want to select different values to report.

In general, we should not expect to be able to select the true value of  $\theta$ , unless we have somehow observed data that unambiguously determine it. Instead, we can only hope to select an estimate that is "close" to the true value. Different definitions of "closeness" can naturally lead to different estimates. The Bayesian approach to point estimation will be to analyze the impact of our choice in terms of a **loss function**, which describes how "bad" different types of mistakes can be. We then select the estimate which appears to be the least "bad" according to our current beliefs about  $\theta$ .

## Decision theory

As mentioned above, the selection of an estimate  $\hat{\theta}$  can be seen as a decision. It turns out that the Bayesian approach to decision theory is rather simple and consistent, so we will introduce it in an abstract form here.

A decision problem will typically have three components. First, we have a **parameter space** (also called a **state space**)  $\Theta$ , with an unknown value  $\theta \in \Theta$ . We will also have a **sample space**  $\mathcal{X}$  representing the potential observations we could theoretically make. Finally, we have an **action space**  $\mathcal{A}$  representing the potential actions we may select from. In the point estimation problem, the potential actions  $\hat{\theta}$  are exactly those in the parameter space, so  $\mathcal{A} = \Theta$ . This might not always be the case, however. Finally, we will have a likelihood function  $p(\mathcal{D} \mid \theta)$  linking potential observations to the parameter space.

After conducting an experiment and observing data, we are compelled to select an action  $a \in \mathcal{A}$ . We define a (deterministic) **decision rule** as a function  $\delta: \mathcal{X} \rightarrow \mathcal{A}$  that selects an action  $a$  given the observations  $\mathcal{D}$ . In general, we may also consider randomized decision rules, where  $\delta$  maps observed data  $\mathcal{D}$  to a probability distribution over  $\mathcal{A}$ , which we select a sample from. This can be useful, for example, when facing an intelligent adversary. Most of the expressions we will derive can be derived analogously for this case; however, we will not do so here.

This decision rule  $\delta$  can be any arbitrary function. How do we select which decision rule to use? To guide our selection, we will define a **loss function**, which is a function  $L: \Theta \times \mathcal{A} \rightarrow \mathbb{R}$ . The value  $L(\theta, a)$  summarizes "how bad" an action  $a$  was if the true value of the parameter was revealed to be  $\theta$ ; larger losses represent worse outcomes. Ideally, we would select the action that minimizes this loss, but unfortunately we will never know the exact value of  $\theta$ ; complicating our decision.

As usual, there are two main approaches to designing decision rules. We begin with the Bayesian approach.

### Bayesian decision theory

The Bayesian approach to decision theory is straightforward. Given our observed data  $\mathcal{D}$ , we find the posterior  $p(\theta \mid \mathcal{D})$ , which represents our current belief about the unknown parameter  $\theta$ . Given a potential action  $a$ , we may define the **posterior expected loss** of  $a$  by averaging the loss function over the unknown parameter:

$$\ell(p(\theta \mid \mathcal{D}), a) = \mathbb{E}[L(\theta, a) \mid \mathcal{D}] = \int_{\Theta} L(\theta, a) p(\theta \mid \mathcal{D}) d\theta.$$

Because the posterior expected loss of each action  $a$  is a scalar value, it defines a total order on the action space  $\mathcal{A}$ . When there is an action minimizing the posterior expected loss, it is the natural choice to make:

$$\delta^*(\mathcal{D}) = \arg \min_{a \in \mathcal{A}} \ell(p(\theta \mid \mathcal{D}), a),$$

representing the action with the lowest expected loss, given our current beliefs about  $\theta$ . Note that  $\delta^*(\mathcal{D})$  may not be unique, in which case we can select any action attaining the minimal value. Any minimizer of the posterior expected loss is called a **Bayes action**. The value  $\delta^*(\mathcal{D})$  may also be found by solving the equivalent minimization problem

$$\delta^*(\mathcal{D}) = \arg \min_{a \in \mathcal{A}} \int_{\Theta} L(\theta, a) p(\mathcal{D} \mid \theta) p(\theta) d\theta;$$

the advantage of this formulation is that it avoids computing the normalization constant  $p(\mathcal{D}) = \int p(\mathcal{D} \mid \theta) p(\theta) d\theta$  in the posterior.

Notice that a Bayes action is tied to a particular set of observed data  $\mathcal{D}$ . This does not limit its utility terribly; after all, we will always have a particular set of observed data at hand when making a decision. However we may extend the notion of Bayes actions in a natural way to define an entire decision rule. We define the **Bayes rule**, a decision rule, by simply always selecting a (not necessarily unique) Bayes action given the observed data. Note that the second formulation above can often be minimized even when  $p(\theta)$  is not necessarily a probability distribution (such priors are called *improper* but are often encountered in practice). A decision rule derived in this way from an improper prior is called a **generalized Bayes rule**.

In the case of point estimation, the decision rule  $\delta$  may be more naturally written  $\hat{\theta}(\mathcal{D})$ . Then, as above, the point estimation problem reduces to selecting a loss function and deriving the decision rule  $\hat{\theta}$  that minimizes the expected loss at every point. A decision rule that minimizes posterior expected loss for every possible set of observations  $\mathcal{D}$  is called a **Bayes estimator**.

We may derive Bayes estimators for some common loss functions. As an example, consider the common loss function

$$L(\theta, \hat{\theta}) = (\theta - \hat{\theta})^2,$$

which represents the squared distance between our estimate and the true value. For the squared loss, we may compute:

$$\begin{aligned} \mathbb{E}[L(\theta, \hat{\theta}) \mid \mathcal{D}] &= \int (\theta - \hat{\theta})^2 p(\theta \mid \mathcal{D}) d\theta \\ &= \int \theta^2 p(\theta \mid \mathcal{D}) d\theta - 2\hat{\theta} \int \theta p(\theta \mid \mathcal{D}) d\theta + \hat{\theta}^2 \int p(\theta \mid \mathcal{D}) d\theta \\ &= \int \theta^2 p(\theta \mid \mathcal{D}) d\theta - 2\hat{\theta} \mathbb{E}[\theta \mid \mathcal{D}] + \hat{\theta}^2 \end{aligned}$$

We may minimize this expression by differentiating with respect to  $\hat{\theta}$  and equating to zero:

$$\frac{\partial \mathbb{E}[L(\theta, \hat{\theta}) | \mathcal{D}]}{\partial \hat{\theta}} = -2\mathbb{E}[\theta | \mathcal{D}] + 2\hat{\theta} = 0$$

from which we may derive  $\hat{\theta} = \mathbb{E}[\theta | \mathcal{D}]$ . Examining the second derivative, we see

$$\frac{\partial^2 \mathbb{E}[L(\theta, \hat{\theta}) | \mathcal{D}]}{\partial \hat{\theta}^2} = 2 > 0,$$

so this is indeed a minimum. Therefore we have shown that the Bayes estimator in the case of squared loss is the posterior mean  $\hat{\theta}(\mathcal{D}) = \mathbb{E}[\theta | \mathcal{D}]$ .

A similar analysis shows that the Bayes estimator for the absolute deviation loss  $L(\theta, \hat{\theta}) = |\theta - \hat{\theta}|$  is the posterior median, and the Bayes estimators for a relaxed 0-1 loss:

$$L(\theta, \hat{\theta}; \varepsilon) = \begin{cases} 0 & |\theta - \hat{\theta}| < \varepsilon; \\ 1 & |\theta - \hat{\theta}| \geq \varepsilon, \end{cases}$$

converge to the posterior mode for small  $\varepsilon$ .

The posterior mode, also called the **maximum a posteriori** (MAP) estimate of  $\theta$  and written  $\hat{\theta}_{\text{MAP}}$ , is a rather common estimator in practice. The reason is that optimization is almost always easier than integration. In particular, we may find the MAP estimate by maximizing the *unnormalized* posterior

$$\hat{\theta}_{\text{MAP}} = \arg \max_{\theta} p(\mathcal{D} | \theta)p(\theta),$$

where we have avoided computing the normalization constant  $p(\mathcal{D}) = \int p(\mathcal{D} | \theta)p(\theta) d\theta$ .


### Frequentist decision theory

The frequentist approach to decision theory is somewhat different. As usual, in classical statistics it is not allowed to place a prior distribution on a parameter such as  $\theta$ ; rather, it is much more common to use the likelihood  $p(\mathcal{D} | \theta)$  to hypothesize what data might look like for different values of  $\theta$ , and use this analysis to drive your action.

The frequentist approach to decision theory involves the notion of **risk functions**. The *frequentist risk* of a decision function  $\delta$  is defined by

$$R(\theta, \delta) = \int_{\mathcal{X}} L(\theta, \delta(\mathcal{D}))p(\mathcal{D} | \theta) d\mathcal{D},$$

that is, it represents the expected loss incurred when repeatedly using the decision rule  $\delta$  on different datasets  $\mathcal{D}$  as a function of the unknown parameter  $\theta$ .

 To a Bayesian, the frequentist risk is a very strange notion: we know the exact value of our data  $\mathcal{D}$  when we make our decision, so why should we average over other datasets that we haven't seen? The frequentist counterargument to this is typically that we might know  $\mathcal{D}$  but can't know  $p(\theta)$ !

Notice that whereas the posterior expected loss was a scalar defining a total order on the action space  $\mathcal{A}$ , which could be extended to naturally define an entire decision rule, the frequentist risk is a function of  $\theta$  and the entire decision rule  $\delta$ . It is very unusual for there to be a single decision rule  $\delta$  that works the best for every potential value of  $\theta$ . For this reason, we must decide on some mechanism to use the frequentist risk to select a "good" decision rule. There are many proposed mechanisms for doing so, but we will simply quickly describe two below.

### Bayes risk

One solution to the problem of comparing decision rules is to place a prior distribution on the unknown  $\theta$  and compute the average risk:

$$r(p(\theta), \delta) = \mathbb{E}[R(\theta, \delta)] = \int_{\Theta} R(\theta, \delta) p(\theta) d\theta = \int_{\Theta} \int_{\mathcal{X}} L(\theta, \delta(\mathcal{D})) p(\mathcal{D} | \theta) p(\theta) d\theta d\mathcal{D}.$$

The function  $r(p(\theta), \delta)$  is called the *Bayes risk* of  $\delta$  under the prior  $p(\theta)$ . Again, the Bayes risk is scalar-valued, so we induce a total order on all decision rules, making identifying a unique decision rule easier. Any  $\delta$  minimizing Bayes risk is called a *Bayes rule*. We have seen this term before! It turns out that given a prior  $p(\theta)$ , the Bayesian procedure described above for defining a decision function by selecting an action with minimum posterior expected loss is guaranteed to minimize Bayes risk and therefore produce a Bayes rule with respect to  $p(\theta)$ . Note, however, that it is unusual in the Bayesian perspective to first find an entire decision rule  $\delta$  and then apply it to a particular dataset  $\mathcal{D}$ . Instead, it is almost always easier to minimize the expected posterior loss only at the actual observed data. After all, why would we need to know what decision we would make with other data?


### Admissibility

Another criterion for selecting between decision rules in the frequentist framework is called *admissibility*. In short, it is often difficult to identify a single best decision rule, but it can sometimes be easy to discard some bad ones, for example if they can be shown to always be no better than (and sometimes worse than) another rule.

Let  $\delta_1$  and  $\delta_2$  be two decision rules. We say that  $\delta_1$  *dominates*  $\delta_2$  if:

- $R(\theta, \delta_1) \leq R(\theta, \delta_2)$  for all  $\theta \in \Theta$ , and
- there exists at least one  $\theta$  for which  $R(\theta, \delta_1) < R(\theta, \delta_2)$ .

If there is a decision rule  $\delta$  that is not dominated by any other rule, it is called *admissible*. One interesting result tying Bayesian and frequentist decision theory is the following:

- 
- Every Bayes rule is admissible.
  - Every admissible decision rule is a generalized Bayes rule for some (possibly improper) prior  $p(\theta)$ .

So, in a sense, all admissible frequentist decision rules can be equivalently derived from a Bayesian perspective.

## Examples

Here we give two quick examples of applying Bayesian decision theory.

### Classification with 0-1 loss

Suppose our observations are of the form  $(x, y)$ , where  $x$  is an arbitrary input, and  $y \in \{0, 1\}$  is a binary label associated with  $x$ . In classification, our goal is to predict the label  $y'$  associated with a new input  $x'$ . The Bayesian approach is to derive a model giving probabilities  $\Pr(y' = 1 \mid x', \mathcal{D})$ . Suppose this model is provided for you. Notice that this model is not conditioned on any additional parameters  $\theta$ ; we have integrated them out via

$$\Pr(y' = 1 \mid x', \mathcal{D}) = \int \Pr(y' = 1 \mid x', \mathcal{D}, \theta) p(\theta \mid \mathcal{D}) d\theta.$$

Given a new datapoint  $x'$ , which label  $a$  should we predict? Notice that the prediction of a label is actually a *decision*. Here our action space is  $\mathcal{A} = \{0, 1\}$ , enumerating the two labels we can predict. Our parameter space is the same: the only uncertainty we have is the unknown label  $y'$ .

Let us suppose a simple loss function for this problem:

$$L(y', a) = \begin{cases} 0 & a = y'; \\ 1 & a \neq y'. \end{cases}$$

This loss function, called the *0-1 loss*, is common in classification problems. We pay a constant loss for every mistake we make. In this case, the expected loss of each possible action is simple to compute:

$$\begin{aligned} \mathbb{E}[L(y', a = 1) \mid x', \mathcal{D}] &= \Pr(y' = 0 \mid x', \mathcal{D}); \\ \mathbb{E}[L(y', a = 0) \mid x', \mathcal{D}] &= \Pr(y' = 1 \mid x', \mathcal{D}). \end{aligned}$$

The Bayes action is then to predict the class with the highest probability. This is not so surprising. Notice that if we change the loss to have different costs of mistakes (so that  $L(0, 1) \neq L(1, 0)$ ), then the Bayes action might compel us to select the less-likely class to avoid a potentially high loss for misclassification!

### Hypothesis testing

The Bayesian approach to hypothesis testing is also rather straightforward. A *hypothesis* is simply a subset of the parameter space  $\mathcal{H} \subseteq \Theta$ . The Bayesian approach allows one to compute the posterior probability of a hypothesis directly:

$$\Pr(\theta \in \mathcal{H} \mid \mathcal{D}) = \int_{\mathcal{H}} p(\theta \mid \mathcal{D}) d\theta.$$

Now, equipped with a loss function, the posterior expected loss framework above can be applied directly to select between different hypotheses.



## The Gaussian distribution

Probably the most-important distribution in all of statistics is the **Gaussian distribution**, also called the **normal distribution**. The Gaussian distribution arises in many contexts and is widely used for modeling continuous random variables.

The probability density function of the univariate (one-dimensional) Gaussian distribution is

$$p(x \mid \mu, \sigma^2) = \mathcal{N}(x; \mu, \sigma^2) = \frac{1}{Z} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right).$$

The normalization constant  $Z$  is

$$Z = \sqrt{2\pi\sigma^2}.$$

The parameters  $\mu$  and  $\sigma^2$  specify the mean and variance of the distribution, respectively:

$$\mu = \mathbb{E}[x]; \quad \sigma^2 = \text{var}[x].$$

Figure 1 shows the probability density function for several sets of parameters  $(\mu, \sigma^2)$ . The distribution is symmetric around the mean and most of the density ( $\approx 99.7\%$ ) is contained within  $\pm 3\sigma$  of the mean.

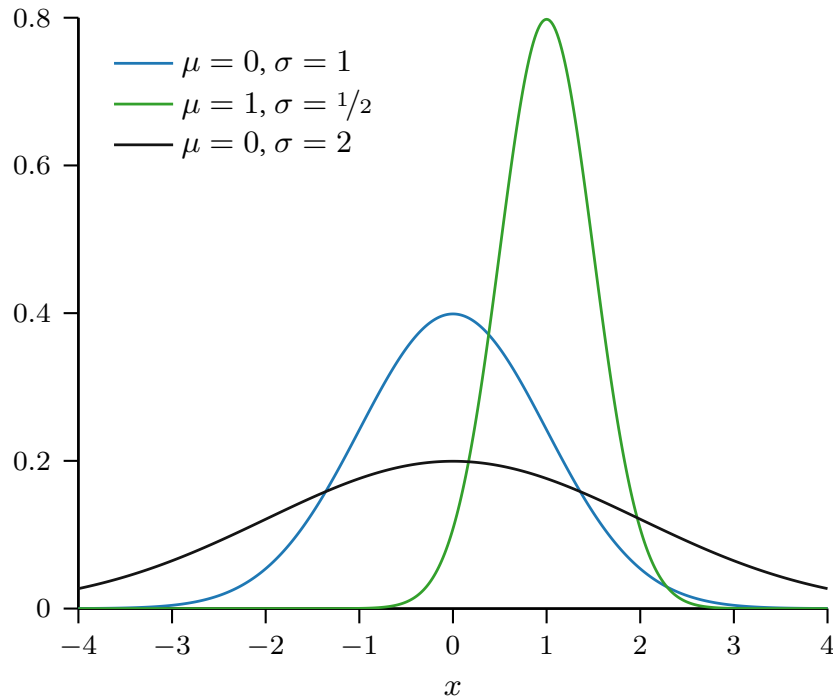


Figure 1: Examples of univariate Gaussian PDFs  $\mathcal{N}(x; \mu, \sigma^2)$ .

We can extend the univariate Gaussian distribution to a distribution over  $d$ -dimensional vectors, producing the multivariate analog. The probability density function of the multivariate Gaussian distribution is

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{Z} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right).$$

The normalization constant  $Z$  is

$$Z = \sqrt{\det(2\pi\boldsymbol{\Sigma})} = (2\pi)^{d/2} (\det \boldsymbol{\Sigma})^{1/2}.$$

Examining these equations, we can see that the multivariate density coincides with the univariate density in the special case when  $\boldsymbol{\Sigma}$  is the scalar  $\sigma^2$ .

Again, the vector  $\boldsymbol{\mu}$  specifies the mean of the multivariate Gaussian distribution. The matrix  $\boldsymbol{\Sigma}$  specifies the **covariance** between each pair of variables in  $\mathbf{x}$ :

$$\boldsymbol{\Sigma} = \text{cov}(\mathbf{x}, \mathbf{x}) = \mathbb{E}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^\top].$$

Covariance matrices are necessarily symmetric and **positive semidefinite**, which means their eigenvalues are nonnegative. Note that the density function above requires that  $\boldsymbol{\Sigma}$  be **positive definite**, or have strictly positive eigenvalues. A zero eigenvalue would result in a determinant of zero, making the normalization impossible.

The dependence of the multivariate Gaussian density on  $\mathbf{x}$  is entirely through the value of the quadratic form

$$\Delta^2 = (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}).$$

The value  $\Delta$  (obtained via a square root) is called the **Mahalanobis distance**, and can be seen as a generalization of the  $Z$ -score  $\frac{(x-\mu)}{\sigma}$ , often encountered in statistics.

To understand the behavior of the density geometrically, we can set the Mahalanobis distance to a constant. The set of points in  $\mathbb{R}^d$  satisfying  $\Delta = c$  for any given value  $c > 0$  is an ellipsoid with the eigenvectors of  $\boldsymbol{\Sigma}$  defining the directions of the principal axes.

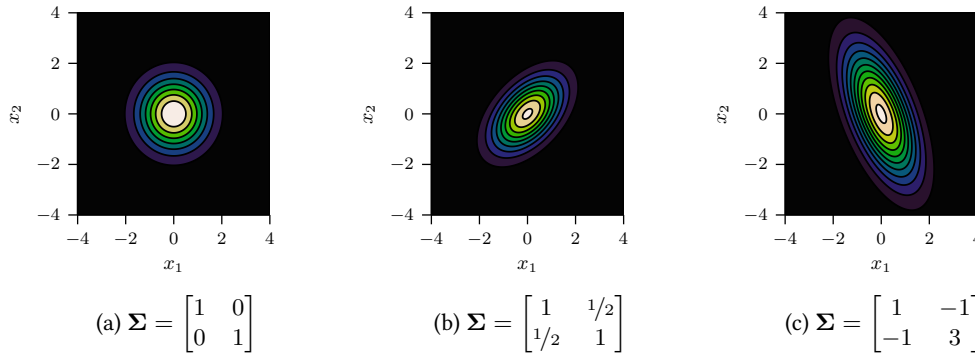


Figure 2: Contour plots for example bivariate Gaussian distributions. Here  $\boldsymbol{\mu} = \mathbf{0}$  for all examples.

Figure 2 shows contour plots of the density of three bivariate (two-dimensional) Gaussian distributions. The elliptical shape of the contours is clear.

The Gaussian distribution has a number of convenient analytic properties, some of which we describe below.

### Marginalization

Often we will have a set of variables  $\mathbf{x}$  with a joint multivariate Gaussian distribution, but only be interested in reasoning about a subset of these variables. Suppose  $\mathbf{x}$  has a multivariate Gaussian distribution:

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

Let us partition the vector into two components:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}.$$

We partition the mean vector and covariance matrix in the same way:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}.$$

Now the marginal distribution of the subvector  $\mathbf{x}_1$  has a simple form:

$$p(\mathbf{x}_1 \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_1, \mu_1, \Sigma_{11}),$$

so we simply pick out the entries of  $\boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$  corresponding to  $\mathbf{x}_1$ .

Figure 3 illustrates the marginal distribution of  $x_1$  for the joint distribution shown in Figure 2(c).

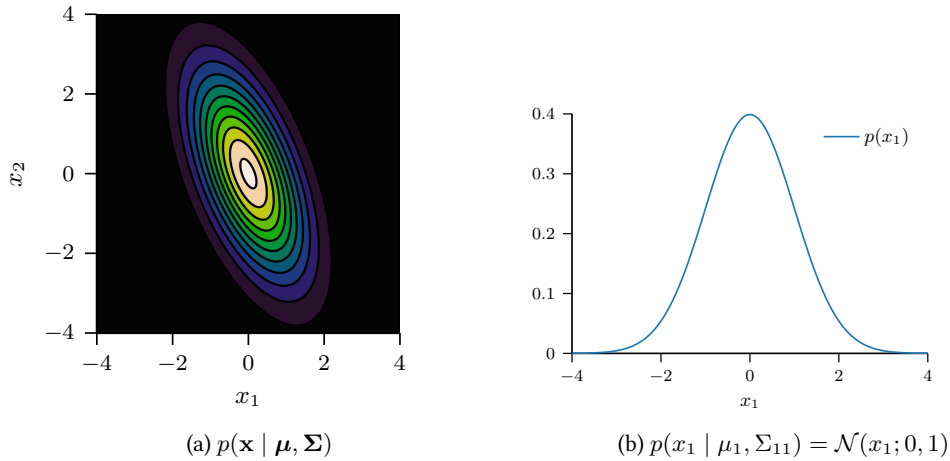


Figure 3: Marginalization example. (a) shows the joint density over  $\mathbf{x} = [x_1, x_2]^\top$ ; this is the same density as in Figure 2(c). (b) shows the marginal density of  $x_1$ .

### Conditioning

Another common scenario will be when we have a set of variables  $\mathbf{x}$  with a joint multivariate Gaussian prior distribution, and are then told the value of a subset of these variables. We may then condition our prior distribution on this observation, giving a posterior distribution over the remaining variables.

Suppose again that  $\mathbf{x}$  has a multivariate Gaussian distribution:

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

and that we have partitioned as before:  $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]^\top$ . Suppose now that we learn the exact value of the subvector  $\mathbf{x}_2$ . Remarkably, the posterior distribution

$$p(\mathbf{x}_1 \mid \mathbf{x}_2, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

is a Gaussian distribution! The formula is

*then* 
$$p(\mathbf{x}_1 \mid \mathbf{x}_2, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_1; \boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{11|2}),$$

with

$$\begin{aligned}\boldsymbol{\mu}_{1|2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12} \boldsymbol{\Sigma}_{22}^{-1} (\mathbf{x}_2 - \boldsymbol{\mu}_2); \\ \boldsymbol{\Sigma}_{11|2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12} \boldsymbol{\Sigma}_{22}^{-1} \boldsymbol{\Sigma}_{21}.\end{aligned}$$

So we adjust the mean by an amount dependent on: (1) the covariance between  $\mathbf{x}_1$  and  $\mathbf{x}_2$ ,  $\boldsymbol{\Sigma}_{12}$ , (2) the prior uncertainty in  $\mathbf{x}_2$ ,  $\boldsymbol{\Sigma}_{22}$ , and (3) the deviation of the observation from the prior mean,  $(\mathbf{x}_2 - \boldsymbol{\mu}_2)$ . Similarly, we reduce the uncertainty in  $\mathbf{x}_1$ ,  $\boldsymbol{\Sigma}_{11}$ , by an amount dependent on (1) and (2). Notably, the reduction of the covariance matrix does *not* depend on the values we observe.

Notice that if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are independent, then  $\boldsymbol{\Sigma}_{12} = \mathbf{0}$ , and the conditioning operation does not change the distribution of  $\mathbf{x}_1$ , as expected.

Figure 4 illustrates the conditional distribution of  $x_1$  for the joint distribution shown in Figure 2(c), after observing  $x_2 = 2$ .

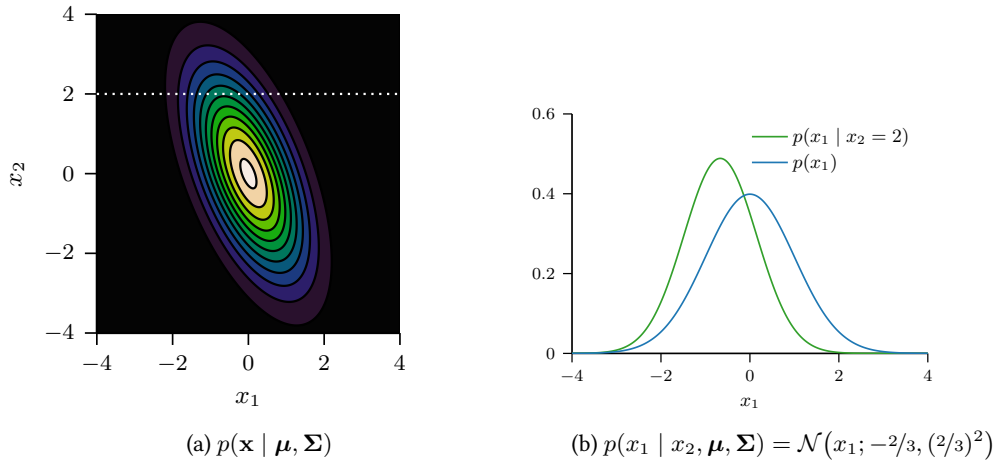


Figure 4: Conditioning example. (a) shows the joint density over  $\mathbf{x} = [x_1, x_2]^\top$ , along with the observation value  $x_2 = 2$ ; this is the same density as in Figure 2(c). (b) shows the conditional density of  $x_1$  given  $x_2 = 2$ .

### Affine transformations

Consider a  $d$ -dimensional vector  $\mathbf{x}$  with a multivariate Gaussian distribution:

*if* 
$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

Suppose we wish to reason about an affine transformation of  $\mathbf{x}$  into  $\mathbb{R}^D$ ,  $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ , where  $\mathbf{A} \in \mathbb{R}^{D \times d}$  and  $\mathbf{b} \in \mathbb{R}^D$ . Then  $\mathbf{y}$  has a  $D$ -dimensional Gaussian distribution:

*then* 
$$p(\mathbf{y} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{A}, \mathbf{b}) = \mathcal{N}(\mathbf{y}, \mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\top).$$

Figure 5 illustrates an affine transformation of the vector  $\mathbf{x}$  with the joint distribution shown in Figure 2(c), for the values

$$\mathbf{A} = \begin{bmatrix} 1/5 & -3/5 \\ 1/2 & 3/10 \end{bmatrix}; \quad \mathbf{b} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

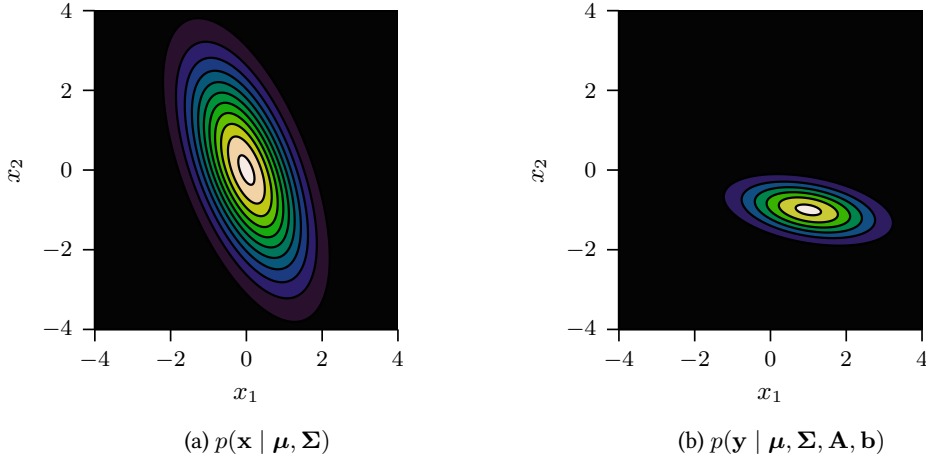


Figure 5: Affine transformation example. (a) shows the joint density over  $\mathbf{x} = [x_1, x_2]^\top$ ; this is the same density as in Figure 2(c). (b) shows the density of  $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ . The values of  $\mathbf{A}$  and  $\mathbf{b}$  are given in the text. The density of the transformed vector is another Gaussian.

The density has been rotated and translated, but remains a Gaussian.

### Pointwise multiplication

Another remarkable fact about multivariate Gaussian density functions is that pointwise multiplication gives another (unnormalized) Gaussian PDF:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \mathcal{N}(\mathbf{x}; \boldsymbol{\nu}, \mathbf{P}) = \frac{1}{Z} \mathcal{N}(\mathbf{x}; \boldsymbol{\omega}, \mathbf{T}),$$

where

$$\begin{aligned} \mathbf{T} &= (\boldsymbol{\Sigma}^{-1} + \mathbf{P}^{-1})^{-1} \\ \boldsymbol{\omega} &= \mathbf{T}(\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathbf{P}^{-1}\boldsymbol{\nu}) \\ Z^{-1} &= \mathcal{N}(\boldsymbol{\mu}; \boldsymbol{\nu}, \boldsymbol{\Sigma} + \mathbf{P}) = \mathcal{N}(\boldsymbol{\nu}; \boldsymbol{\mu}, \boldsymbol{\Sigma} + \mathbf{P}). \end{aligned}$$

## Convolutions

Gaussian probability density functions are closed under convolutions. Let  $\mathbf{x}$  and  $\mathbf{y}$  be  $d$ -dimensional vectors, with distributions

$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}); \quad p(\mathbf{y} \mid \boldsymbol{\nu}, \mathbf{P}) = \mathcal{N}(\mathbf{y}; \boldsymbol{\nu}, \mathbf{P}).$$

Then the convolution of their density functions is another Gaussian PDF:

$$f(\mathbf{y}) = \int \mathcal{N}(\mathbf{y} - \mathbf{x}; \boldsymbol{\nu}, \mathbf{P}) \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{x} = \mathcal{N}(\mathbf{y}; \boldsymbol{\mu} + \boldsymbol{\nu}, \boldsymbol{\Sigma} + \mathbf{P}),$$

where the mean and covariances add in the result.

If we assume that  $\mathbf{x}$  and  $\mathbf{y}$  are independent, then the distribution of their sum  $\mathbf{z} = \mathbf{x} + \mathbf{y}$  will also have a multivariate Gaussian distribution, whose density is precisely the convolution of the individual densities:

$$p(\mathbf{z} \mid \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\Sigma}, \mathbf{P}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu} + \boldsymbol{\nu}, \boldsymbol{\Sigma} + \mathbf{P}).$$

These results will often come in handy.

## Selecting parameters

The  $d$ -dimensional multivariate Gaussian distribution is specified by the parameters  $\boldsymbol{\mu}$  and  $\boldsymbol{\Sigma}$ . Without any further restrictions, specifying  $\boldsymbol{\mu}$  requires  $d$  parameters and specifying  $\boldsymbol{\Sigma}$  requires a further  $\binom{d}{2} = \frac{d(d-1)}{2}$  parameters. The number of parameters therefore grows quadratically in the dimension, which can sometimes cause difficulty. For this reason, we sometimes restrict the covariance matrix  $\boldsymbol{\Sigma}$  in some way to reduce the number of parameters.

Common choices are to set  $\boldsymbol{\Sigma} = \text{diag } \boldsymbol{\tau}$ , where  $\boldsymbol{\tau}$  is a vector of marginal variances, and  $\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}$ , a constant diagonal matrix. Both of these options assume independence between the variables in  $\mathbf{x}$ . The former case is more flexible, allowing a different scale parameter for each entry, whereas the latter assumes an equal marginal variance of  $\sigma^2$  for each variable. Geometrically, the densities are axis-aligned, as in Figure 2(a), and in the latter case, the isoprobability contours are spherical (also as in Figure 2(a)).

## Linear regression

We are ready to consider our first machine-learning problem: **linear regression**. Suppose that we are interested in the values of a function  $y(\mathbf{x}): \mathbb{R}^d \rightarrow \mathbb{R}$ , where  $\mathbf{x}$  is a  $d$ -dimensional vector-valued input. We assume that we have made some observations of this mapping,  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ , to serve as training data. Given these examples, the goal of regression is to be able to predict the value of  $y$  at a new input location  $\mathbf{x}^*$ . In other words, we want to learn from  $\mathcal{D}$  a (hopefully accurate!) prediction function (also called a **regression function**)  $\hat{y}(\mathbf{x}^*)$ .

Without any further restrictions on  $\hat{y}$ , the problem is not well-posed. In general, I could choose any function  $\hat{y}$ . How do I choose a good function? In particular, how do I effectively use the training observations  $\mathcal{D}$  to guide this choice?

The space of all possible regression functions  $\hat{y}$  is enormous; to restrict this space and make learning more tractable, linear regression assumes the relationship between  $\mathbf{x}$  and  $y$  is “mostly” linear:

$$y(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + \varepsilon(\mathbf{x}), \quad (1)$$

where  $\mathbf{w} \in \mathbb{R}^d$  is a vector of parameters, and  $\varepsilon(\mathbf{x})$  represents a departure from the underlying linear function  $\mathbf{x}^\top \mathbf{w}$  at  $\mathbf{x}$ . The value of  $\varepsilon(\mathbf{x})$  is also called the **residual**. If the underlying linear trend closely matches the data, we can expect small residuals.

Note that this model does not explicitly contain an intercept term. Instead, we normally add a constant feature to the input vector  $\mathbf{x}$  as in  $\mathbf{x}' = [1, \mathbf{x}]^\top$ . In this case, the value  $w_0$  may be seen as an intercept term, allowing the hyperplane to avoid passing through the origin at  $\mathbf{x} = 0$ . To avoid messy notation, we will not explicitly write the prime on the inputs and rather assume this expansion has already been done.

In general, we can imagine applying any function to  $\mathbf{x}$  to serve as the inputs to our linear regression model. For example, to accomplish polynomial regression of a given degree  $k$ , we might take  $\mathbf{x}' = [1, \mathbf{x}, \mathbf{x}^2, \dots, \mathbf{x}^k]^\top$ , where exponentiation is pointwise. A transformation of this type may be summarized by a map  $\mathbf{x} \mapsto \phi(\mathbf{x})$ ; this is known as a **basis expansion**. With a nonlinear basis expansion, we can learn nonlinear regression functions. We assume in the following that such an expansion, if desired, has already been applied to the inputs.

Let us collect the  $N$  training examples into a matrix of training inputs  $\mathbf{X} \in \mathbb{R}^{n \times d}$  and a vector of associated outputs  $\mathbf{y} \in \mathbb{R}^n$ . Then our assumed linear relationship in (1), when applied to the training data, may be written

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \varepsilon,$$

where we have also collected the residuals into the vector  $\varepsilon$ .

There are many classical approaches for estimating the parameter vector  $\mathbf{w}$ ; below, we describe two.

### Ordinary least squares

As mentioned previously, a “good” linear mapping  $\mathbf{x}^\top \mathbf{w}$  should result in small residuals. Our goal is to minimize the magnitude of the residuals at every possible test input  $\mathbf{x}^*$ . Of course, this is impossible, so we settle for the next best thing: we minimize the magnitude of the residuals on our training data.

The classical approach is to estimate  $\mathbf{w}$  by minimizing the sum of the squared residuals on the training data:

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2. \quad (2)$$

This approach is called **ordinary least squares**. The underlying assumption is that the magnitude of the residuals are all independent from each other, so that we can simply sum up the squared residuals and minimize. If the residuals were instead correlated, we would have to consider the interaction between pairs of residuals. This is also possible and gives rise to **generalized least squares**.

It turns out that the minimization problem in (2) has an exact, closed-form solution:

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

To predict the value of  $y$  at a new input  $\mathbf{x}^*$ , we plug our estimate of the parameter vector into (1):

$$\hat{y}(\mathbf{x}^*) = \mathbf{x}^{*T} \hat{\mathbf{w}} = \mathbf{x}^{*T} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Ordinary least squares is by far the most commonly applied linear regression procedure.

### Ridge regression

Occasionally, the ordinary least squares approach can lead to problems. In particular, when the training data lie extremely close to a particular hyperplane (or when the number of observations is less than the dimension, and we may find a hyperplane intersecting the data exactly), the magnitudes of the estimated parameters  $\hat{\mathbf{w}}$  can become ill-behaved and extremely large. This is an example of **overfitting**.

*A priori*, we might reasonably expect or desire the values of  $\mathbf{w}$  to be relatively small. For this reason, so-called **penalization** or **regularization** methods modify the objective in (2) to avoid such pathologies. The most common approach is to add a term to the objective encouraging small entries of  $\mathbf{w}$ , for example

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 + \lambda \|\mathbf{w}\|_2^2, \quad (3)$$

where we have added the squared 2-norm of  $\mathbf{w}$  to the function to be minimized, thereby penalizing large-magnitude entries of  $\mathbf{w}$ . The scalar  $\lambda \geq 0$  serves to trade off the contributions of the residual term and the penalty term, and is a parameter of the model that we may tune as desired. As  $\lambda \rightarrow 0$ , we recover the ordinary least squares objective.

The minimization problem in (3) may also be solved exactly, giving the slightly modified estimator

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Again, as  $\lambda \rightarrow 0$  (corresponding to very small penalties on the magnitude of  $\mathbf{w}$  entries), we see that the resulting estimator converges to the ordinary least squares estimator. As  $\lambda \rightarrow \infty$  (corresponding to very large penalties on the magnitude of  $\mathbf{w}$  entries), the estimator converges to  $\mathbf{0}$ .

This approach is known as *ridge regression*, named for the additional term  $\lambda \mathbf{I}$  in the estimator, which when viewed from “above” can whimsically be described as resembling a mountain ridge.



## Bayesian linear regression

Here we consider a Bayesian approach to linear regression. Consider again the assumed underlying linear relationship (1):

$$y(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + \varepsilon(\mathbf{x}).$$

Here the vector  $\mathbf{w}$  serves as the unknown parameter of the model, which we will reason about using the Bayesian method. For convenience, we will select a multivariate Gaussian prior distribution on  $\mathbf{w}$ :

*prior*

$$p(\mathbf{w} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

As before, we might expect *a priori* that the entries of  $\mathbf{w}$  are small. Furthermore, in the absence of any evidence otherwise, we might reason that all potential directions of  $\mathbf{w}$  are equally likely. These two assumptions can be codified by choosing a multivariate Gaussian distribution for  $\mathbf{w}$  with zero mean and isotropic diagonal covariance  $s^2 \mathbf{I}$ :

$$p(\mathbf{w} \mid s^2) = \mathcal{N}(\mathbf{w}; \mathbf{0}, s^2 \mathbf{I}). \quad (4)$$

Given our observed data  $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ , we wish to form the posterior distribution  $p(\mathbf{w} \mid \mathcal{D})$ . We will do so by deriving the joint distribution between the weight vector  $\mathbf{w}$  and the observed vector of values  $\mathbf{y}$ , given the inputs  $\mathbf{X}$ . We write again the assumed linear relationship in our training data:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}.$$

Given  $\mathbf{w}$  and  $\mathbf{X}$ , the only uncertainty in the above is the additive residuals  $\boldsymbol{\varepsilon}$ . We assume that the residuals are independent, unbiased, and tend to be “small”. We may accomplish this by modeling each entry of  $\boldsymbol{\varepsilon}$  as arising from zero-mean, independent, identically distributed Gaussian noise with variance  $\sigma^2$ :

*likelihood*

$$p(\boldsymbol{\varepsilon} \mid \sigma^2) = \mathcal{N}(\boldsymbol{\varepsilon}; \mathbf{0}, \sigma^2 \mathbf{I}).$$

Define  $\mathbf{f} = \mathbf{X}\mathbf{w}$ , and note that  $\mathbf{f}$  is a linear transformation of the multivariate-Gaussian distributed  $\mathbf{w}$ ; therefore, we may easily derive its distribution:

$$p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{f}; \mathbf{X}\boldsymbol{\mu}, \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top).$$

Now the observation vector  $\mathbf{y}$  is a sum of two independent multivariate Gaussian distributed vectors  $\mathbf{f}$  and  $\boldsymbol{\varepsilon}$ . Therefore, we may derive its prior distribution as well:

$$p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{y}; \mathbf{X}\boldsymbol{\mu}, \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2 \mathbf{I}).$$

Finally, we compute the covariance between  $\mathbf{y}$  and  $\mathbf{w}$ :

$$\text{cov}[\mathbf{y}, \mathbf{w}] = \text{cov}[\mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}, \mathbf{w}] = \text{cov}[\mathbf{X}\mathbf{w}, \mathbf{w}] = \mathbf{X} \text{cov}[\mathbf{w}, \mathbf{w}] = \mathbf{X}\boldsymbol{\Sigma},$$

where we have used the linearity of covariance and the fact that the noise vector  $\boldsymbol{\varepsilon}$  is independent of  $\mathbf{w}$ .

Now the joint distribution of  $\mathbf{w}$  and  $\mathbf{y}$  can be expressed as a multivariate Gaussian:

$$p\left(\begin{bmatrix} \mathbf{w} \\ \mathbf{y} \end{bmatrix} \mid \mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{w} \\ \mathbf{y} \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \mathbf{X}\boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}\mathbf{X}^\top \\ \mathbf{X}\boldsymbol{\Sigma} & \mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2 \mathbf{I} \end{bmatrix}\right).$$

We apply the formula for conditioning multivariate Gaussians on subvectors to derive the posterior distribution of  $\mathbf{w}$  given the data  $\mathcal{D}$ :

posterior

$$p(\mathbf{w} \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}),$$

where

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\mu} + \boldsymbol{\Sigma}\mathbf{X}^\top (\mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\mu}); \\ \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\Sigma} - \boldsymbol{\Sigma}\mathbf{X}^\top (\mathbf{X}\boldsymbol{\Sigma}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}\mathbf{X}\boldsymbol{\Sigma}.\end{aligned}$$

### Making predictions

Suppose we wish to use our model to predict the outputs  $\mathbf{y}^*$  associated with a set of inputs  $\mathbf{X}^*$ . Writing again  $\mathbf{y}^* = \mathbf{X}^*\mathbf{w} + \varepsilon$ , we derive:

$$p(\mathbf{y}_* \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) = \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \mathbf{X}_*\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}\mathbf{X}_*^\top + \sigma^2\mathbf{I}).$$

Therefore, we have a full, joint probability distribution over the outputs  $\mathbf{y}_*$ . If we must make point estimates, we choose a loss function for our predictions and use Bayesian decision theory. For example, to minimize expected squared loss, we predict the posterior mean.

An equivalent way to derive this result is to use the sum rule as follows:

inference

$$\begin{aligned}p(\mathbf{y}_* \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) &= \int p(\mathbf{y}_* \mid \mathbf{w}, \sigma^2) p(\mathbf{w} \mid \mathcal{D}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \sigma^2) d\mathbf{w} \\ &= \int \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\mathbf{w}, \sigma^2\mathbf{I}) \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}) d\mathbf{w} \\ &= \mathcal{N}(\mathbf{y}_*; \mathbf{X}_*\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \mathbf{X}_*\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}\mathbf{X}_*^\top + \sigma^2\mathbf{I}),\end{aligned}$$

where we have used a slightly more general form of the convolution formula for multivariate Gaussians. In this case, we view the prediction process as considering the predictions we would make with every possible value of the parameters  $\mathbf{w}$  and averaging them according to their plausibility.

### Connection to ridge regression

For the simple prior (4), the posterior mean can be rewritten as:

$$\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}} = s^2\mathbf{X}^\top (s^2\mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I})^{-1}\mathbf{y} = \left( \mathbf{X}^\top\mathbf{X} + \frac{\sigma^2}{s^2}\mathbf{I} \right)^{-1} \mathbf{X}^\top\mathbf{y},$$

where we have factored out the  $s^2$  in the inverse and have applied the matrix identity

$$(\mathbf{A}\mathbf{B} + c\mathbf{I})^{-1}\mathbf{A} = \mathbf{A}(\mathbf{B}\mathbf{A} + c\mathbf{I})^{-1}.$$

This is exactly equal to the ridge regression solution (3) with  $\lambda = \frac{\sigma^2}{s^2}$ ! Why is this?

Consider finding the *maximum a posteriori* estimator of  $\mathbf{w}$  using the simple prior (4). Bayes' theorem tells us that the posterior distribution  $p(\mathbf{w} \mid \mathcal{D})$  is proportional to the likelihood times the prior:

$$p(\mathbf{w} \mid \mathbf{X}, \mathbf{y}, \sigma^2, s^2) \propto p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}, \sigma^2) p(\mathbf{w} \mid s^2).$$

The distribution of  $\mathbf{y}$  given  $\mathbf{w}$  is simple:

$$p(\mathbf{y} \mid \mathbf{X}, \mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{y}; \mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}) = \prod_{i=1}^n \mathcal{N}(y_i; \mathbf{x}_i^\top\mathbf{w}, \sigma^2).$$

Rather than maximizing the posterior directly, we may instead maximize its logarithm:

$$\begin{aligned}\hat{\mathbf{w}}_{\text{MAP}} &= \arg \max_{\mathbf{w}} -\frac{1}{2\sigma^2} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 - \frac{1}{2s^2} \sum_{i=1}^d w_i^2 \\ &= \arg \min_{\mathbf{w}} \sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2 + \frac{\sigma^2}{s^2} \|\mathbf{w}\|_2^2,\end{aligned}$$

where we multiplied by  $-2\sigma^2$  to derive the second line. This is exactly the ridge regression objective. Therefore ridge regression with  $\ell^2$  regularization can be seen from the Bayesian perspective as placing a Gaussian prior on  $\mathbf{w}$  and finding the MAP estimator. In general, many regularization methods can be interpreted as implicitly choosing a particular likelihood for the data and prior for the parameters of a model and maximizing the posterior density. For example,  $\ell^1$  regularization (as seen in LASSO) is equivalent to placing a Laplace distribution prior on the weight parameters, encouraging a sparse solution.

Continuing with this argument, we can interpret the squared loss function  $\sum_{i=1}^n (\mathbf{x}_i^\top \mathbf{w} - y_i)^2$  as coming from an iid Gaussian noise assumption. In general, loss functions in minimization problems such as (3) can often be interpreted as negative log likelihoods arising from an implicit independent noise assumption.

Until now we have always worked with likelihoods and prior distributions that were conjugate to each other, allowing the computation of the posterior distribution to be done in closed form. Unfortunately, there are numerous situations where this will not be the case, forcing us to approximate the posterior and related quantities (such as the model evidence or expectations under the posterior distribution). Logistic regression is a common linear method for binary classification, and attempting to use the Bayesian approach directly will be intractable.

## Logistic Regression

In linear regression, we supposed that we were interested in the values of a real-valued function  $y(\mathbf{x}): \mathbb{R}^d \rightarrow \mathbb{R}$ , where  $\mathbf{x}$  is a  $d$ -dimensional vector-valued input. Here, we will consider a similar setup, but with a twist: we restrict the output of the function  $y$  to the discrete space  $y \in \{0, 1\}$ . In machine learning, problems of this form fall under the category of **binary classification**: given an input  $\mathbf{x}$ , we wish to **classify** it into one of two categories, in this case denoted arbitrarily by 0 and 1.

We again assume that we have made some observations of this mapping,  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ , to serve as training data. Given these examples, the goal of binary classification is to be able to predict the label at a new input location  $\mathbf{x}^*$ .

As in linear regression, the problem is not yet well-posed without some restrictions on  $y$ . In linear regression, we assumed that the relationship between  $\mathbf{x}$  and  $y$  was “mostly” linear:

$$y(\mathbf{x}) = \mathbf{x}^\top \mathbf{w} + \varepsilon(\mathbf{x}),$$

where  $\mathbf{w} \in \mathbb{R}^d$  is a vector of parameters, and  $\varepsilon(\mathbf{x})$  is the residual. This assumption is not very desirable in the classification case, where the outputs are restricted to  $\{0, 1\}$  (note, for example, that  $\mathbf{x}^\top \mathbf{w}$  is unbounded as the norm of  $\mathbf{x}$  increases, forcing the residuals to grow ever larger).

In linear classification methods, we instead assume that the class-conditional probability of belonging to the “1” class is given by a nonlinear transformation of an underlying linear function of  $\mathbf{x}$ :

$$\Pr(y = 1 \mid \mathbf{x}, \mathbf{w}) = \sigma(\mathbf{x}^\top \mathbf{w}),$$

where  $\sigma$  is a so-called “sigmoid” (or “s-shaped”) increasing function mapping the real line to valid probabilities in  $(0, 1)$ . The most-commonly used functions  $\sigma$  are the **logistic function**:

$$\sigma(a) = \frac{\exp(a)}{1 + \exp(a)},$$

or the standard normal cumulative distribution function:

$$\sigma(a) = \Phi(a) = \int_{-\infty}^a \mathcal{N}(x; 0, 1^2) dx.$$

These two choices are compared in Figure 1. The main qualitative difference is that the logistic function has slightly heavier tails than the normal CDF. Linear classification using the logistic function is called **logistic regression**; linear classification using the normal CDF is called **probit regression**. Logistic regression is more commonly encountered in practice. Notice that the linear assumption above combined with the logistic function sigmoid implies that the **log odds** are a linear function of the input  $\mathbf{x}$ :

$$\log \frac{\Pr(y = 1 \mid \mathbf{x}, \mathbf{w})}{\Pr(y = 0 \mid \mathbf{x}, \mathbf{w})} = \log \frac{\frac{\exp(\mathbf{x}^\top \mathbf{w})}{1 + \exp(\mathbf{x}^\top \mathbf{w})}}{1 - \frac{\exp(\mathbf{x}^\top \mathbf{w})}{1 + \exp(\mathbf{x}^\top \mathbf{w})}} = \log \frac{\exp(\mathbf{x}^\top \mathbf{w})}{1} = \mathbf{x}^\top \mathbf{w}.$$

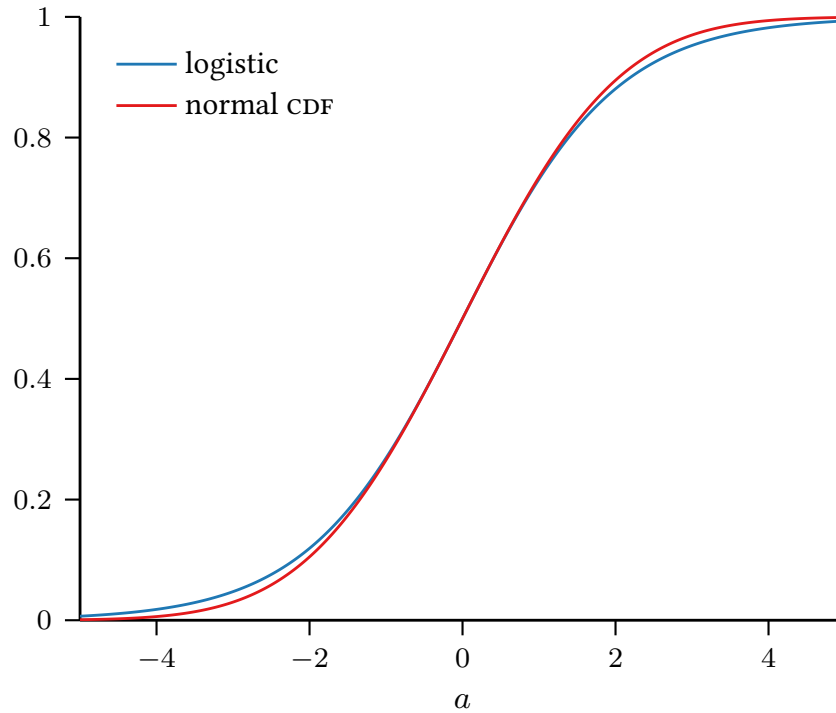


Figure 1: A comparison of the two sigmoid functions described in the text. The normal CDF curve in this example uses the transformation  $\Phi(\sqrt{\frac{\pi}{8}}a)$ , which ensures the slopes of the two curves are equal at the origin.

With the choice of the sigmoid function, and an assumption that our training labels  $\mathbf{y}$  are generated independently given  $\mathbf{w}$ , we have defined our likelihood  $\Pr(\mathbf{y} \mid \mathbf{X}, \mathbf{w})$ :

$$\Pr(\mathbf{y} \mid \mathbf{X}, \mathbf{w}) = \prod_{i=1}^n \sigma(\mathbf{x}_i^\top \mathbf{w})^{y_i} (1 - \sigma(\mathbf{x}_i^\top \mathbf{w}))^{1-y_i}. \quad (1)$$

To verify this equation, notice that each  $y_i$  will either be 0 or 1, so exactly one of  $y_i$  or  $1 - y_i$  will be nonzero, which picks out the correct contribution to the likelihood.

The traditional approach to logistic regression is to maximize the likelihood of the training data as a function of the parameters  $\mathbf{w}$ :

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w}} \Pr(\mathbf{y} \mid \mathbf{X}, \mathbf{w});$$

$\hat{\mathbf{w}}$  is therefore a maximum-likelihood estimator (MLE). Unlike in linear regression, where there was a closed-form expression for the maximum-likelihood estimator, there is no such solution for logistic regression. Things aren't too bad though, because it turns out that for logistic regression the negative log-likelihood is convex and positive definite, which means there is a unique global minimum (and therefore a unique MLE). There are numerous off-the-shelf methods available for finding  $\hat{\mathbf{w}}$ : gradient descent, Newton's method, etc.

## Bayesian logistic regression

A Bayesian approach to logistic regression requires us to select a prior distribution for the parameters  $\mathbf{w}$  and derive the posterior distribution  $p(\mathbf{w} \mid \mathcal{D})$ . For the former, we will consider a multivariate Gaussian prior, identical to the one we used for linear regression:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

Now we apply Bayes' theorem to write down the desired posterior:

$$p(\mathbf{w} \mid \mathcal{D}) = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})}{\int p(\mathbf{y} \mid \mathbf{X}, \mathbf{w})p(\mathbf{w}) d\mathbf{w}} = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y} \mid \mathbf{X})}.$$

Unfortunately, the product of the Gaussian prior on  $\mathbf{w}$  and the likelihood (1), for either choice of sigmoid, does not result in a posterior distribution in a nice parametric family that we know. Likewise, the integral in the normalization constant (the model evidence)  $p(\mathbf{y} \mid \mathbf{X})$  is **intractable** as well.

How can we proceed? There are two main approaches to continuing Bayesian inference in such a situation. The first is to use a deterministic method to find an approximation to the posterior (that will typically live inside some chosen parametric family). The second is to forgo a closed-form expression for the posterior and instead derive an algorithm to draw *samples* from the posterior distribution, which we may use to, for example, make Monte Carlo estimates to expectations. Here we will consider the **Laplace approximation**, an example of the first type of approach.

## Laplace Approximation to the Posterior

Suppose we have an arbitrary parameter prior  $p(\boldsymbol{\theta})$  and an arbitrary likelihood  $p(\mathcal{D} \mid \boldsymbol{\theta})$ , and wish to approximate the posterior

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{1}{Z} p(\mathcal{D} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}),$$

where the normalization constant  $Z$  is the unknown evidence. We define the following function:

$$\Psi(\boldsymbol{\theta}) = \log p(\mathcal{D} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}).$$

$\Psi$  is therefore the logarithm of the *unnormalized* posterior distribution. The Laplace approximation is based on a Taylor expansion to  $\Psi$  around its maximum. First, we find the maximum of  $\Psi$ :

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}} \Psi(\boldsymbol{\theta}).$$

Notice that the point  $\hat{\boldsymbol{\theta}}$  is a **maximum a posteriori** (MAP) approximation to the parameters. Finding  $\hat{\boldsymbol{\theta}}$  can be done in a variety of ways, but in practice it is usually fairly easy to find the gradient and Hessian of  $\Psi$  with respect to  $\boldsymbol{\theta}$  and use some off-the-shelf optimization routine. This is an example of the saying "optimization is easier than integration."

Once we have found  $\hat{\boldsymbol{\theta}}$ , we make a second-order Taylor expansion to  $\Psi$  around this point:

$$\Psi(\boldsymbol{\theta}) \approx \Psi(\hat{\boldsymbol{\theta}}) - \frac{1}{2} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^\top \mathbf{H} (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}),$$

where  $\mathbf{H}$  is the Hessian of the negative log posterior evaluated at  $\hat{\boldsymbol{\theta}}$ :

$$\mathbf{H} = -\nabla \nabla \Psi(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}.$$

Notice that the first-order term in the Taylor expansion vanishes because we expand around a maximum, where the gradient is zero. Exponentiating, we may derive an approximation to the (unnormalized) posterior distribution:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) \approx \exp(\Psi(\hat{\boldsymbol{\theta}})) \exp\left(-\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^\top \mathbf{H}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\right), \quad (2)$$

which we recognize as being proportional to a Gaussian distribution! The Laplace approximation therefore results in a normal approximation to the posterior distribution:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) \approx \mathcal{N}(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}, \mathbf{H}^{-1}).$$

The approximation is a Gaussian centered on the mode of the posterior,  $\hat{\boldsymbol{\theta}}$ , with covariance compelling the log of the approximation to match the curvature of the true log posterior at that point.

We note that the Laplace approximation also gives an approximation to the normalizing constant  $Z$ . In this case, it's simply a question of which normalizing constant we had to use to get (2) to normalize. A fairly straightforward calculation gives

$$Z = \int \exp(\Psi(\boldsymbol{\theta})) \, d\boldsymbol{\theta} \approx \int \exp(\Psi(\hat{\boldsymbol{\theta}})) \exp\left(-\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^\top \mathbf{H}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\right) \, d\boldsymbol{\theta} = \exp(\Psi(\hat{\boldsymbol{\theta}})) \sqrt{\frac{(2\pi)^d}{\det \mathbf{H}}},$$

where  $d$  is the dimension of  $\boldsymbol{\theta}$ .

## Making Predictions

Suppose we have obtained a Gaussian approximation to the posterior distribution  $p(\mathbf{w} \mid \mathcal{D})$ ; for example, applying the Laplace approximation above gives  $p(\mathbf{w} \mid \mathcal{D}) \approx \mathcal{N}(\mathbf{w}; \hat{\mathbf{w}}, \mathbf{H}^{-1})$ , where  $\hat{\mathbf{w}}$  is the MAP approximation to the parameters and  $\mathbf{H}$  is the Hessian of the negative log posterior evaluated at  $\hat{\mathbf{w}}$ .

Suppose now that we are given a test input  $\mathbf{x}^*$  and wish to predict the binary label  $y^*$ . In the Bayesian approach, we marginalize the unknown parameters  $\mathbf{w}$  to find the **predictive distribution**:

$$\Pr(y^* = 1 \mid \mathbf{x}^*, \mathcal{D}) = \int \Pr(y^* = 1 \mid \mathbf{x}^*, \mathbf{w}) p(\mathbf{w} \mid \mathcal{D}) \, d\mathbf{w} = \int \sigma(\mathbf{x}^{*T} \mathbf{w}) p(\mathbf{w} \mid \mathcal{D}) \, d\mathbf{w}.$$

Unfortunately, even with our Gaussian approximation to  $p(\mathbf{w} \mid \mathcal{D})$ , this integral cannot be evaluated if we use the logistic function in the role of the sigmoid  $\sigma$ . We can, however, **compute the integral** when using the **normal CDF for  $\sigma$** :

$$\Pr(y^* = 1 \mid \mathbf{x}^*, \mathcal{D}) = \int \Phi(\mathbf{x}^{*T} \mathbf{w}) \mathcal{N}(\mathbf{w}; \hat{\mathbf{w}}, \mathbf{H}^{-1}) \, d\mathbf{w}.$$

This looks like an annoying  $d$ -dimensional integral, but notice that the vector  $\mathbf{w}$  only appears in the expectation via the scalar product  $\mathbf{x}^{*T} \mathbf{w}$ . To proceed, we define the scalar value  $a = \mathbf{x}^{*T} \mathbf{w}$  and rewrite this as

$$\Pr(y^* = 1 \mid \mathbf{x}^*, \mathcal{D}) = \int_{-\infty}^{\infty} \Phi(a) p(a \mid \mathcal{D}) \, da.$$

Notice that  $a$  is a linear transformation of the Gaussian-distributed  $\mathbf{w}$ ; therefore,  $a$  has a Gaussian distribution:

$$p(a \mid \mathcal{D}) = \mathcal{N}(a; \mu_{a|\mathcal{D}}, \sigma_{a|\mathcal{D}}^2),$$

where

$$\mu_{a|\mathcal{D}} = \mathbf{x}^{*T} \hat{\mathbf{w}}; \quad \sigma_{a|\mathcal{D}}^2 = \mathbf{x}^{*T} \mathbf{H}^{-1} \mathbf{x}^*.$$

Now we may finally compute the integral:

$$\Pr(y^* = 1 \mid \mathbf{x}^*, \mathcal{D}) = \int_{-\infty}^{\infty} \Phi(a) \mathcal{N}(a; \mu_{a|\mathcal{D}}, \sigma_{a|\mathcal{D}}^2) da = \Phi\left(\frac{\mu_{a|\mathcal{D}}}{\sqrt{1 + \sigma_{a|\mathcal{D}}^2}}\right).$$

Notice that  $\Phi(\mu_{a|\mathcal{D}})$  would be the estimate we would make using the MAP  $\hat{\mathbf{w}}$  as a plug-in estimator. The  $\sqrt{1 + \sigma_{a|\mathcal{D}}^2}$  term has the effect of making our prediction less confident (that is, closer to  $1/2$ ) according to our uncertainty in the value of  $a = \mathbf{x}^{*T} \mathbf{w}$ . This procedure is sometimes called **moderation**, because we force our predictions to be more moderate than we would have using a plug-in point estimate of  $\mathbf{w}$ .

We also note that if we only want to make point predictions of  $y^*$  using the 0-1 loss function, we only need to know which class is more probable (this was a general result from our discussion of Bayesian decision theory). In this case, the moderation has no effect on our ultimate predictions (you can check that we never change which side of  $1/2$  the final probability is), and we may instead simply find  $\hat{\mathbf{w}}$ . This is similar to the result we had in linear regression, where we could simply find the MAP estimator for  $\mathbf{w}$  if we only ultimately cared about point predictions under a squared loss.

With loss functions different from the 0-1 loss, however, the uncertainty in  $\mathbf{w}$  can indeed be important.



## The Kernel Trick

Consider the assumption of linear regression with an explicit feature transformation  $\phi: \mathbf{x} \mapsto \phi(\mathbf{x})$ :

$$y(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w} + \varepsilon(\mathbf{x}).$$

Given training data  $\mathcal{D} = \{(\mathbf{x}, y)\}_{i=1}^n = (\mathbf{X}, \mathbf{y})$ , we first apply  $\phi$  to each data point:

$$\Phi = \phi(\mathbf{X}) = \begin{bmatrix} \phi(\mathbf{x}_1)^T \\ \vdots \\ \phi(\mathbf{x}_n)^T \end{bmatrix}.$$

Next, we proceed as we did before by modeling the residuals  $\varepsilon(\mathbf{x})$  as zero-mean independent, identically distributed Gaussians with variance  $\sigma^2$ :

$$p(\varepsilon) = \mathcal{N}(\varepsilon; \mathbf{0}, \sigma^2 \mathbf{I}),$$

giving rise to the following likelihood:

$$p(\mathbf{y} \mid \Phi, \mathbf{w}, \sigma^2) = \mathcal{N}(\mathbf{y}; \Phi \mathbf{w}, \sigma^2 \mathbf{I}).$$

In Bayesian linear regression, we further chose a multivariate Gaussian prior for  $\mathbf{w}$ :

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

For simplicity, in the derivations below, we will assume the prior mean for  $\mathbf{w}$  is  $\boldsymbol{\mu} = \mathbf{0}$ .

Given these assumptions, we can derive the posterior distribution of  $\mathbf{w}$  given the (transformed) data  $\mathcal{D}$ :

$$p(\mathbf{w} \mid \mathcal{D}, \sigma^2) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}),$$

where

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\Sigma} \Phi^\top (\Phi \boldsymbol{\Sigma} \Phi^\top + \sigma^2 \mathbf{I})^{-1} \mathbf{y}; \\ \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}} &= \boldsymbol{\Sigma} - \boldsymbol{\Sigma} \Phi^\top (\Phi \boldsymbol{\Sigma} \Phi^\top + \sigma^2 \mathbf{I})^{-1} \Phi \boldsymbol{\Sigma}. \end{aligned}$$

If we wish to use our model to predict the outputs  $\mathbf{y}^*$  associated with a set of inputs  $\mathbf{X}^*$ , we previously derived:

$$p(\mathbf{y}^* \mid \Phi^* = \phi(\mathbf{X}^*), \mathcal{D}, \sigma^2) = \mathcal{N}(\mathbf{y}^*; \Phi^* \boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}, \Phi^* \boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}} \Phi^{*T} + \sigma^2 \mathbf{I}).$$

Examining the forms of these expressions (after plugging in the posteriors  $\boldsymbol{\mu}_{\mathbf{w}|\mathcal{D}}$  and  $\boldsymbol{\Sigma}_{\mathbf{w}|\mathcal{D}}$ , we see that the feature expansion  $\phi$  always appears in one of the following expressions:

$$\Phi \boldsymbol{\Sigma} \Phi^\top, \quad \Phi^* \boldsymbol{\Sigma} \Phi^\top, \quad \Phi \boldsymbol{\Sigma} \Phi^{*T}, \quad \Phi^* \boldsymbol{\Sigma} \Phi^{*T}.$$

The entries of these matrices are always of the form  $\phi(\mathbf{x})^\top \boldsymbol{\Sigma} \phi(\mathbf{x}')$ , where  $\mathbf{x}$  and  $\mathbf{x}'$  are two arbitrary inputs. To simplify our expressions, we can define a function

$$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \boldsymbol{\Sigma} \phi(\mathbf{x}').$$

Because  $\Sigma$  is positive definite, it has a “matrix square root”,  $\Sigma^{1/2}$  with the property  $(\Sigma^{1/2})^2 = \Sigma$ .<sup>1</sup>

If we define the function  $\psi(\mathbf{x}) = \Sigma^{1/2} \phi(\mathbf{x})$ , we can see that  $K$  is simply an inner product:

$$K(\mathbf{x}, \mathbf{x}') = \psi(\mathbf{x})^\top \psi(\mathbf{x}').$$

Such a function  $K$  is called a **kernel** or **covariance function**. **Valid kernel functions** are guaranteed to always produce positive semi-definite Gram matrices: a **Gram matrix** is a square matrix of inner products between pairs of elements. Formally, given a set of vectors

$$X = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$$

the Gram matrix

$$\begin{aligned} K(X, X) &= \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \cdots & K(\mathbf{x}_1, \mathbf{x}_n) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \cdots & K(\mathbf{x}_2, \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(\mathbf{x}_n, \mathbf{x}_1) & K(\mathbf{x}_n, \mathbf{x}_2) & \cdots & K(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix} \\ &= \begin{bmatrix} \psi(\mathbf{x}_1)^\top \psi(\mathbf{x}_1) & \psi(\mathbf{x}_1)^\top \psi(\mathbf{x}_2) & \cdots & \psi(\mathbf{x}_1)^\top \psi(\mathbf{x}_n) \\ \psi(\mathbf{x}_2)^\top \psi(\mathbf{x}_1) & \psi(\mathbf{x}_2)^\top \psi(\mathbf{x}_2) & \cdots & \psi(\mathbf{x}_2)^\top \psi(\mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ \psi(\mathbf{x}_n)^\top \psi(\mathbf{x}_1) & \psi(\mathbf{x}_n)^\top \psi(\mathbf{x}_2) & \cdots & \psi(\mathbf{x}_n)^\top \psi(\mathbf{x}_n) \end{bmatrix} \end{aligned}$$

is **positive semi-definite**  $\forall$  sets  $X$ .

Sometimes it is possible to specify a covariance function  $K$  directly without ever computing the feature map explicitly. With such a function, we could perform efficient Bayesian linear regression even with a high-dimensional (or infinite dimensional) feature expansion  $\phi$  *implicitly*. This idea of computing inner products in a feature space directly is called the **kernel trick** and has been the basis of a large amount of work in the machine-learning community. Effectively, any algorithm that operates purely in terms of inner products between input vectors can be made nonlinear by replacing normal inner products with the evaluation of a kernel.

With this definition, we may rewrite the predictive distribution for  $\mathbf{y}^*$ :

$$p(\mathbf{y}_* | \mathbf{X}^*, \mathcal{D}, \sigma^2) = \mathcal{N}(\mathbf{y}^*; \mu_{\mathbf{y}^*|\mathcal{D}}, K_{\mathbf{y}^*|\mathcal{D}}),$$

where

$$\begin{aligned} \mu_{\mathbf{y}^*|\mathcal{D}} &= K(\mathbf{X}^*, \mathbf{X}) (K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I})^{-1} \mathbf{y}; \\ K_{\mathbf{y}^*|\mathcal{D}} &= K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X}) (K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I})^{-1} K(\mathbf{X}, \mathbf{X}^*) + \sigma^2 \mathbf{I}^*. \end{aligned}$$

<sup>1</sup>You can prove this via the singular value decomposition (svd):  $\Sigma = \mathbf{U} \mathbf{D} \mathbf{U}^\top$ , where  $\mathbf{U}$  is unitary and  $\mathbf{D}$  is diagonal with positive entries (because  $\Sigma$  is positive definite), then  $\Sigma^{1/2} = \mathbf{U} \mathbf{D}^{1/2} \mathbf{U}^\top$  as  $\mathbf{U}^\top \mathbf{U} = \mathbf{I}$ .

## Examples

Perhaps the most-commonly used kernel is the **squared exponential** covariance function:

$$K(\mathbf{x}, \mathbf{x}'; \lambda, \ell) = \lambda^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right),$$

where  $\lambda$  and  $\ell$  are hyperparameters that control the covariance function's behavior. The former is simply a multiplicative scaling constant (you can think of this as an implicit scalar multiplication in the implicit feature map  $\phi$ ). The latter takes the role of a **length scale**; vectors separated by more than a couple length scales will have a kernel value near zero.

An example of Bayesian linear regression using this kernel function is shown in Figure 1. We see that the use of this kernel function allowed us to achieve nice nonlinear regression without computing explicit basis expansions. In fact, you can show that the squared exponential kernel corresponds to an *infinite-dimensional* basis expansion, where we use a Gaussian basis function centered on every point. Such a feature expansion would be impossible to use if we attempted to use explicit feature computation.

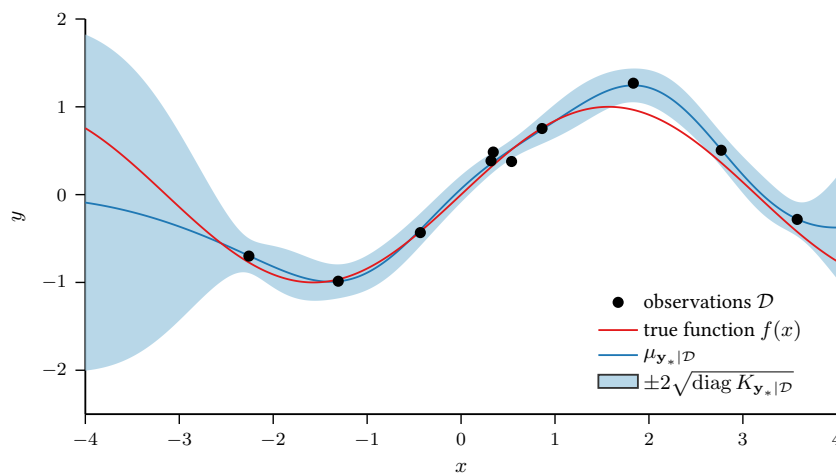


Figure 1: Example of Bayesian linear regression using the squared exponential covariance function. The true function is  $f = \sin(x)$ . The kernel parameters are  $\lambda = \ell = 1$ , and the noise variance was set to  $\sigma^2 = 0.1^2$ .

Sometimes thinking in terms of the kernel can help even when you have an explicit feature expansion on hand. As an example, imagine our inputs are binary vectors of length  $n$  (so each input  $\mathbf{x}$  is a subset, a member of the power set  $\mathcal{P}(n)$ ). One rather expensive feature expansion we could try would be to enumerate every member of  $\mathcal{P}(n)$  and define  $\phi(\mathbf{x})_i = \mathbf{s}_i \subset \mathbf{x}$ , where  $\mathbf{s}_i$  is the  $i^{\text{th}}$  element of the power set. So we represent our set  $\mathbf{x}$  by a feature vector of length  $2^n$  indicating every subset of  $\mathbf{x}$ . This is a very expensive feature expansion, requiring exponential space to store for each input. However, if we take  $\Sigma = \mathbf{I}$ , we can compute the dot product as:

$$K(\mathbf{x}, \mathbf{x}') = 2^{|\mathbf{x} \cap \mathbf{x}'|},$$

which only requires time and space linear in  $n$ !

## Bayesian model selection

In this course, we will learn about many kernel functions for probabilistic modeling. One major question that we will need to address is how to choose a good kernel for a given modeling task. There are some settings where prior knowledge can inform this decision but is there a more quantitative way to select which kernel function to use? Even more generally, how do I select what probabilistic model I should use to explain my data in the first place? These are questions of **model selection**, and naturally there is a Bayesian approach to it.

Before we continue our discussion of model selection, we will first define the word **model**: a model is a parametric family of probability distributions, each of which could explain some observed dataset. Another way to explain the concept of a model is that if we have chosen a likelihood  $p(\mathcal{D} | \theta)$  for our data, which depends on a parameter  $\theta$ , then the model is the set of all likelihoods (each one of which is a distribution over  $\mathcal{D}$ ) for every possible value of the parameter  $\theta$ .

As an example, consider the setting of coin flipping: flipping a coin  $n$  times with an unknown bias  $\theta$  and observing the number of heads  $x$ , the model is

$$\{p(x | n, \theta)\} = \{\text{Binomial}(x, n, \theta)\},$$

where there is a binomial distribution for every possible  $\theta \in (0, 1)$ . In the Bayesian method, we maintain a belief over which elements in the model we consider plausible by reasoning about  $p(\theta | \mathcal{D})$  via Bayes' theorem.

Suppose now that I have at my disposal a finite set of models  $\{\mathcal{M}_i\}_{i=1}^n$  that I may use to explain my observed data  $\mathcal{D}$ , and let us write  $\theta_i$  for the parameters of model  $\mathcal{M}_i$ . How do we know which model to prefer? We can work out the posterior probability over the models via Bayes' theorem:

$$\Pr(\mathcal{M}_i | \mathcal{D}) = \frac{p(\mathcal{D} | \mathcal{M}_i) \Pr(\mathcal{M}_i)}{\sum_j p(\mathcal{D} | \mathcal{M}_j) \Pr(\mathcal{M}_j)}.$$

Here  $\Pr(\mathcal{M}_i)$  is a prior distribution over models that we have selected; a common practice is to set this to a uniform distribution over the models. The value  $p(\mathcal{D} | \mathcal{M}_i)$  may also be written in a more familiar form:

$$p(\mathcal{D} | \mathcal{M}_i) = \int p(\mathcal{D} | \theta_i, \mathcal{M}_i) p(\theta_i | \mathcal{M}_i) d\theta_i.$$

This is exactly the denominator when applying Bayes' theorem to find the posterior  $p(\theta_i | \mathcal{D}, \mathcal{M}_i)$ !

$$p(\theta_i | \mathcal{D}, \mathcal{M}_i) = \frac{p(\mathcal{D} | \theta_i, \mathcal{M}_i) p(\theta_i | \mathcal{M}_i)}{\int p(\mathcal{D} | \theta_i, \mathcal{M}_i) p(\theta_i | \mathcal{M}_i) d\theta_i} = \frac{p(\mathcal{D} | \theta_i, \mathcal{M}_i) p(\theta_i | \mathcal{M}_i)}{p(\mathcal{D} | \mathcal{M}_i)},$$

where we have made the conditioning on  $\mathcal{M}_i$  explicit. In the context of model selection, the term  $p(\mathcal{D} | \mathcal{M}_i)$  is known as the **model evidence** or simply the **evidence**. One interpretation of the model evidence is the probability that your model could have generated the observed data, under the chosen prior belief over its parameters  $\theta_i$ .

If we have only two models for the observed data that we wish to compare,  $\mathcal{M}_1$  and  $\mathcal{M}_2$ , it is easiest to compute the **posterior odds** or the ratio of the models' probabilities given the data:

$$\frac{\Pr(\mathcal{M}_1 | \mathcal{D})}{\Pr(\mathcal{M}_2 | \mathcal{D})} = \frac{\Pr(\mathcal{M}_1) p(\mathcal{D} | \mathcal{M}_1)}{\Pr(\mathcal{M}_2) p(\mathcal{D} | \mathcal{M}_2)} = \frac{\Pr(\mathcal{M}_1) \int p(\mathcal{D} | \theta_1, \mathcal{M}_1) p(\theta_1 | \mathcal{M}_1) d\theta_1}{\Pr(\mathcal{M}_2) \int p(\mathcal{D} | \theta_2, \mathcal{M}_2) p(\theta_2 | \mathcal{M}_2) d\theta_2},$$

which is simply the prior odds multiplied by the ratio of the evidence for each model. The latter quantity is also called the **Bayes factor** in favor of  $\mathcal{M}_1$ . Publishing Bayes factors allows another practitioner to easily substitute their own model priors and derive their own conclusions about the models being considered.

### Example (from [Wikipedia's article on Bayes factor](#))

Suppose I am presented with a coin and want to compare two models for explaining its behavior. The first model,  $\mathcal{M}_1$ , assumes that the heads probability is fixed to  $1/2$  (this model does not have any parameters). The second model,  $\mathcal{M}_2$ , assumes that the heads probability is fixed to an unknown value  $\theta \in (0, 1)$ , with a uniform prior on  $\theta$ :  $p(\theta | \mathcal{M}_2) = 1$  (this is equivalent to a beta prior on  $\theta$  with  $\alpha = \beta = 1$ ). For simplicity, we choose a uniform model prior:  $\Pr(\mathcal{M}_1) = \Pr(\mathcal{M}_2) = 1/2$ .

Suppose we flip the coin  $n = 200$  times and observe  $x = 115$  heads. Which model should we prefer in light of this data? We compute the model evidence for each model. The model evidence for  $\mathcal{M}_1$  is quite straightforward, as it has no parameters:

$$\Pr(x | n, \mathcal{M}_1) = \text{Binomial}(n, x, 1/2) = \binom{200}{115} \frac{1}{2^{200}} \approx 0.005956.$$

The model evidence for  $\mathcal{M}_2$  requires integrating over the parameter  $\theta$ :

$$\begin{aligned} \Pr(x | n, \mathcal{M}_2) &= \int \Pr(x | n, \theta, \mathcal{M}_2) p(\theta | \mathcal{M}_2) d\theta \\ &= \int_0^1 \binom{200}{115} \theta^{115} (1 - \theta)^{200-115} d\theta = \frac{1}{201} \approx 0.004975. \end{aligned}$$

The Bayes factor in favor of  $\mathcal{M}_1$  is approximately 1.2, so the data give very weak evidence in favor of the simpler model  $\mathcal{M}_1$ .

An interesting aside here is that a frequentist hypothesis test would reject the null hypothesis  $\theta = \frac{1}{2}$  at the  $\alpha = 0.05$  level. The probability of generating at least 115 heads under model  $\mathcal{M}_1$  is approximately 0.02 and similarly, the probability of generating at least 115 tails is also 0.02, so a two-sided test would give a  $p$ -value of approximately 4%.

### Occam's razor

One spin on Bayesian decision theory is that it automatically gives a preference towards simpler models, in line with Occam's razor. One way to see this is to consider the model evidence  $p(\mathcal{D} | \mathcal{M})$  as a probability distribution over datasets  $\mathcal{D}$ . More complex models can explain more datasets, so the support of this distribution is wider in the sample space. But note that the distribution must normalize over the sample space as well, so we pay a price for generality. When moving from a simpler model to a more complex model, the probability of some datasets that are well explained by the simpler model must inevitably decrease to "give up" probability mass for the newly explained datasets in the widened support of the more-complex model. The model selection process then drives us to select the model that is "just complex enough" to explain the data at hand.

In the coin flipping example above, model  $\mathcal{M}_1$  can only explain datasets with empirical heads probability reasonably near  $1/2$ . An observation of 200 heads, for example, would have astronomically small probability under this model. The second model  $\mathcal{M}_2$  can explain *any* set of observations by selecting an appropriate  $\theta$ . The price for this generality, though, is that datasets with a roughly equal number of heads and tails have a smaller prior probability under the model than before.

## Bayesian Model Averaging

Note that a “fully Bayesian” approach to models would eschew model selection entirely. Instead, when making predictions, we should theoretically use the sum rule to marginalize the unknown model, giving rise to the model-marginal predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}) = \sum_i p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i) \Pr(\mathcal{M}_i \mid \mathcal{D}).$$

Such an approach is called **Bayesian model averaging**. Although this is sometimes seen, model selection is much more common because the computational overhead of using a single model is much lower than having to continually retrain multiple models. Also, the model-marginal predictive distribution tends to have annoying analytic properties, which can make it difficult to work with.

## Gaussian Processes

We previously considered the **kernel trick**, which allowed us to define the **kernel function**

$$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \Sigma \phi(\mathbf{x}').$$

Then, when doing Bayesian linear regression with prior

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \Sigma),$$

and data  $\mathcal{D}$ , we derived the predictive distribution

$$p(\mathbf{y}^* | \mathbf{X}^*, \mathcal{D}, \sigma^2) = \mathcal{N}(\mathbf{y}^*; \boldsymbol{\mu}_{\mathbf{y}^*|\mathcal{D}}, \mathbf{K}_{\mathbf{y}^*|\mathcal{D}}),$$

where

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{y}^*|\mathcal{D}} &= \mathbf{K}^{*T} (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} \\ \mathbf{K}_{\mathbf{y}^*|\mathcal{D}} &= \mathbf{K}^{**} - \mathbf{K}^{*T} (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{K}^* + \sigma^2 \mathbf{I}^*, \end{aligned}$$

and we have defined

$$\mathbf{K} = K(\mathbf{X}, \mathbf{X}) \quad \mathbf{K}^* = K(\mathbf{X}, \mathbf{X}^*) \quad \mathbf{K}^{**} = K(\mathbf{X}^*, \mathbf{X}^*).$$

Examining the form of this predictive distribution, we barely need to perform inference over the weight vector  $\mathbf{w}$  at all; its covariance simply appears in the definition of  $K$ . If we treat  $K$  as a black-box, maybe we can skip the inference over  $\mathbf{w}$  entirely and simply work with  $K$  directly. This line of thinking motivates **Gaussian processes**.

Instead of reasoning about  $\mathbf{w}$  and  $\phi$ , the Gaussian process approach to regression directly reasons about the *function*  $f$ : we construct an explicit prior distribution for  $f$ , that we reason about via the Bayesian method. This seems like it might be difficult, because  $f$  might in general be infinite-dimensional. A Gaussian process provides a natural extension of the familiar multivariate Gaussian distribution to potentially infinite-dimensional function spaces. The key definition is the following.

**Definition 1.** A Gaussian process (GP) is a (potentially infinite) collection of random variables such that the joint distribution of any finite number of them is multivariate Gaussian.

These infinitely many random variables can be thought of as the values of the function  $f$  at every location  $x \in \mathcal{X}$ . The way we extend the finite-dimensional Gaussian distribution to such a collection of variables is by allowing arbitrary inputs but only ever reasoning about finitely many of them at a time.

A GP prior distribution on  $f$  is written

$$p(f) = \mathcal{GP}(f; \mu, K),$$

and just like the multivariate Gaussian distribution, is parameterized by its first two moments (now functions):

- $\mathbb{E}[f] = \mu: \mathcal{X} \rightarrow \mathbb{R}$ , the **mean function**, and
- $\mathbb{E}[(f(x) - \mu(x))(f(x') - \mu(x')))] = K: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ , a **positive semidefinite covariance function or kernel**.<sup>1</sup>

<sup>1</sup>A function is **positive semidefinite** if, for every finite set of points  $\mathbf{X}$ , the **Gram matrix**  $\mathbf{K} = K(\mathbf{X}, \mathbf{X})$  is positive semidefinite. Note that a matrix is a valid covariance matrix if and only if it is positive semidefinite.

The function  $K$  should look familiar! Rather than explicitly construct  $K$ , we simply select any positive semidefinite function, any one of which will define a GP.

The mean function encodes the central tendency of the function, and is often assumed to be a constant (usually zero). The covariance function encodes information about the shape and structure we expect the function to have. A simple and very common example is the **squared exponential** covariance:

$$K(\mathbf{x}, \mathbf{x}'; \lambda, \ell) = \lambda^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right),$$

which encodes the notation that “nearby points should have similar function values.”

Suppose we have selected a GP prior  $\mathcal{GP}(f; \mu, K)$  for the function  $f$ . Consider a finite set of points  $\mathbf{X} \subseteq \mathcal{X}$ . The GP prior on  $f$ , by definition, implies the following joint distribution on the associated function values  $\mathbf{f} = f(\mathbf{X})$ :

$$p(\mathbf{f} \mid \mathbf{X}) = \mathcal{N}(\mathbf{f}; \mu(\mathbf{X}), K(\mathbf{X}, \mathbf{X})).$$

That is, we simply evaluate the mean and covariance functions at  $\mathbf{X}$  and compute the associated multivariate Gaussian distribution.

### Connection to Bayesian linear regression

As it turns out, we have been specifying a Gaussian process on a latent function  $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$  all along! Let us again write down the abstract problem of regression. We have an unknown function  $f: \mathcal{X} \rightarrow \mathbb{R}$ , where  $\mathcal{X}$  is an arbitrary input space (for example  $\mathcal{X} = \mathbb{R}^d$ ).

In linear regression, we made the assumption that  $f$  was linear:  $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}$  where  $\phi$  is some arbitrary feature expansion. Given a Gaussian prior on  $\mathbf{w}$ ,  $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ , and input locations  $\mathbf{X}$ , the prior distribution for the vector of latent values  $\mathbf{f} = f(\mathbf{X}) = \boldsymbol{\Phi} \mathbf{w}$  is a linear transformation of the Gaussian-distributed vector  $\mathbf{w}$ :

$$p(\mathbf{f} \mid \mathbf{X}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\Phi} \boldsymbol{\mu}, \boldsymbol{\Phi} \boldsymbol{\Sigma} \boldsymbol{\Phi}^\top).$$

Therefore we have shown that the joint distribution of any finite combination of our random variables  $\{f(\mathbf{x})\}_{\mathbf{x} \in \mathcal{X}}$  is multivariate Gaussian, and thus according to the above definition we have a Gaussian process! We may identify the mean and covariance functions of the Bayesian linear regression Gaussian process:

$$\mu(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\mu}$$

$$K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \boldsymbol{\Sigma} \phi(\mathbf{x}').$$

By starting over “from the beginning,” we may free ourselves from the constraint of needing to explicitly specify the feature expansion function  $\phi$ . Instead, we may choose arbitrary mean and covariance functions  $\mu$  and  $K$  and reason about  $f$  in an essentially identical manner.

### Posterior GPs

Given a prior GP belief on  $f$ ,  $\mathcal{GP}(f; \mu, K)$ , and after observing some training data  $\mathcal{D} = (\mathbf{X}, \mathbf{f})$ , suppose we wish to make predictions about the value of  $f$  at some test points  $\mathbf{X}^*$ .

We begin by writing the joint distribution between the training function values  $f(\mathbf{X}) = \mathbf{f}$  and the test function values  $f(\mathbf{X}^*) = \mathbf{f}^*$ :

$$p(\mathbf{f}, \mathbf{f}^*) = \mathcal{N}\left(\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix}; \begin{bmatrix} \mu(\mathbf{X}) \\ \mu(\mathbf{X}^*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) & K(\mathbf{X}, \mathbf{X}^*) \\ K(\mathbf{X}^*, \mathbf{X}) & K(\mathbf{X}^*, \mathbf{X}^*) \end{bmatrix}\right).$$



We can then condition this multivariate Gaussian on the known training values  $\mathbf{f}$ , something we already know how to do!

$$p(\mathbf{f}^* | \mathbf{X}^*, \mathcal{D}) = \mathcal{N}(\mathbf{f}^*; \mu_{f|\mathcal{D}}(\mathbf{X}^*), K_{f|\mathcal{D}}(\mathbf{X}^*, \mathbf{X}^*)),$$

where

$$\begin{aligned}\mu_{f|\mathcal{D}}(\mathbf{x}) &= \mu(\mathbf{x}) + K(\mathbf{x}, \mathbf{X})\mathbf{K}^{-1}(\mathbf{f} - \mu(\mathbf{X})) \\ K_{f|\mathcal{D}}(\mathbf{x}, \mathbf{x}') &= K(\mathbf{x}, \mathbf{x}') - K(\mathbf{x}, \mathbf{X})\mathbf{K}^{-1}K(\mathbf{X}, \mathbf{x}').\end{aligned}$$

Notice that the functions  $\mu_{f|\mathcal{D}}$  and  $K_{f|\mathcal{D}}$  are valid mean and covariance functions, respectively. This means that the posterior distribution over  $f$  is itself a Gaussian process! This can also be inferred from the fact that the posterior distribution over any finite set of test points  $\mathbf{X}^*$  is a multivariate Gaussian.

Figure 1 shows an example of going from a GP prior to a GP posterior; the prior is a zero-mean GP with a squared exponential covariance function where  $\lambda = \ell = 1$ .

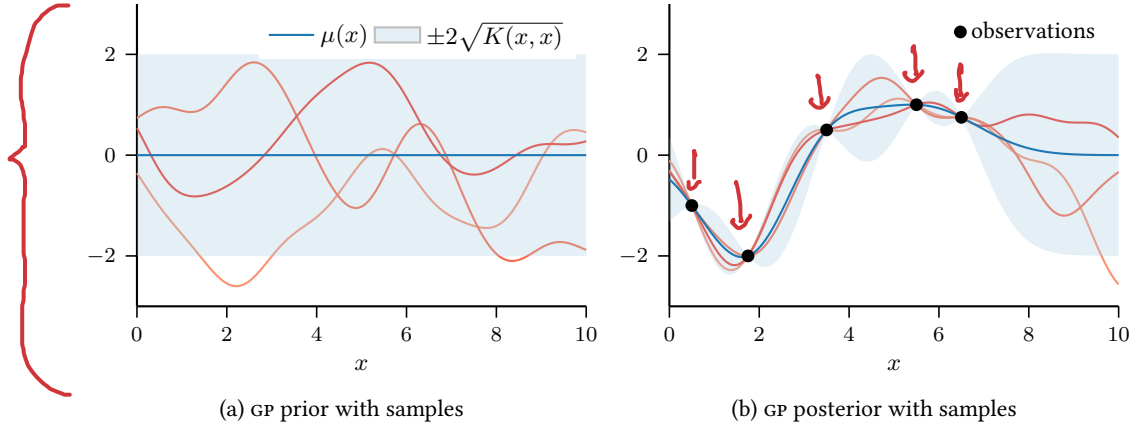


Figure 1: GP beliefs before and after observing data. Each figure also shows samples drawn from the depicted GP belief to give a sense for the kind of functions modeled by the belief: note how the squared exponential kernel corresponds to smooth sample functions.

### Dealing with noise

GPs deal with observation noise in effectively the same way as Bayesian linear regression. Here we will continue to assume that the observed values  $\mathbf{y}$  are generated by adding zero-mean, independent Gaussian noise with variance  $\sigma^2$  to the true function values  $\mathbf{f} = f(\mathbf{X})$ . Probabilistically, we write

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

Starting with a GP prior on  $f$ ,  $\mathcal{GP}(f; \mu, K)$ , but this time given noisy observations  $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ , we again write the joint distribution between the training function values  $\mathbf{y}$  and the test function values  $\mathbf{f}^*$ :

$$p(\mathbf{y}, \mathbf{f}^*) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y} \\ \mathbf{f}^* \end{bmatrix}; \begin{bmatrix} \mu(\mathbf{X}) \\ \mu(\mathbf{X}^*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I} & K(\mathbf{X}, \mathbf{X}^*) \\ K(\mathbf{X}^*, \mathbf{X}) & K(\mathbf{X}^*, \mathbf{X}^*) \end{bmatrix}\right).$$

Conditioning as before, we end up with:

$$p(\mathbf{f}^* | \mathbf{X}^*, \mathcal{D}) = \mathcal{N}(\mathbf{f}^*; \mu_{f|\mathcal{D}}(\mathbf{X}^*), K_{f|\mathcal{D}}(\mathbf{X}^*, \mathbf{X}^*)),$$

where

$$\mu_{f|\mathcal{D}}(\mathbf{x}) = \mu(\mathbf{x}) + K(\mathbf{x}, \mathbf{X}) (K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I})^{-1} (\mathbf{y} - \mu(\mathbf{X}))$$

$$K_{f|\mathcal{D}}(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}, \mathbf{x}') - K(\mathbf{x}, \mathbf{X}) (K(\mathbf{X}, \mathbf{X}) + \sigma^2 \mathbf{I})^{-1} K(\mathbf{X}, \mathbf{x}')$$

Figure 2 shows the effect of different noise levels  $\sigma$ , using the same GP prior and data set depicted in Figure 1.

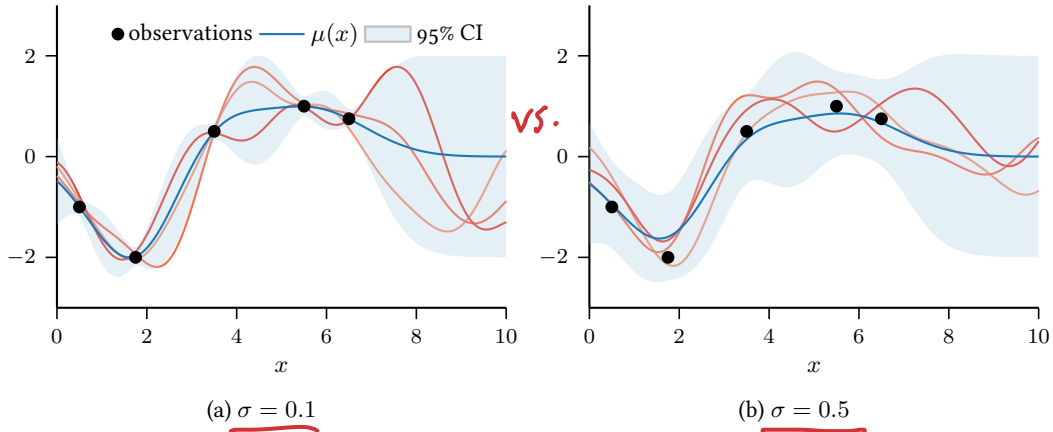


Figure 2: Noisy GP posteriors with samples: note how in the setting with higher noise, the mean and samples no longer pass directly through the observations as more of the observations' behavior is ascribed to noise.

## Hyperparameters

So far, we have assumed that the Gaussian process prior distribution on  $f$  has been specified *a priori*. But this prior distribution itself will generally have parameters that we need to specify: for example, the kernel length scale  $\ell$ , the kernel output scale  $\lambda$ , and the noise variance  $\sigma^2$ . As parameters of a prior distribution, we call these **hyperparameters**.

For notational convenience, we will write  $\theta$  to denote the vector of all hyperparameters of the model (including of  $\mu$  and  $K$ ). Assume we have chosen a prior

$$p(f | \theta) = \mathcal{GP}(f; \mu(\mathbf{x}; \theta), K(\mathbf{x}, \mathbf{x}'; \theta))$$

where the dependence of  $\mu$  and  $K$  on the parameters  $\theta$  has been made explicit. We will measure the quality of the fit to our training data  $\mathcal{D} = (\mathbf{X}, \mathbf{y})$  with the **marginal likelihood**, the probability of observing  $\mathcal{D}$  under our prior:

$$p(\mathbf{y} | \mathbf{X}, \theta) = \int p(\mathbf{y} | \mathbf{f}) p(\mathbf{f} | \mathbf{X}, \theta) d\mathbf{f},$$

where we have marginalized the unknown function values  $\mathbf{f}$  (hence, marginal likelihood).

Thankfully, this is an integral we can do analytically under the additive Gaussian noise assumption!

$$\begin{aligned}
 p(\mathbf{y} \mid \mathbf{X}, \theta) &= \int p(\mathbf{y} \mid \mathbf{f}) p(\mathbf{f} \mid \mathbf{X}, \theta) d\mathbf{f} \\
 &= \int \mathcal{N}(\mathbf{y}; \mathbf{f}, \sigma^2 \mathbf{I}) \mathcal{N}(\mathbf{f}; \mu(\mathbf{X}; \theta), K(\mathbf{X}, \mathbf{X}; \theta)) d\mathbf{f} \\
 &= \mathcal{N}(\mathbf{y}; \mu(\mathbf{X}; \theta), K(\mathbf{X}, \mathbf{X}; \theta) + \sigma^2 \mathbf{I})
 \end{aligned}$$

where we have made use of the closure under convolutions property of Gaussians.

The log-likelihood of our data under the chosen prior is then

$$\begin{aligned}
 \log p(\mathbf{y} \mid \mathbf{X}, \theta) &= -\frac{(\mathbf{y} - \mu(\mathbf{X}; \theta))^T (K(\mathbf{X}, \mathbf{X}; \theta) + \sigma^2 \mathbf{I})^{-1} (\mathbf{y} - \mu(\mathbf{X}; \theta))}{2} \\
 &= \frac{\log \det(K(\mathbf{X}, \mathbf{X}; \theta) + \sigma^2 \mathbf{I})}{2} - \frac{N \log 2\pi}{2}
 \end{aligned}$$

There is an inherent trade off between the first two terms in this expression: the first term is large when the data fit the model well and the second term is large when the volume of the prior covariance is small, i.e., when the model is simpler.

Figure 3 shows the effect of different settings of the model hyperparameters,  $(\lambda, \ell, \sigma)$ , when conditioning a GP prior on a fixed set of observations to arrive at a GP posterior: note how in the setting with higher noise, the mean and samples no longer pass directly through the observations as more of the observations' behavior is ascribed to noise.

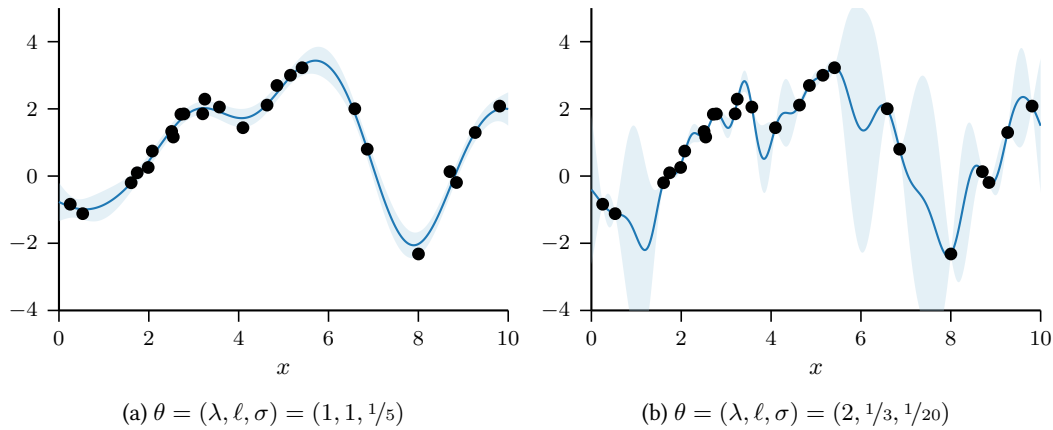


Figure 3: GPs with different hyperparameters fit to the same data set: the log-likelihood of the data under the hyperparameter setting in Figure 3a is  $-27.6$  and under the hyperparameter setting in Figure 3b, it is  $-46.5$ .

To be fully Bayesian, we would choose a **hyperprior** over  $\theta$ ,  $p(\theta)$ , and marginalize the unknown hyperparameters when making predictions:

$$p(f^* \mid \mathbf{x}^*, \mathcal{D}) = \frac{\int p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta) p(\mathbf{y} \mid \mathbf{X}, \theta) p(\theta) d\theta}{\int p(\mathbf{y} \mid \mathbf{X}, \theta) p(\theta) d\theta}$$

Unfortunately, this integral cannot generally be resolved analytically (of course...).

Instead, if we believe the posterior distribution over  $\theta$  is well-concentrated (for example, if we have many training examples), we may approximate  $p(\theta \mid \mathcal{D})$  with a Dirac delta distribution at the point with maximum marginal likelihood:

$$\theta_{\text{MLE}} = \arg \max_{\theta} p(\mathbf{y} \mid \mathbf{X}, \theta).$$

This is called **maximum likelihood-II** (ML-II) inference and effectively makes the approximation

$$p(f^* \mid \mathbf{x}^*, \mathcal{D}) \approx p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta_{\text{MLE}}).$$

As long as  $\mu(\mathbf{X}; \theta)$  and  $K(\mathbf{X}, \mathbf{X}; \theta)$  are differentiable w.r.t.  $\theta$ , then we can compute the gradient  $\partial \log p(\mathbf{y} \mid \mathbf{X}, \theta) / \partial \theta$ . This allows us to find  $\theta_{\text{MLE}}$  by minimizing the negative log likelihood (which is equivalent to maximizing the likelihood) using off-the-shelf gradient-based methods.

## Gaussian Process Classification

Just as we could use the kernel trick to extend Bayesian linear regression to Gaussian processes for general-purpose nonlinear regression, we may also extend Bayesian linear classification in the same way. In Gaussian process classification, we assume there is a latent function  $f: \mathcal{X} \rightarrow \mathbb{R}$  that is commensurate with the probability of a positive observation; higher latent function values correspond to higher probabilities of positive observations. In Bayesian linear classification, we assumed a parametric (linear) form for this latent function:

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}.$$

In Gaussian process classification, rather than choosing a parametric form for  $f$ , we instead place a Gaussian process prior on  $f$ :

$$p(f) = \mathcal{GP}(f; \mu, K).$$

Note that a Gaussian prior on the weight vector  $\mathbf{w}$  above induces a Gaussian process prior on  $f$  with mean function

$$\mu(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\mu}$$

and covariance function

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x},$$

where  $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ . The Gaussian process formalism allows us to model arbitrary nonlinear classification boundaries by using any desired mean and covariance function for  $f$ .

### Likelihood

Suppose we have made binary observations at a set of values  $\mathbf{X}$ , and define  $\mathbf{f} = f(\mathbf{X})$  to be the associated set of latent function values. As we did when discussing Bayesian logistic regression, we will assume the following likelihood for a given label observation  $y_i$  associated with  $\mathbf{x}_i$ :

$$p(y_i = 1 \mid f_i) = \sigma(f_i),$$

where  $\sigma: \mathbb{R} \rightarrow (0, 1)$  is a monotonically increasing sigmoid function such as the logistic function or the standard normal CDF. We again assume the observations are conditionally independent given the latent function values:

$$p(\mathbf{y} \mid \mathbf{f}) = \prod p(y_i \mid f_i).$$

### Inference

Given our prior  $p(f) = \mathcal{GP}(f; \mu, K)$  and a set of observations  $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ , we wish to find the posterior distribution of the latent function values  $\mathbf{f} = f(\mathbf{X})$ . Note that if we had a Gaussian posterior  $\mathbf{f}$ , this would induce a GP posterior for the function  $f$  given  $\mathcal{D}$ . The posterior is

$$p(\mathbf{f} \mid \mathcal{D}) = \frac{1}{Z} p(\mathbf{f} \mid \mathbf{X}) p(\mathbf{y} \mid \mathbf{f}) = \mathcal{N}(\mathbf{f}; \mu(\mathbf{X}), K(\mathbf{X}, \mathbf{X})) \prod_i p(y_i \mid f_i).$$

Unfortunately, as we saw, the sigmoid likelihood coupled with the Gaussian prior does not form a tractable posterior. Instead, we must approximate this posterior in some way. Previously we described the Laplace approximation, which approximates the unnormalized log posterior with a second-order Taylor expansion, resulting in a Gaussian approximate posterior centered at the posterior mode

$$\hat{\mathbf{f}} = \arg \max_{\mathbf{f}} p(\mathbf{f} \mid \mathcal{D}).$$

Here we will consider another approximation technique for approximating intractable posterior distributions, which is useful when the likelihood factorizes into one-dimensional terms, as in GP classification when the labels are assumed to be conditionally independent.

## Assumed Density Filtering

Consider a posterior distribution of the form

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{1}{Z} p_0(\boldsymbol{\theta}) \prod_{i=1}^n t_i(\boldsymbol{\theta}),$$

where the  $t_i$  are typically likelihood terms, for example our  $p(y_i \mid f_i)$  above. We assume the prior  $p_0(\boldsymbol{\theta})$  has been chosen to be of some nice form, for example a Gaussian, and we will use the Gaussian case to illustrate the idea below.

In **assumed density filtering** (ADF), we assume that the posterior has the same form as the prior  $p_0$ , and we seek an approximating distribution

$$q(\boldsymbol{\theta}) \approx p(\boldsymbol{\theta} \mid \mathcal{D})$$

from the same family as  $p_0$  that approximates the true posterior “as well as possible.”

Mechanically, ensuring that our approximate distribution  $q$  remains in the same family as  $p_0$  is achieved by selecting an (unnormalized) member  $\tilde{t}_i$  from the likelihood conjugate to the prior for each of the likelihood terms  $t_i$ . The result is

from the same family

$$q(\boldsymbol{\theta}) = p_0(\boldsymbol{\theta}) \prod_{i=1}^n \tilde{Z}_i \tilde{t}_i(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_i),$$

and this product will belong to the desired family by **conjugacy**. Here the constants  $\tilde{Z}_i$  and the local parameter vectors  $\{\tilde{\boldsymbol{\eta}}_i\}$  are free parameters, called **site parameters**. We will choose the site parameters for each of the approximating distributions  $\tilde{t}_i$  to try to improve the fit of the approximating distribution.

For example, the Gaussian distribution is self-conjugate, so the ADF approximation will in this case take the form

$$q(\boldsymbol{\theta}) = p_0(\boldsymbol{\theta}) \prod_{i=1}^n t_i(\boldsymbol{\theta}) \approx \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{i=1}^n \tilde{Z}_i \mathcal{N}(\boldsymbol{\theta}; \tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\Sigma}}_i),$$

where the site parameters are the constants  $\tilde{Z}_i$  (chosen so that the approximation normalizes) as well as the local mean vectors  $\tilde{\boldsymbol{\mu}}_i$  and covariance matrices  $\tilde{\boldsymbol{\Sigma}}_i$ .

Note that in the GP classification case, the likelihood terms in the product are (very conveniently) one-dimensional, because each term simply models a single scalar label,  $y_i$ , and only depends on a single latent function value  $f_i$ :

$$q(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{i=1}^n \tilde{Z}_i \mathcal{N}(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2).$$

Consider just the first two terms in this product:

$$p(\boldsymbol{\theta} \mid \mathcal{D}_1) \propto p_0(\boldsymbol{\theta})t_1(\boldsymbol{\theta}).$$

This product will not have the same nice form as the prior, but has only been warped “slightly” away from the prior via a single likelihood term. In many cases, this distribution will be at least partially manageable. Perhaps there will not be a nice closed expression, but we might still be able to compute the normalizing constant or the moments of the posterior.

In assumed density filtering, we will approximate this product with a member of the desired family:

$$q_1(\boldsymbol{\theta}) = \tilde{Z}_1 p_0(\boldsymbol{\theta}) \tilde{t}_1(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_1) \approx p(\boldsymbol{\theta} \mid \mathcal{D}_1) \propto p_0(\boldsymbol{\theta})t_1(\boldsymbol{\theta}).$$

This approximation is done by matching the moments between the approximation  $q_1(\boldsymbol{\theta})$  and the true posterior  $p(\boldsymbol{\theta} \mid \mathcal{D}_1)$ .

For example, consider  $p_0(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ . We select the site parameters  $\boldsymbol{\eta} = \{\tilde{Z}_1^{-1}, \tilde{\boldsymbol{\mu}}_1, \tilde{\boldsymbol{\Sigma}}_1\}$  such that

$$\left\{ \begin{array}{l} \int q_1(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1 \\ \mathbb{E}[q_1(\boldsymbol{\theta})] = \mathbb{E}[p(\boldsymbol{\theta} \mid \mathcal{D}_1)] \\ \text{cov}[q_1(\boldsymbol{\theta})] = \text{cov}[p(\boldsymbol{\theta} \mid \mathcal{D}_1)]. \end{array} \right.$$

Now the approximate distribution is in the desired density family (Gaussians) and matches the true posterior up to the second moment.

Now consider the first three terms of the product:

$$p(\boldsymbol{\theta} \mid \mathcal{D}_1, \mathcal{D}_2) \propto p_0(\boldsymbol{\theta})t_1(\boldsymbol{\theta})t_2(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta} \mid \mathcal{D}_1)t_2(\boldsymbol{\theta}).$$

The idea in assumed density filtering is to substitute in our approximation  $q_1(\boldsymbol{\theta})$ , giving:

$$p(\boldsymbol{\theta} \mid \mathcal{D}_1, \mathcal{D}_2) \approx q_1(\boldsymbol{\theta})t_2(\boldsymbol{\theta}).$$

Now we are in the same situation we were in before! We have a nice “prior” distribution  $q_1$  multiplied by a single likelihood term  $t_2$ . We proceed as before, replacing the true likelihood term  $t_2$  with an approximate (conjugate) term  $\tilde{Z}_2 \tilde{t}_2(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_2)$ , where we again choose the site parameters  $(\tilde{Z}_2, \tilde{\boldsymbol{\eta}}_2)$  to match the moments between our new approximation

$$q_2(\boldsymbol{\theta}) = \tilde{Z}_2 q_1(\boldsymbol{\theta}) \tilde{t}_2(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_2) = \tilde{Z}_1 \tilde{Z}_2 p_0(\boldsymbol{\theta}) \tilde{t}_1(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_1) \tilde{t}_2(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_2).$$

and the “less-approximate” posterior  $q_1(\boldsymbol{\theta})t_2(\boldsymbol{\theta})$ . We proceed in this fashion until we have processed all the local likelihood terms, resulting in the final approximation

$$q(\boldsymbol{\theta}) = p_0(\boldsymbol{\theta}) \prod_{i=1}^n \tilde{Z}_i \tilde{t}_i(\boldsymbol{\theta}; \tilde{\boldsymbol{\eta}}_i).$$

### Moments for Bayesian Probit Regression

As an example, suppose we use the standard normal CDF as our sigmoid function

$$\sigma(f_i) = \Phi(f_i) = \int_{-\infty}^{f_i} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx.$$

In this setting, we can compute the site parameters  $\tilde{Z}_i$ ,  $\tilde{\mu}_i$ , and  $\tilde{\sigma}_i^2$  in closed-form. To do so, first observe that if we encode the labels as  $y_i \in \{-1, +1\}$ , we can succinctly express the likelihood as

$$\Pr(\mathbf{y} \mid \mathbf{f}) = \prod_{i=1}^n \Phi(y_i f_i)$$

where we have made use of the fact that

$$\Pr(y_i = -1 \mid f_i) = 1 - \Pr(y_i = +1 \mid f_i) = 1 - \Phi(f_i) = \Phi(-f_i).$$

Now suppose that the product of the prior and the first  $i - 1$  likelihood terms has already been approximated as

$$q_{i-1}(\mathbf{f}) \approx \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}^{(i-1)}, \boldsymbol{\Sigma}^{(i-1)})$$

and we are interested in computing the moments of

$$p(\mathbf{f} \mid \mathbf{X}_{1:i}) \propto q_{i-1}(\mathbf{f}) t_i(\mathbf{f})$$

where  $t_i(\mathbf{f}) = \Phi(f_i)$ . First, consider the normalizing constant

$$\begin{aligned} \tilde{Z}_i^{-1} &= \zeta = \int q_{i-1}(\mathbf{f}) t_i(\mathbf{f}) d\mathbf{f} \\ &= \int_{-\infty}^{\infty} \mathcal{N}\left(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}\right) \Phi(y_i f_i) df_i \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{y_i f_i} \frac{1}{\sqrt{2\pi}} e^{-\frac{a^2}{2}} \frac{1}{\sqrt{2\pi \Sigma_{ii}^{(i-1)}}} e^{-\frac{(f_i - \mu_i^{(i-1)})^2}{2\Sigma_{ii}^{(i-1)}}} da df_i \end{aligned}$$

where we have made use of the (again, incredibly convenient) fact that the likelihood factorizes along individual latent function values and the closure property of Gaussians under marginalization. Note that the superscript  $(i - 1)$  indicates that the mean and variance come from the current approximate belief  $q_{i-1}(\mathbf{f})$ .

Let us first consider the case where  $y_i = +1$ . We will define two intermediate quantities:

$$\begin{aligned} \alpha &= a - f_i + \mu_i^{(i-1)} \rightarrow d\alpha = da \text{ and } a = (+1)f_i \rightarrow \alpha = \mu_i^{(i-1)} \\ \beta &= f_i - \mu_i^{(i-1)} \rightarrow d\beta = df_i \\ &\rightarrow a = \alpha + \beta \end{aligned}$$



Using these substitutions and swapping the order of the integrals gives

$$\begin{aligned}
\zeta_{y_i=+1} &= \int_{-\infty}^{\mu_i^{(i-1)}} \int_{-\infty}^{\infty} \frac{1}{2\pi\sqrt{\Sigma_{ii}^{(i-1)}}} e^{-\frac{(\alpha+\beta)^2}{2} - \frac{\beta^2}{2\Sigma_{ii}^{(i-1)}}} d\beta d\alpha \\
&= \int_{-\infty}^{\mu_i^{(i-1)}} \int_{-\infty}^{\infty} \frac{1}{2\pi\sqrt{\Sigma_{ii}^{(i-1)}}} e^{-\frac{1}{2} \begin{bmatrix} \beta \\ \alpha \end{bmatrix}^T \begin{bmatrix} 1 + \frac{1}{\Sigma_{ii}^{(i-1)}} & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \beta \\ \alpha \end{bmatrix}} d\beta d\alpha \\
&= \int_{-\infty}^{\mu_i^{(i-1)}} \int_{-\infty}^{\infty} \mathcal{N} \left( \begin{bmatrix} \beta \\ \alpha \end{bmatrix}; \mathbf{0}, \begin{bmatrix} \Sigma_{ii}^{(i-1)} & -\Sigma_{ii}^{(i-1)} \\ -\Sigma_{ii}^{(i-1)} & 1 + \Sigma_{ii}^{(i-1)} \end{bmatrix} \right) d\beta d\alpha \\
&= \int_{-\infty}^{\mu_i^{(i-1)}} \mathcal{N}(\alpha; 0, 1 + \Sigma_{ii}^{(i-1)}) d\alpha = \Phi \left( \frac{\mu_i^{(i-1)}}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}} \right).
\end{aligned}$$

For the case where  $y_i = -1$ , the procedure is similar except we define the substitution variables as

$$\begin{aligned}
\alpha &= a + f_i - \mu_i \rightarrow d\alpha = da \text{ and } a = (-1)f_i \rightarrow \alpha = -\mu_i \\
\beta &= f_i - \mu_i \rightarrow d\beta = df_i \\
&\rightarrow a = \alpha - \beta
\end{aligned}$$

Using these new substitutions and stepping through the same procedure as above gives an unsurprisingly similar result:

$$\zeta_{y_i=-1} = \Phi \left( \frac{-\mu_i^{(i-1)}}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}} \right).$$

Thus, we can collect the two results into a single, concise expression by again using the fact that the labels are encoded as  $y_i \in \{-1, +1\}$ :

$$\zeta = \Phi \left( \frac{y_i \mu_i^{(i-1)}}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}} \right) := \Phi(z_i)$$

Next, we need to compute the first two moments of the true posterior:

$$\begin{aligned}
\mathbb{E}[f_i] &= \int_{-\infty}^{\infty} f_i \frac{1}{\zeta} \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) \Phi(y_i f_i) df_i \\
\mathbb{E}[f_i^2] &= \int_{-\infty}^{\infty} f_i^2 \frac{1}{\zeta} \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) \Phi(y_i f_i) df_i
\end{aligned}$$

To compute these, first observe that

$$\begin{aligned}
\frac{\partial \zeta}{\partial \mu_i^{(i-1)}} &= \int_{-\infty}^{\infty} \Phi(y_i f_i) \frac{\partial \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)})}{\partial \mu_i^{(i-1)}} df_i \\
&= \int_{-\infty}^{\infty} \frac{f_i - \mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} \Phi(y_i f_i) \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) df_i \\
&= \frac{1}{\Sigma_{ii}^{(i-1)}} \int_{-\infty}^{\infty} f_i \Phi(y_i f_i) \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) df_i - \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} \zeta
\end{aligned}$$

and

$$\frac{\partial \zeta}{\partial \mu_i^{(i-1)}} = \frac{\partial}{\partial \mu_i^{(i-1)}} \Phi(z_i) = \mathcal{N}(z_i) \frac{\partial z_i}{\partial \mu_i^{(i-1)}} = \mathcal{N}(z_i) \frac{y_i}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}}$$

where  $\mathcal{N}(\cdot)$  is the standard normal PDF. The first term in the top equation is precisely the first moment we wish to compute multiplied by  $\frac{\zeta}{\Sigma_{ii}^{(i-1)}}$ ! Thus, we can derive an expression for the first moment by setting the two (equivalent) expressions for  $\frac{\partial \zeta}{\partial \mu_i^{(i-1)}}$  equal to each other and solving:

$$\begin{aligned}
&\frac{1}{\Sigma_{ii}^{(i-1)}} \int_{-\infty}^{\infty} f_i \Phi(y_i f_i) \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) df_i - \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} \zeta = \mathcal{N}(z_i) \frac{y_i}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}} \\
&\rightarrow \frac{\zeta}{\Sigma_{ii}^{(i-1)}} \int_{-\infty}^{\infty} f_i \frac{1}{\zeta} \Phi(y_i f_i) \mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) df_i = \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} \zeta + \mathcal{N}(z_i) \frac{y_i}{\sqrt{1 + \Sigma_{ii}^{(i-1)}}} \\
&\rightarrow \mathbb{E}[f_i] = \mu_i^{(i-1)} + \mathcal{N}(z_i) \frac{y_i \Sigma_{ii}^{(i-1)}}{\zeta \sqrt{1 + \Sigma_{ii}^{(i-1)}}} := \mu_i^{(i-1)} + \frac{\mathcal{N}(z_i) s_i}{\zeta}
\end{aligned}$$

Similarly, taking the second derivative of  $\zeta$  with respect to  $\mu_i$  gives

$$\begin{aligned}
\frac{\partial^2 \zeta}{\partial \mu_i^{(i-1)^2}} &= \frac{1}{\Sigma_{ii}^{(i-1)^2}} \int_{-\infty}^{\infty} \left( f_i^2 - f_i \mu_i^{(i-1)} \right) \Phi(y_i f_i) \mathcal{N} \left( f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)} \right) df_i \\
&\quad - \frac{1}{\Sigma_{ii}^{(i-1)}} \int_{-\infty}^{\infty} \Phi(y_i f_i) \mathcal{N} \left( f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)} \right) df_i \\
&\quad - \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)^2}} \int_{-\infty}^{\infty} \left( f_i - \mu_i^{(i-1)} \right) \Phi(y_i f_i) \mathcal{N} \left( f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)} \right) df_i \\
&= \frac{1}{\Sigma_{ii}^{(i-1)^2}} \int_{-\infty}^{\infty} f_i^2 \Phi(y_i f_i) \mathcal{N} \left( f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)} \right) df_i \\
&\quad - 2 \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)^2}} \int_{-\infty}^{\infty} f_i \Phi(y_i f_i) \mathcal{N} \left( f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)} \right) df_i - \frac{1}{\Sigma_{ii}^{(i-1)}} \zeta + \frac{\mu_i^{(i-1)^2}}{\Sigma_{ii}^{(i-1)^2}} \zeta
\end{aligned}$$

and

$$\frac{\partial^2 \zeta}{\partial \mu_i^{(i-1)^2}} = -\mathcal{N}(z_i) \frac{z_i}{1 + \Sigma_{ii}^{(i-1)}}$$

Setting these two equal and solving for the (normalized) first term gives

$$\begin{aligned}
\frac{1}{\Sigma_{ii}^{(i-1)^2}} \zeta \mathbb{E}[f_i^2] - 2 \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)^2}} \zeta \mathbb{E}[f_i] - \frac{1}{\Sigma_{ii}^{(i-1)}} \zeta + \frac{\mu_i^{(i-1)^2}}{\Sigma_{ii}^{(i-1)^2}} \zeta &= -\mathcal{N}(z_i) \frac{z_i}{1 + \Sigma_{ii}^{(i-1)}} \\
\rightarrow \mathbb{E}[f_i^2] &= 2\mu_i^{(i-1)} \mathbb{E}[f_i] + \Sigma_{ii}^{(i-1)} - \mu_i^{(i-1)^2} - \mathcal{N}(z_i) \frac{\Sigma_{ii}^{(i-1)^2} z_i}{\zeta \left( 1 + \Sigma_{ii}^{(i-1)} \right)} \\
&:= 2\mu_i^{(i-1)} \mathbb{E}[f_i] + \Sigma_{ii}^{(i-1)} - \mu_i^{(i-1)^2} - \frac{\mathcal{N}(z_i) s_i^2 z_i}{\zeta}.
\end{aligned}$$

Finally, the variance of  $f_i$  can be computed as

$$\begin{aligned}
\text{var}(f_i) &= \mathbb{E}[f_i^2] - \mathbb{E}[f_i]^2 = 2\mu_i^{(i-1)} \mathbb{E}[f_i] + \Sigma_{ii}^{(i-1)} - \mu_i^{(i-1)^2} - \frac{\mathcal{N}(z_i) s_i^2 z_i}{\zeta} \\
&\quad - \mu_i^{(i-1)^2} - 2\mu_i^{(i-1)} \frac{\mathcal{N}(z_i) s_i}{\zeta} - \left( \frac{\mathcal{N}(z_i) s_i}{\zeta} \right)^2 \\
&= \Sigma_{ii}^{(i-1)} - \frac{\mathcal{N}(z_i) s_i^2}{\zeta} \left( z_i + \frac{\mathcal{N}(z_i)}{\zeta} \right)
\end{aligned}$$

Incredibly, we are only part way through the derivation! The next step is to compute the site parameters,  $\tilde{\mu}_i$  and  $\tilde{\sigma}_i^2$ , such that the product  $q_{i-1}(f_i)\tilde{t}_i(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)$  is a Gaussian with the mean and variance that we just computed.

To do so, recall that the product of two independent Gaussian PDFs is an unnormalized Gaussian i.e.

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, ) \mathcal{N}(\mathbf{x}; \boldsymbol{\nu}, \mathbf{P}) = \frac{1}{Z} \mathcal{N}(\mathbf{x}; \boldsymbol{\omega}, \mathbf{T}),$$

where

$$\mathbf{T} = (\boldsymbol{\Sigma}^{-1} + \mathbf{P}^{-1})^{-1} \text{ and } \boldsymbol{\omega} = \mathbf{T}(\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathbf{P}^{-1}\boldsymbol{\nu})$$

In the context of assumed density filtering for Gaussian process classification, we want

$$\mathcal{N}(f_i; \mu_i^{(i-1)}, \Sigma_{ii}^{(i-1)}) \mathcal{N}(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2) \propto \mathcal{N}(f_i; \mathbb{E}[f_i], \text{var}(f_i)).$$

Thus, we can solve for  $\tilde{\mu}_i$  and  $\tilde{\sigma}_i^2$  as follows:

$$\begin{aligned} \text{var}(f_i) &= \left( \frac{1}{\Sigma_{ii}^{(i-1)}} + \frac{1}{\tilde{\sigma}_i^2} \right)^{-1} \rightarrow \tilde{\sigma}_i^2 = \left( \frac{1}{\text{var}(f_i)} - \frac{1}{\Sigma_{ii}^{(i-1)}} \right)^{-1} \\ \mathbb{E}[f_i] &= \text{var}(f_i) \left( \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} + \frac{\tilde{\mu}_i}{\tilde{\sigma}_i^2} \right) \rightarrow \tilde{\mu}_i = \tilde{\sigma}_i^2 \left( \frac{\mathbb{E}[f_i]}{\text{var}(f_i)} - \frac{\mu_i^{(i-1)}}{\Sigma_{ii}^{(i-1)}} \right). \end{aligned}$$

Finally, we can now combine  $q_{i-1}(\mathbf{f})$  with our newly computed approximation  $\mathcal{N}(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)$  to get the next approximation in the sequence,  $q_i(\mathbf{f})$ .

However, to do so, we now need to “project” our 1-dimensional likelihood term back into the full  $n$ -dimensional space corresponding to  $\mathbf{f}$ . We can do this by constructing

- the matrix  $\mathbf{S}$ , an  $n \times n$  matrix of all zeros except for the  $i^{\text{th}}$  diagonal element, which is equal to  $\frac{1}{\tilde{\sigma}_i^2}$  and
- the vector  $\mathbf{m}$ , which is an  $n$  length vector of all zeros except for the  $i^{\text{th}}$  element, which is  $\frac{\tilde{\mu}_i}{\tilde{\sigma}_i^2}$ .

Then the updated approximation can be expressed as

$$q_i(\mathbf{f}) \propto q_{i-1}(\mathbf{f}) \tilde{t}_i(\mathbf{f}; \tilde{\boldsymbol{\eta}}_i) \propto \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}^{(i)}, \boldsymbol{\Sigma}^{(i)})$$

where we again apply the closure of Gaussian PDFs to get

$$\boldsymbol{\Sigma}^{(i)} = \left( \boldsymbol{\Sigma}^{(i-1)^{-1}} + \mathbf{S} \right)^{-1} \text{ and } \boldsymbol{\mu}^{(i)} = \boldsymbol{\Sigma}^{(i)} \left( \boldsymbol{\Sigma}^{(i-1)^{-1}} \boldsymbol{\mu}^{(i-1)} + \mathbf{m} \right).$$

We would then repeat this process for  $i+1, i+2, \dots$  and so on until all  $n$  likelihood terms have been approximated as Gaussians, leading to a final Gaussian approximation for the intractable posterior.

compare and contrast w/ ADF

## Expectation Propagation

One thing to note about assumed density filtering is that our final approximation is dependent on the sequence in which we process the likelihood terms  $\{t_i\}$ . Note that **we are always matching the moments between**

$$q_{i-1}(\theta) t_i(\theta),$$

the approximation using data up to the  $i^{\text{th}}$  term as well as the true  $i^{\text{th}}$  likelihood term, and the new approximation

$$q_i(\theta) = \tilde{Z}_i q_{i-1}(\theta) \tilde{t}_i(\theta; \tilde{\theta}_i).$$

This moment matching at time  $i$  therefore never considers data that appears in future terms. For this reason, we might accumulate errors and/or wish to later “revisit” a particular term and update the site parameters  $(\tilde{Z}_i, \tilde{\theta}_i)$  in light of future data. This idea leads to **expectation propagation**, a refinement of assumed density filtering that can address some of these issues.

The idea is simple: once we have processed each of the likelihood terms, resulting in the approximation

$$q(\theta) = p_0(\theta) \prod_{i=1}^n \tilde{Z}_i \tilde{t}_i(\theta; \tilde{\theta}_i),$$

we repeatedly revisit each term and update its site parameters. First, we **select a site  $1 \leq i \leq n$  to update, then form the so-called cavity distribution**, which is our approximation using all but the  $i^{\text{th}}$  term in the product:

$$q_{-i}(\theta) = p_0(\theta) \prod_{j \neq i} \tilde{Z}_j \tilde{t}_j(\theta; \tilde{\theta}_j).$$

Another way of conceptualizing the cavity distribution is that we have divided the previous approximation by the old site term  $\tilde{Z}_i \tilde{t}_i(\theta; \tilde{\theta}_i)$ . Next, we replace the removed site term with the true likelihood term  $t_i$ , forming the **tilted distribution**

$$q_{-i}(\theta) t_i(\theta).$$

Finally, we select new site parameters to (re)**match the moments between the tilted distribution** and our new approximation:

$$q_{\text{new}}(\theta) = \tilde{Z}_i q_{-i}(\theta) \tilde{t}_i(\theta; \tilde{\theta}_i).$$

setting these

We proceed continually updating site parameters in this manner until we reach convergence (i.e., none of the site parameters changes very much) or we expend a chosen computational budget.

Note that we do not need to use assumed density filtering to first initialize the site parameters, we can use basically any **initialization scheme**; a common alternative is to simply initialize all of the site parameters to  $\tilde{Z}_i = 1$ ,  $\tilde{\mu}_i = 0$  and  $\tilde{\sigma}_i = \infty \forall 1 \leq i \leq n$ , making the initial approximation  $q(\theta) = p_0(\theta)$ .

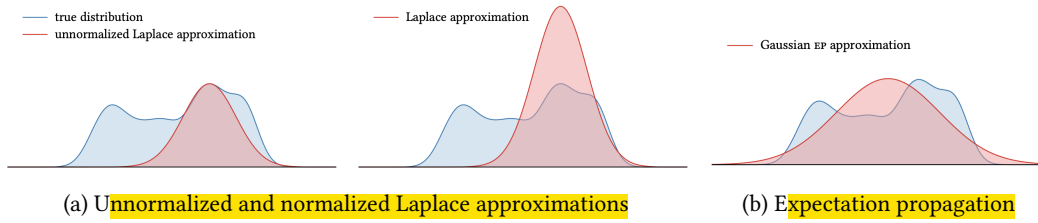


Figure 1: A comparison of the Laplace approximation and expectation propagation on a one-dimensional, intractable posterior. *compare ~ contrast*

Figures 1a and 1b compare the Laplace approximation and expectation propagation on a one-dimensional posterior: while the Laplace approximation does an okay job representing the posterior around its mode, expectation propagation is better at capturing the entirety of the distribution.

### EP for Bayesian Probit Regression

Continuing our previous example, suppose that we have used assumed density filtering to approximate the intractable posterior on the latent function  $f$  for Bayesian probit regression with the standard normal CDF as our sigmoid function:

$$\frac{1}{Z} \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, ) \prod_{i=1}^n \Phi(y_i f_i) \approx q(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, ) \prod_{i=1}^n \tilde{Z}_i \mathcal{N}(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2).$$

Suppose we are updating the parameters for the  $i^{\text{th}}$  term and the current marginal belief about  $f_i$  using our approximation  $q$  is

$$q(f_i) = \mathcal{N}(f_i, \mu_i, \sigma_i^2).$$

We first compute the parameters of the cavity distribution:

$$q_{-i}(f_i) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, ) \prod_{j \neq i} \tilde{Z}_j \mathcal{N}(f_j; \tilde{\mu}_j, \tilde{\sigma}_j^2) \propto \mathcal{N}(f_i, \mu_{-i}, \sigma_{-i}^2)$$

where

$$\mu_{-i} = \sigma_{-i}^2 \left( \frac{\mu_i}{\sigma_i^2} - \frac{\tilde{\mu}_i}{\tilde{\sigma}_i^2} \right) \text{ and } \sigma_{-i}^2 = \left( \frac{1}{\sigma_i^2} - \frac{1}{\tilde{\sigma}_i^2} \right)^{-1}.$$

Next, we compute the moments of the tilted distribution,

$$q_{-i}(f_i) \Phi(y_i f_i),$$

which are

$$\begin{aligned} \hat{Z}_i &= \Phi(z_i) \text{ where } z_i = \frac{y_i \mu_{-i}}{\sqrt{1 + \sigma_{-i}^2}} \\ \hat{\mu}_i &= \mu_{-i} + \frac{\mathcal{N}(z_i)}{\Phi(z_i)} s_i \text{ where } s_i = \frac{y_i \sigma_{-i}^2}{\sqrt{1 + \sigma_{-i}^2}} \\ \hat{\sigma}_i^2 &= \sigma_{-i}^2 - \frac{\mathcal{N}(z_i)}{\Phi(z_i)} s_i^2 \left( z_i + \frac{\mathcal{N}(z_i)}{\Phi(z_i)} \right). \end{aligned}$$

Note the similarity between these moments and the moments we computed during assumed density filtering. The derivation of these moments is almost identical to the derivation of the moments we matched in assumed density filtering, they only differ in the parameters of the Gaussian being multiplied by the standard normal CDF; for brevity, the derivations have been omitted here.

Lastly, we update the site parameters such that the moments of

$$q_{\text{new}}(f_i) = \tilde{Z}_i q_{-i}(f_i) \mathcal{N}(f_i; \tilde{\mu}_i, \tilde{\sigma}_i^2)$$

to match the moments above. To do this, we set the site parameters to be

$$\begin{aligned}\tilde{\mu}_i &= \tilde{\sigma}_i^2 \left( \frac{\hat{\mu}_i}{\hat{\sigma}_i^2} - \frac{\mu_{-i}}{\sigma_{-i}^2} \right) \\ \tilde{\sigma}_i^2 &= \left( \frac{1}{\hat{\sigma}_i^2} - \frac{1}{\sigma_{-i}^2} \right)^{-1} \\ \tilde{Z}_i &= \hat{Z}_i \sqrt{2\pi (\sigma_{-i}^2 + \tilde{\sigma}_i^2)} \exp \left( -\frac{(\mu_{-i} - \tilde{\mu}_i)^2}{2 (\sigma_{-i}^2 + \tilde{\sigma}_i^2)} \right).\end{aligned}$$

We then recompute the approximation  $q(\mathbf{f})$  and resample another term to update our belief about.

#### EP for truncation

Another common use case for expectation propagation is to approximately truncate a Gaussian random variable at some threshold. For a univariate Gaussian,  $x$ , we can represent this truncation using a single (non-Gaussian) “likelihood” term:

$$t(x) = \mathbb{1}(x < a)$$

where  $a$  is the threshold and  $\mathbb{1}$  is the indicator function, which takes value 1 when the argument is true and 0 otherwise.

Again, suppose our initial approximation of the truncation term is  $\tilde{t}(x) = \mathcal{N}(x; \tilde{\mu}, \tilde{\sigma}^2)$ , giving rise to the approximate belief  $q(x) = \mathcal{N}(x; \mu, \sigma^2)$ . From here, we can calculate the mean and variance of the cavity distribution as before:

$$\mu_- = \sigma_-^2 \left( \frac{\mu}{\sigma^2} - \frac{\tilde{\mu}}{\tilde{\sigma}^2} \right) \text{ and } \sigma_-^2 = \left( \frac{1}{\sigma^2} - \frac{1}{\tilde{\sigma}^2} \right)^{-1}.$$

Omitting the tedious derivations, the resulting updates to the site parameters are

$$\begin{aligned}\tilde{\mu} &= \mu_- + \sigma_- \left( \frac{\phi(z)}{\Phi(z)} + z \right)^{-1} \text{ where } z = \frac{a - \mu_-}{\sigma_-} \\ \tilde{\sigma}^2 &= \frac{\Phi(z)}{\phi(z)} \left( \frac{\phi(z)}{\Phi(z)} + z \right)^{-1} - \sigma_-^2 \\ \tilde{Z} &= \Phi(z) \sqrt{2\pi (\tilde{\sigma}^2 + \sigma_-^2)} \exp \left( -\frac{1}{2 \left( 1 + \frac{\Phi(z)}{\phi(z)} z \right)} \right).\end{aligned}$$

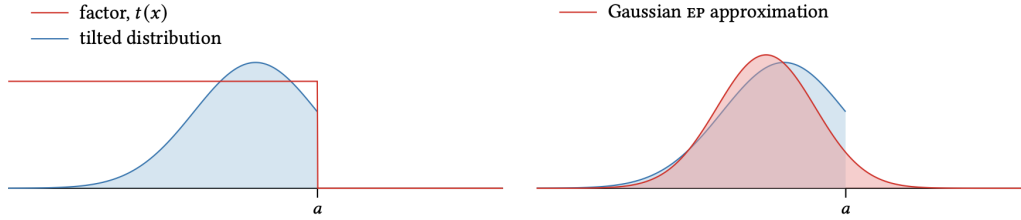


Figure 2: An EP approximation to a truncated Gaussian.

Figure 2 shows the effect of truncating a 1-dimensional Gaussian to be less than a threshold  $a$ . As an approximation, there is still some mass above the threshold but it accounts for only roughly 5% of the total probability mass.

### Theoretical motivation

We conclude with one brief note about the theoretical motivation behind the moment matching used in assumed density filtering and expectation propagation. The Kullback–Leibler (KL) divergence (also called *relative entropy*) is a notion of “distance” between probability distributions, defined by

$$d_{\text{KL}}(p(\boldsymbol{\theta}) \parallel q(\boldsymbol{\theta})) = \int p(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta})}{q(\boldsymbol{\theta})} d\boldsymbol{\theta}.$$

Notice that KL divergence is not a true distance, as it is not symmetric, but it does satisfy  $d_{\text{KL}}(p \parallel q) \geq 0$  with  $d_{\text{KL}}(p \parallel q) = 0$  if and only if  $p(\boldsymbol{\theta}) = q(\boldsymbol{\theta})$  almost everywhere.

A well-known result is that the KL divergence between an arbitrary probability distribution  $p(\boldsymbol{\theta})$  and a multivariate Gaussian distribution  $q(\boldsymbol{\theta})$  (in the direction  $d_{\text{KL}}(p \parallel q)$ ) is minimized when  $q$  is chosen to match the moments of  $p$ . Therefore, in the case of Gaussian approximations, these methods can be seen as iteratively building up an approximate posterior in the desired family by minimizing the KL divergence at every step.

Minimizing KL divergence in “the other direction,”  $d_{\text{KL}}(q \parallel p)$ , gives rise to another family of approximation techniques known as *variational Bayesian inference*, which we will learn more about later in the semester.



## Covariance Functions

For the most part, our discussion of covariance functions to date has been limited to the commonly-used squared exponential kernel,

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\ell^2}\right).$$

kernel variance  $\lambda^2$  length scale  $\ell$

While this covariance function is powerful, it makes some rather strong assumptions about the function being modeled: formally, the squared exponential function is appropriate for modeling functions that are infinitely differentiable. This can most easily be seen through drawing samples from a GP with a squared exponential kernel: Because of the (frequently unrealistic) assumption

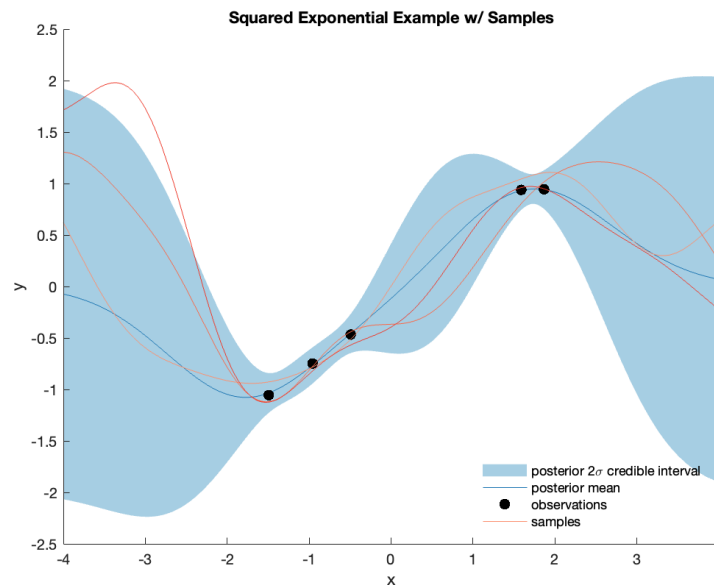


Figure 1: An example of a (zero-mean) Gaussian process with a squared exponential kernel fit to 5 observations of some unknown function as well as three sample paths drawn from the posterior Gaussian posterior. The kernel hyperparameters are set to be  $\ell = \lambda = 1$ .

of such a high degree of smoothness, many experts have argued against the use of the squared exponential kernel. Here, we will consider a few alternatives and discuss their properties.

### The Matérn Covariance Function

The Matérn class of covariance functions is a family of kernels parametrized by a value  $\nu$ , which controls how smooth functions modeled by the corresponding covariance function are: formally, functions modeled by a Matérn kernel are  $k$ -times differentiable if and only if  $k < \nu$ .

The general functional form is somewhat complex and rarely seen in practice. Instead, a few members of this class corresponding to specific choices of the parameter  $\nu$  are commonly used:

- When  $\nu = \frac{1}{2}$ ,

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\ell}\right)$$

- When  $\nu = \frac{3}{2}$ ,

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \left(1 + \sqrt{3} \frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\ell}\right) \exp\left(-\sqrt{3} \frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\ell}\right)$$

- When  $\nu = \frac{5}{2}$ ,

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \left(1 + \sqrt{5} \frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\ell} + 5 \frac{\|\mathbf{x} - \mathbf{x}'\|_1^2}{3\ell^2}\right) \exp\left(-\sqrt{5} \frac{\|\mathbf{x} - \mathbf{x}'\|_1}{\ell}\right)$$

Figure 2 shows examples of a GP prior with these covariance functions conditioned on the same set of observations; note how the samples from these GPs start out quite jagged but become smoother as  $\nu$  increases. Indeed, for larger values of  $\nu$ , the modeled functions becomes so smooth as to be basically indistinguishable from those modeled by a squared exponential kernel. In fact, as  $\nu \rightarrow \infty$  approaches infinity, the corresponding Matérn covariance function precisely approaches the squared exponential kernel.

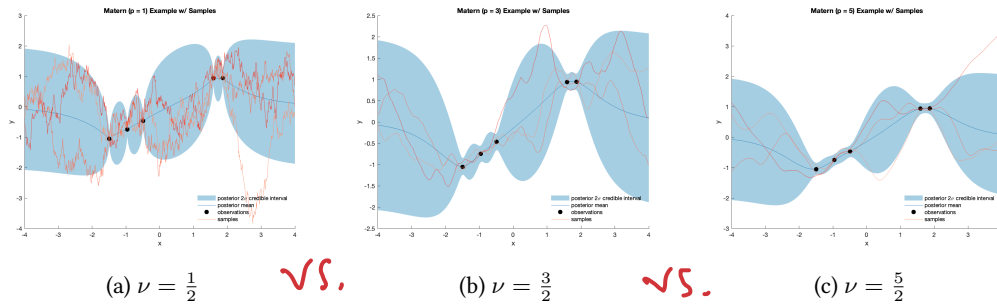


Figure 2: Gaussian process posteriors with Matérn covariance functions conditioned on the same set of 5 observations as well as three sample paths from each posterior. Each Matérn kernel uses  $\ell = \lambda = 1$  and a different value of the hyperparameter  $\nu$ .

### The Rational Quadratic Covariance Function

Another extension or variant of the squared exponential kernel is the rational quadratic kernel, which can be seen as a scale mixture of squared exponential kernels (i.e., a weighted sum of infinitely many squared exponentials), each with a different length scale. Intuitively, this kernel can be used to model smooth (i.e., infinitely differentiable) functions that vary across a variety of length scales i.e., functions that can change both very quickly or very slowly but always smoothly.

Formally, the rational quadratic kernel is given by

$$k(x, x') = \lambda^2 \left(1 + \frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2c\ell}\right)^{-c}$$

where  $c$  is a hyperparameter that controls the relative preference for length scales closer to 0 vs. length scales closer to  $\ell$ : smaller values of  $c$  result in a bias towards shorter length scales whereas larger values of  $c$  result in a bias towards longer length scales.

Figure 3 shows examples of a GP with a rational quadratic covariance function conditioned on the same dataset with two different values of  $c$ ; note how the samples from the GP with  $c = 2$  tend to vary over longer length scales than the samples from the GP with  $c = 0.2$ .

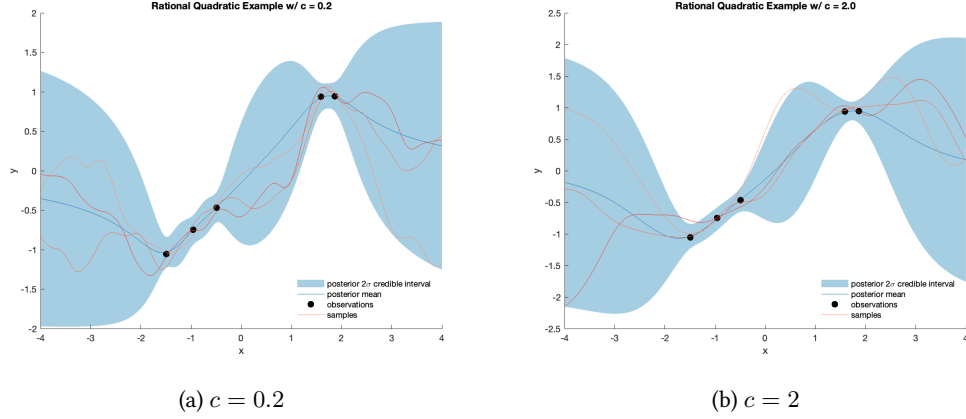


Figure 3: Two Gaussian processes fit to the same set of 5 observations using the rational quadratic kernel. The kernel hyperparameters are again set to be  $\ell = \lambda = 1$  and 3 samples from each posterior are shown.

### Automatic Relevance Determination

Suppose we have 2-dimensional inputs,  $\mathbf{x} = [x_1 \ x_2]^T$ . In this setting, we can write out the norm in the squared exponential kernel as

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp \left( -\frac{(x_1 - x'_1)^2}{2\ell^2} - \frac{(x_2 - x'_2)^2}{2\ell^2} \right)$$

Written out this way, one possible interpretation is that the dimensions  $x_1$  and  $x_2$  are constrained to have the same length scale,  $\ell$ . While this may be appropriate in some settings, it may not always be the case that the modeled function behaves the same in both dimensions.

A natural extension in such settings is the squared exponential kernel with *automatic relevance determination*: this modification effectively gives each dimension its own length scale hyperparameter, allowing the GP belief to change at different rates along each dimension. Again, in two dimensions, this would look like:

$$k(\mathbf{x}, \mathbf{x}') = \lambda^2 \exp \left( -\frac{(x_1 - x'_1)^2}{2\ell_1^2} - \frac{(x_2 - x'_2)^2}{2\ell_2^2} \right)$$

Figure 4 shows an example of a 2-dimensional posterior GP mean with a squared exponential covariance function using automatic relevance determination; note how the mean changes rapidly as  $x_1$  (the horizontal axis) changes and does not change very much as  $x_2$  (the vertical axis) changes.

Roughly speaking, one can conceptualize the length scale as the amount of distance required between inputs in the corresponding distance in order for the belief about the function at those inputs to be uncorrelated. Therefore, one interpretation of a long length scale (relative to the input domain) is that the posterior mean is almost independent of the corresponding feature or that the feature is “irrelevant”.

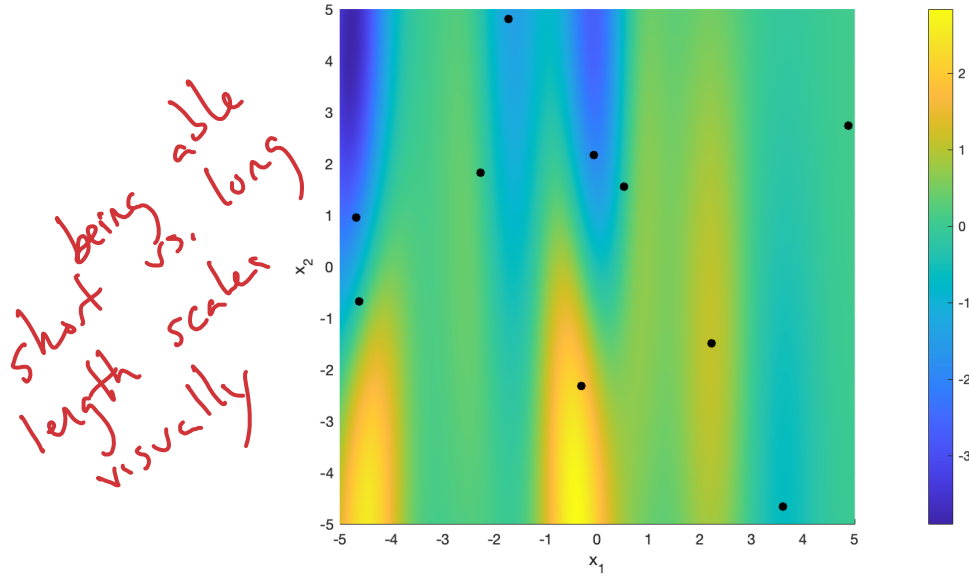


Figure 4: An example of a Gaussian process posterior mean using a squared exponential kernel with automatic relevance determination: the length scale in the  $x_1$  dimension is  $\ell_1 = 0.5$  while the length scale in the  $x_2$  dimension is  $\ell_2 = 5$ . The black dots are the data points used to condition the posterior belief.

Thus, one possible use for kernels with automatic relevance determination is to determine which features are relevant for a given task (hence the name) or equivalently, which features can be removed from the analysis. This can be done by first setting the kernel hyperparameters using something like maximum likelihood estimation and then identifying which length scales are relatively long and relatively short.

#### Covariance Functions for Discrete Data

Given discrete or categorical input features, one straightforward approach to building a kernel is to simply define the function for all possible pairs of input values e.g., if  $x$  can take on values  $\{A, B, C\}$ , then one way of describing a kernel over this variable is through a table:

$$k(x, x') = \begin{array}{c|ccc} x \backslash x' & A & B & C \\ \hline A & k_{AA} & k_{AB} & k_{AC} \\ \hline B & k_{AB} & k_{BB} & k_{BC} \\ \hline C & k_{AC} & k_{BC} & k_{CC} \end{array}$$

Unfortunately, this kernel has  $O(n^2)$  hyperparameters which must be specified where  $n$  is the number of possible values  $x$  can take on. A simpler form for a kernel over a discrete variable is:

$$k(x, x') = \begin{cases} s^2 & \text{if } x = x' \\ cs^2 & \text{otherwise} \end{cases}$$

for  $c < 1$  (to ensure that the resulting Gram matrices are positive semi-definite). Essentially, this kernel defines all non-identical pairs of values as equivalently dissimilar, significantly reducing the number of hyperparameters that need to be specified.

For multiple discrete features, the above protocol can be extended so that a kernel is defined over all possible pairs of combinations of features (for those of you so inclined, you can think of this as defining a kernel on the tensor product of the feature spaces). Of course, this approach will scale very poorly: if both  $x_1$  and  $x_2$  can take on one of three values a kernel  $K(\mathbf{x} = [x_1, x_2], \mathbf{x}' = [x'_1, x'_2])$  defined this way will have  $(9 + 1)9/2 = 45$  hyperparameters! This can be reduced using an approach like the one above, where the kernel treats all different values in each dimension  $x_i$  as equivalently dissimilar but that will also scale poorly as the number of dimensions (i.e., the number of discrete features) increases.

An alternative approach is to define a kernel for each discrete feature and then combine them (see below); the drawback here is that the resulting kernel will effectively ignore all interaction effects between the discrete features.

### Combining Covariance Functions

What can be done if our data is mixed i.e., the input has both continuous and discrete features? Well it turns out that kernels can be combined in a few convenient ways:

Formally, if  $k_1$  and  $k_2$  are valid kernels then:

- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$
- $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$

are also both valid kernels.

Returning to the scenario of mixed data, if the input vector  $x$  can be decomposed into  $\mathbf{x} = [\mathbf{x}_c, \mathbf{x}_d]^T$  where  $\mathbf{x}_c$  are all the continuous features and  $\mathbf{x}_d$  are all the discrete features, then one possible kernel for this data is

$$k(\mathbf{x}, \mathbf{x}') = k_c(\mathbf{x}_c, \mathbf{x}'_c) + k_d(\mathbf{x}_d, \mathbf{x}'_d)$$

where  $k_c$  is some arbitrary kernel defined over continuous inputs and  $k_d$  is some arbitrary kernel defined over discrete inputs.

The ability to combine kernels can also be useful even when the domain is not mixed. Consider the dataset shown in Figure 5.

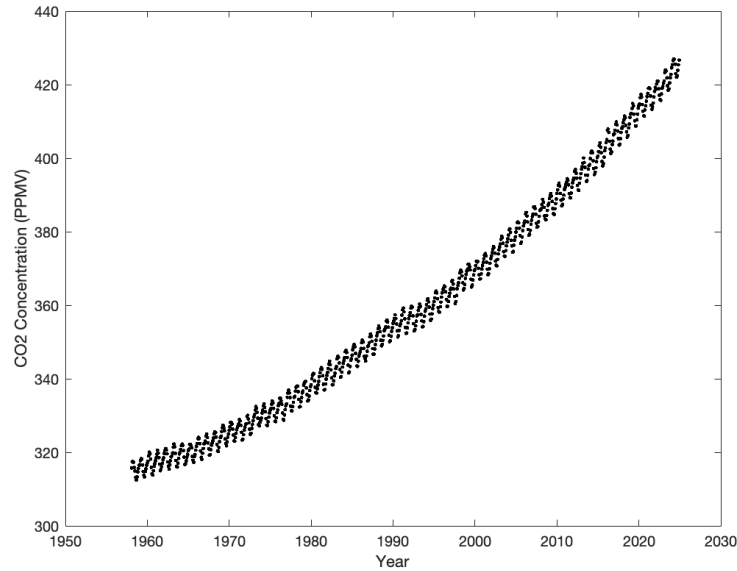


Figure 5: A sample dataset where the underlying function to be predicted displays multiple disparate trends/behaviors. This type of function is not particularly well modeled by any single covariance function and might benefit from a combination of different kernels.

This data set plots the monthly average atmospheric CO<sub>2</sub> concentrations in parts per million by volume (PPMV) at the Mauna Loa observatory in Hawaii (data found here: [https://scrippsco2.ucsd.edu/data/atmospheric\\_co2/mlo.html](https://scrippsco2.ucsd.edu/data/atmospheric_co2/mlo.html)).

Looking at the data, there are multiple trends:

- a long-term, smoothly increasing trend,
- a seasonal, periodic or repeating component,
- medium-term irregularities and,
- short-term irregularities or noise.

In [GPML Section 5.4.3](#), Rasumussen and Williams suggest modeling each component with one or two kernel functions and then combining them into a single kernel. Specifically, they use the following kernels:

- a squared exponential kernel:  $k_1(x, x') = \theta_1^2 \exp\left(-\frac{(x-x')^2}{2\theta_2^2}\right)$ ,
- a squared exponential kernel multiplied by a *periodic* kernel:  

$$k_2(x, x') = \theta_3^2 \exp\left(-\frac{(x-x')^2}{2\theta_4^2} - \frac{2 \sin(\pi(x-x'))}{\theta_5^2}\right),$$

- a rational quadratic kernel:  $k_3(x, x') = \theta_6^2 \left(1 + \frac{(x-x')^2}{2\theta_8\theta_7^2}\right)^{\theta_8}$ , and
- another squared exponential kernel:  $k_4(x, x') = \theta_9^2 \exp\left(-\frac{(x-x')^2}{2\theta_{10}^2}\right)$ .

The final kernel is simply  $k(x, x') = k_1(x, x') + k_2(x, x') + k_3(x, x') + k_4(x, x')$ ; the hyperparameters  $\{\theta_1, \theta_2, \dots, \theta_{10}\}$  are fit to maximize the likelihood of the data between 1958 and 2003.

Figure 6 shows the posterior belief over the years 2003 to 2025 when fit on the data from 1958 to 2003. Note how the posterior has captured the seasonal trend but has underestimated the longer-term increases.

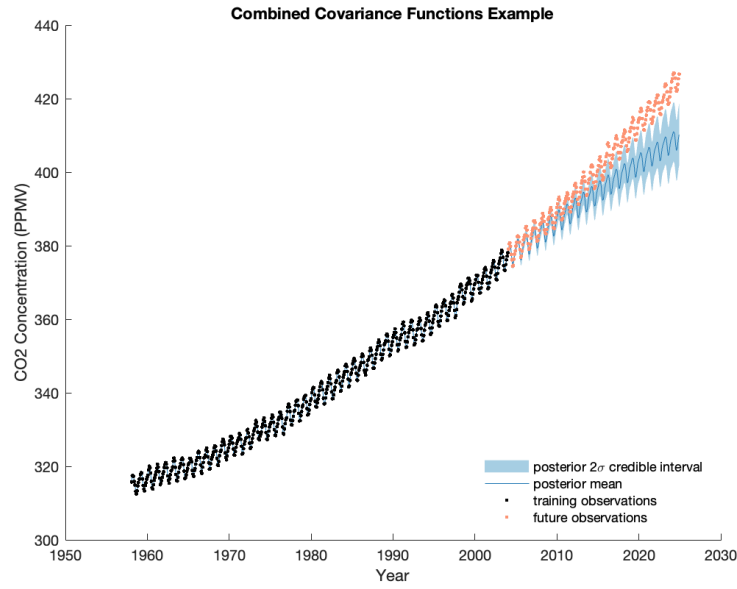


Figure 6: A sample dataset where the underlying function to be predicted displays multiple disparate trends/behaviors. This type of function is not particularly well modeled by any single covariance function and might benefit from a combination of different kernels.

### Covariance Functions for Non-vector Data

So far, we have only considered GPs over vector data i.e., data that can be represented as  $x = [x_1, \dots, x_d]^T$ . However, there are many interesting settings where the inputs are not vector inputs. GPs (and most other kernel methods for that matter) are convenient in these settings because they can be applied in a relatively straightforward way to non-vector inputs: all that's need is a kernel defined over the inputs i.e., some way of describing how similar or dissimilar any pair of inputs are. Two common and useful examples of non-vector inputs are strings and graphs.

#### String Kernels

A common class of string kernels involves looking at the frequency of substrings between two input strings; intuitively, the two input strings  $S$  and  $S'$  are similar (and therefore, should be more related under a GP model or have a higher covariance) if there are many substrings that frequently occur in both  $S$  and  $S'$ .

Let a string  $S$  be a collection of characters from some finite alphabet  $\mathcal{A}$  and define  $N_\sigma(S)$  as the number of times some substring  $\sigma$  occurs in the  $S$ ;  $\sigma$  is a substring of  $S$  if  $S = \rho\sigma\tau$  for some possibly empty strings  $\rho$  and  $\tau$  where  $\rho\sigma\tau$  denotes the concatenation of the strings  $\rho$ ,  $\sigma$  and  $\tau$ . Also let  $|S|$  denote the length of the string  $S$ . Then the above kernel can be expressed as:

$$k(S, S') = \sum_{\sigma \in \mathcal{A}^*} w_\sigma (N_\sigma(S) * N_\sigma(S'))$$

where the notation  $\sigma \in \mathcal{A}^*$  indicates that the sum is over all possible strings using characters in  $\mathcal{A}$  (the  $*$  is meant to indicate the regular expression quantifier). While this is technically an infinite sum, observe that if  $|\sigma| > \max(|S|, |S'|)$ , the term inside the sum will always be 0.

The hyperparameters for this kernel are the weights  $w_s$ ; note that in this most general form, there are exponentially many hyperparameters that need to be specified! Correspondingly, computing this kernel naïvely would require searching each input for exponentially many substrings. Fortunately, there are convenient algorithmic tricks for computing this kernel given certain settings of the weights (for those of you who are interested in such things, these algorithms make clever use of suffix trees to compute string kernels in  $O(|S| + |S'|)$ ).

A few common ways of constraining these weights are

- The bag-of-characters kernel:  $w_\sigma = 0$  if  $|\sigma| > 1$
- The k-spectrum kernel:  $w_\sigma = 0$  if  $|\sigma| \neq k$
- The bag-of-words kernel:  $w_s = 0$  if  $\sigma \notin$  some corpus or dictionary
- The weights in a bag-of-words kernel can be set using some importance metric over words e.g., term frequency inverse document frequency or TF-IDF.



## Graph Kernels

Graphs are a mathematical abstraction defined over a set of nodes or vertices where some pairs of vertices are connected by an edge. Graphs are used in a variety of machine learning tasks e.g., to represent physical/social networks or chemical structures.

Unfortunately, graphs are notoriously computationally complex and many straightforward approaches or definitions of similarity between graphs are known (or assumed) to be NP-complete e.g., the binary isomorphism kernel where  $k(G, G')$  is 1 if the graphs are topographically identical and 0 otherwise.


A few common types of graph kernels are

- Walk-based kernels: given labels for the vertices in each graph, a kernel can be defined by performing a bunch of random walks in both graphs and comparing the number of walks (or sequences of vertices) that occurred in both graphs.
- Subgraph-based kernels: these kernels compare the counts of all types of subgraphs (usually of a small, fixed size e.g., all subgraphs of 3 or 4 vertices) in the input graphs.
- Neighborhood-based kernels: again, given vertex labels, a kernel can be defined by comparing how similar each vertex's neighborhood (i.e., the set of vertices adjacent to a given vertex) is across the two graphs.

Most graph kernels are very complex to formally define and compute so are beyond the scope of this course. However, if you are interested in learning more about this topic, I encourage you to look over the (optional) survey paper found in the readings/resources of this lecture.

## Bayesian Experimental Design

Many problems that arise in scientific discovery and design share the following properties:

- 
- the *design space* or the space of viable candidates is massive,
  - the percentage of relevant or desirable observations is low, and
  - the act of discovery or exploring the space is inherently slow.

As a motivating example, consider the problem of crop breeding: agricultural scientists are interested in predicting (and often, maximizing) the yield of some crop but there are infinitely many possible genetic varieties or strains, most are not viable or will not outperform current commercial strains and crucially, gathering the relevant yield data requires a full growth cycle i.e., it takes months or even years before it can be determined whether or not some strain was a success. Similar problems arise in drug and material discovery, robotics and reinforcement learning settings, and many engineering disciplines.

The Bayesian approach to solving such **experimental design** problems has been found to be very powerful in terms of accelerating and improving the discovery process. One particularly notable success story is “AutoML” or automated **hyperparameter optimization**, which also fits neatly into this paradigm: the search space of possible hyperparameter settings is massive, especially as the number of hyperparameters in the models grows. Yet most hyperparameter settings perform quite poorly and the only way to evaluate some setting of the hyperparameters requires training (or partially training) the model, which could take days or weeks on expensive, specialized hardware. Famously, the hyperparameters of AlphaGo, the Go playing agent that beat top human experts, were tuned in part using a technique known as Bayesian optimization.<sup>1</sup>

At a high-level, the Bayesian approach to experimental design combines a variety of tools we have already developed in this course:

- We assume the existence of some unknown function that maps elements of the design space to some performance metric or property of interest. This function is typically, but not necessarily, modeled with a Gaussian process.
- We query some oracle (e.g., by running an experiment or gathering data) to refine/improve our model. The location of this query is determined by specifying an objective (or equivalently, a loss) and applying Bayesian decision theory.
- We update our model using Bayesian inference and repeat until some notion of convergence has been achieved or an allocated budget of queries has been expended.

Figure 1 depicts an outline of an experimental design pipeline.

The exact nature of the loss function used to define the query selection will depend on our inference goal. We will explore a variety of inference tasks in the following weeks that broadly fall under the umbrella of **probabilistic numerics**. Probabilistic numerics treats intractable or unknowable quantities as random variables to be estimated, commonly using Bayesian inference techniques. The next few lectures will focus on one such numerical analysis task: optimization.

---

<sup>1</sup><https://arxiv.org/pdf/1812.06855>

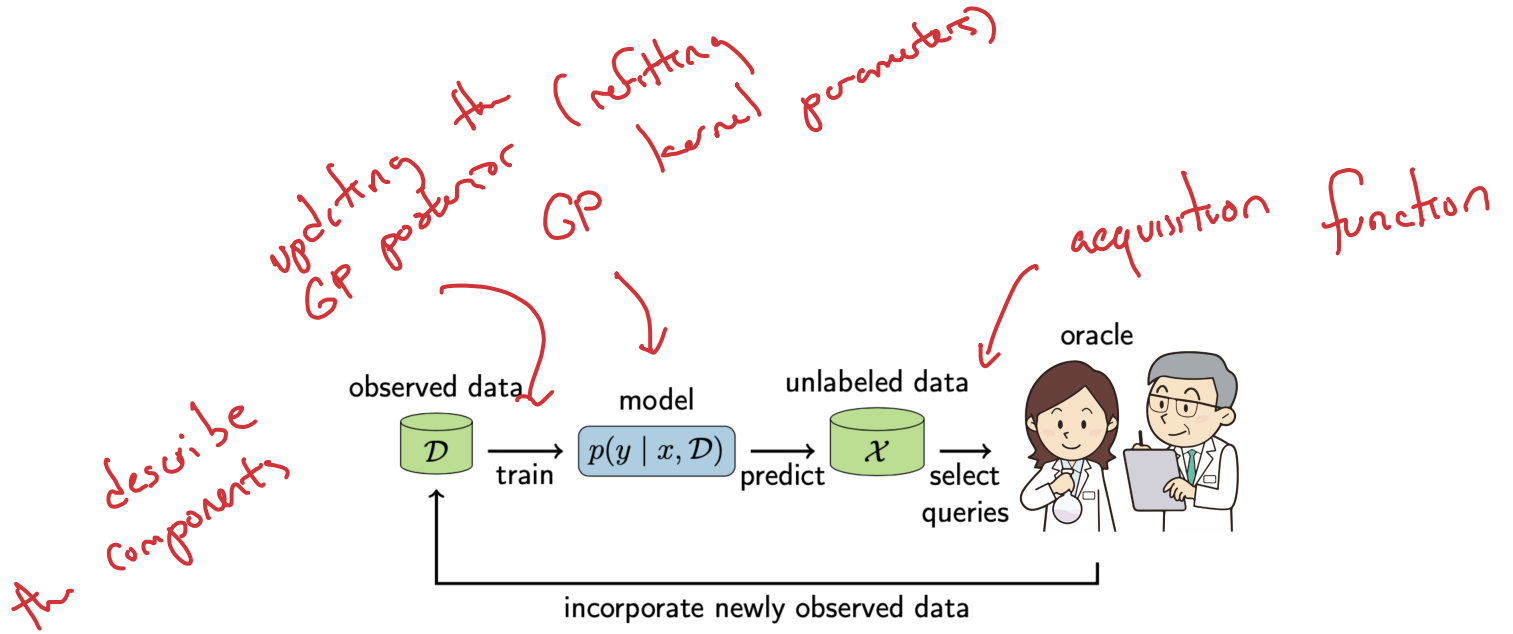


Figure 1: A high-level overview of the Bayesian approach to experimental design. Note the iterative, sequential nature of the problem: new observations are fed back into the model, which is subsequently used to gather more observations.

## Bayesian Optimization

Suppose we have a function  $f: \mathcal{X} \rightarrow \mathbb{R}$  that we wish to maximize on some domain  $X \subseteq \mathcal{X}$ . That is, we wish to find

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

In numerical analysis, this problem is typically called (global) **optimization** and has been the subject of decades of study. We draw a distinction between global optimization, where we seek the absolute optimum in  $X$ , and local optimization, where we seek to find a local optimum in the neighborhood of a given initial point  $x_0$ .

If an exact functional form for  $f$  is not available (that is,  $f$  behaves as a “black box”), what can we do? **Bayesian optimization** proceeds by maintaining a probabilistic belief about  $f$  and designing a so-called **acquisition function** to determine where to evaluate the function next. Bayesian optimization is particularly well-suited to global optimization problems where  $f$  is an *expensive* black-box function; for example, evaluating  $f$  might require running an expensive simulation or training a large, machine learning model.

Although not strictly required, Bayesian optimization almost always reasons about  $f$  by choosing an appropriate Gaussian process prior:

$$p(f) = \mathcal{GP}(f; \mu, K).$$

Given observations  $\mathcal{D} = (\mathbf{X}, \mathbf{f})$ ,<sup>2</sup> we can condition our distribution on  $\mathcal{D}$  as usual:

$$p(f | \mathcal{D}) = \mathcal{GP}(f; \mu_{f|\mathcal{D}}, K_{f|\mathcal{D}}).$$

We have already developed the machinery to perform this modeling and updating. The key question becomes, given this set of observations, how do we select where to observe the function next?

The meta-approach in Bayesian optimization is to design an **acquisition function**,  $a(x)$ . The acquisition function is typically an inexpensive function that can be evaluated at a given point that is commensurate with how desirable evaluating  $f$  at  $x$  is expected to be for the maximization

<sup>2</sup>We will assume these observations to be noiseless here, but we could extend the methods here to the noisy case without difficulty.

problem. We then optimize the acquisition function to select the location of the next observation. Of course, we have merely replaced our original optimization problem with another optimization problem, but on a much-cheaper, easier to optimize function  $a(x)$ .

### Exploration vs. Exploitation

A key consideration in designing an acquisition function for Bayesian optimization is navigating the tradeoff between exploration and exploitation. Exploration is when we use our queries to learn more about the function at places where we are uncertain about its behavior in order to set up potential future gains. Exploitation is when we use our queries to for short-term gain by improving our belief around locations we currently believe to be high yield. This tradeoff is illustrated in Figure 2 for the Bayesian optimization context, where “high yield” is defined in terms having a higher target function value.

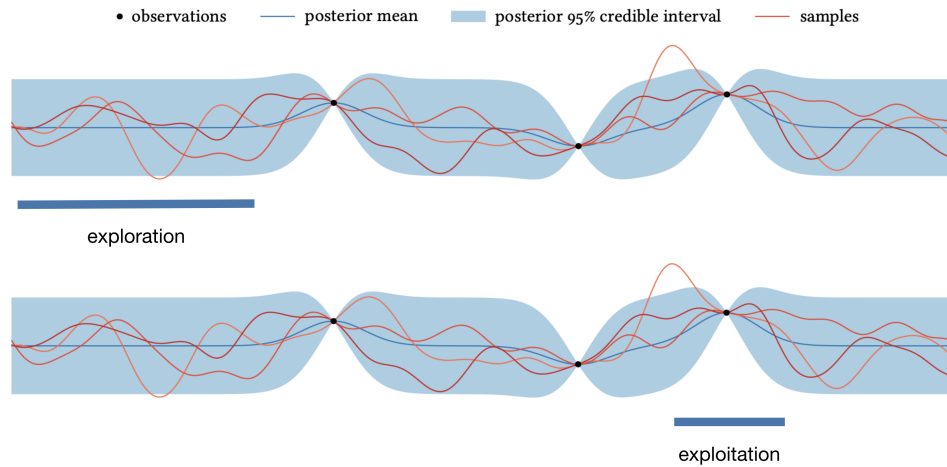


Figure 2: A depiction of the exploration-exploitation tradeoff in the context of Bayesian optimization: exploration (top) prioritizes exploring regions far from the observations, where the behavior of the function is most unknown, while exploitation (bottom) would query locations near where the model *currently* believes the highest function value to be.

We will see that many acquisition functions for Bayesian optimization naturally and elegantly navigate this trade-off, seamlessly and automatically transitioning from an exploration phase to an exploitation phase and back as needed.

### Acquisition Functions

Many acquisition functions can be interpreted in the framework of Bayesian decision theory as corresponding to an expected loss associated with evaluating  $f$  at a point  $x$ . We then select the point with the lowest expected loss, as usual.

In the following sections, we will drop the  $f \mid \mathcal{D}$  subscripts on the mean  $\mu$  and covariance  $K$  functions for  $f$ ; assume everything is based on a posterior distribution when data is available.

### Probability of improvement

Perhaps the first acquisition function designed for Bayesian optimization was **probability of improvement**. Suppose

$$f' = \max \mathbf{f}$$

is the maximal value of  $f$  observed so far. Probability of improvement evaluates  $f$  at the point most likely to improve upon this value. This corresponds to the following utility function<sup>3</sup> associated with evaluating  $f$  at a given point  $x$ :

$$u(x) = \begin{cases} 0 & f(x) < f' + \varepsilon \\ 1 & f(x) \geq f' + \varepsilon. \end{cases}$$

That is, we receive a unit reward if  $f(x)$  turns out to be “better than” (i.e., greater than)  $f'$  by at least  $\varepsilon$ , and no reward otherwise; here,  $\varepsilon$  is a tunable parameter of this acquisition function. The probability of improvement acquisition function is then the expected utility as a function of  $x$ :

$$\begin{aligned} a_{\text{PI}}(x) &= \mathbb{E}[u(x) \mid x, \mathcal{D}] = \int_{f' + \varepsilon}^{\infty} \mathcal{N}(f; \mu(x), K(x, x)) \, df \\ &= 1 - \Phi(f' + \varepsilon; \mu(x), K(x, x)). \end{aligned}$$

The point with the highest probability of improvement (the maximal expected utility) is selected. This is the Bayes optimal action under this loss. Figure 3 shows the effect of the parameter  $\varepsilon$  on this acquisition function: observe that as  $\varepsilon$  grows, the acquisition policy becomes more exploratory.

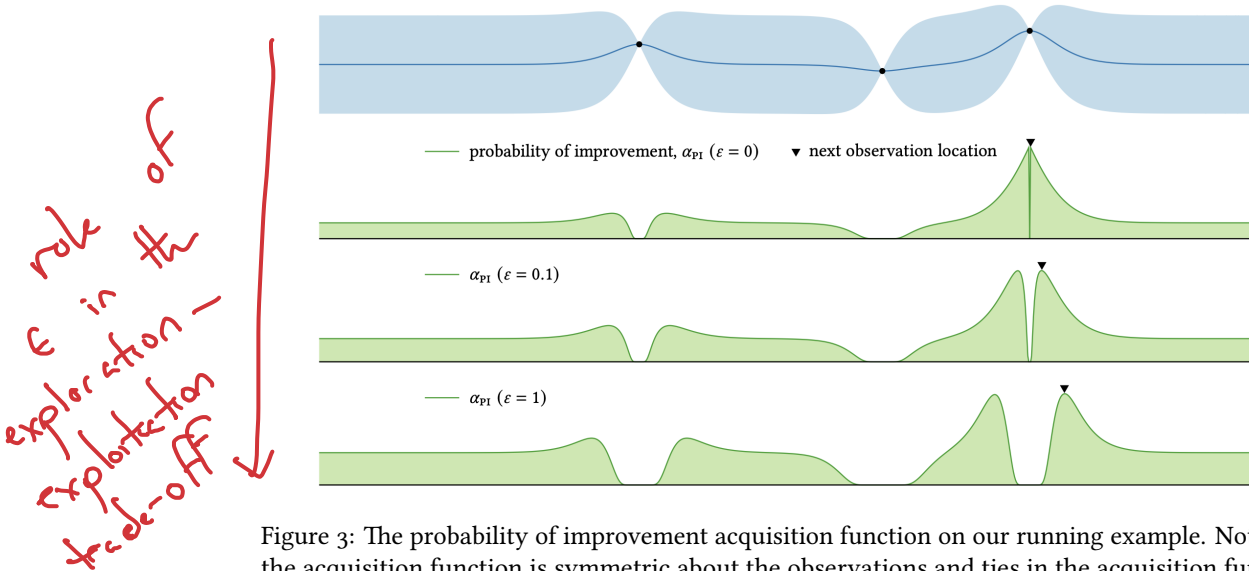


Figure 3: The probability of improvement acquisition function on our running example. Note that the acquisition function is symmetric about the observations and ties in the acquisition function are broken arbitrarily.

<sup>3</sup>Recall a utility function is simply a negative loss function.

compare and contrast w/ probability of improvement

### Expected improvement

The loss function associated with probability of improvement is somewhat odd: we get a reward for improving by at least a certain amount upon the current maximum, but that reward is independent of the size of the improvement! This acquisition function can sometimes lead to odd behavior, and in practice can get stuck in local optima and under-explore globally.

An alternative acquisition function that does account for the size of the improvement is **expected improvement**. Again suppose that  $f'$  is the maximal value of  $f$  observed so far. Expected improvement evaluates  $f$  at the point that, in expectation, improves upon  $f'$  the most. This corresponds to the following utility function:

$$u(x) = \max(0, f(x) - f').$$

That is, we receive a reward equal to the “improvement” of the observation over our current best observation,  $f(x) - f'$ . The expected improvement acquisition function is then the expected utility as a function of  $x$ :

$$\begin{aligned} a_{\text{EI}}(x) &= \mathbb{E}[u(x) \mid x, \mathcal{D}] = \int_{f'}^{\infty} (f - f') \mathcal{N}(f; \mu(x), K(x, x)) \, df \\ &= \underbrace{(\mu(x) - f') \Phi\left(\frac{\mu(x) - f'}{\sqrt{K(x, x)}}\right)}_{\text{exploitation}} + \underbrace{K(x, x) \mathcal{N}\left(\frac{\mu(x) - f'}{\sqrt{K(x, x)}}\right)}_{\text{exploration}}. \end{aligned}$$

The point with the highest expected improvement (the maximal expected utility) is selected.

The expected improvement has two components. The first can be increased by increasing the mean function  $\mu(x)$ . The second can be increased by increasing the variance  $K(x, x)$ . These two terms can be interpreted as explicitly encoding the tradeoff between *exploitation* (evaluating at points with high mean) and *exploration* (evaluating at points with high uncertainty). The expected improvement acquisition function *automatically* captures both as a result of the Bayesian decision theoretic treatment.

Figure 4 depicts the expected improvement acquisition function on our running example as well as the observations made by the algorithm over 20 iterations; the optimal function value was located after 19 queries.

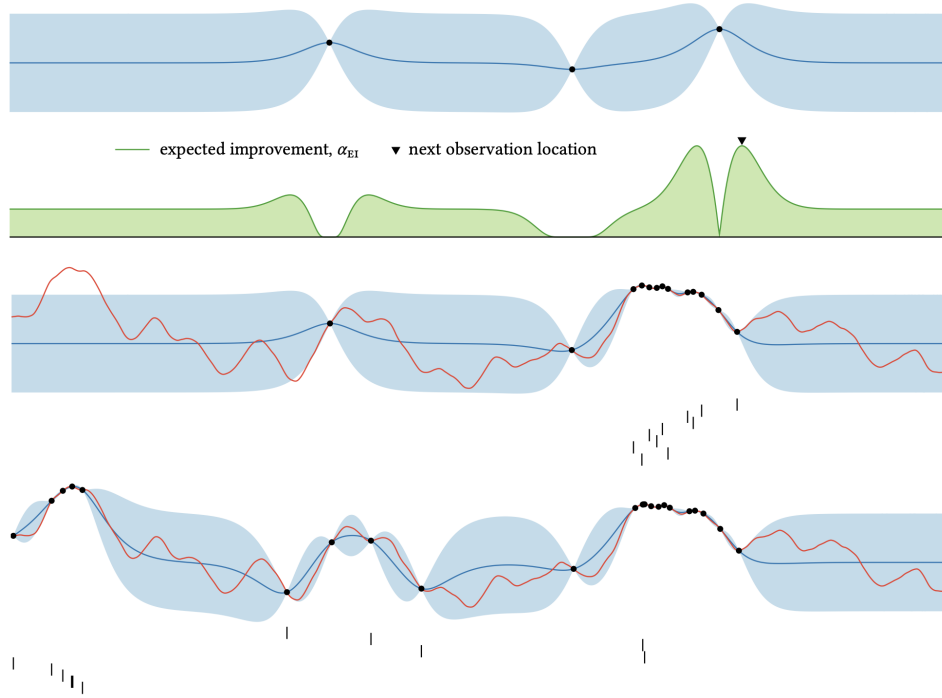


Figure 4: The expected improvement acquisition function evaluated on our running example (top) as well as the first ten observation locations selected by this policy (middle) and the next ten observation locations (bottom). The height of the tick marks indicates the relative order that points are selected to be observed in, with higher tick marks being selected earlier; the thicker tick marks indicate observations with 0.2 units of the true optimum.

## Acquisition Functions

Previously, we introduced two relatively simple acquisition functions for Bayesian optimization: Probability of Improvement (PI) and Expected Improvement (EI). These acquisition functions are commonly used in practice, in part because they permit closed-form, easy-to-optimize expressions when the underlying probabilistic model is a Gaussian process. Below, we will detail a few alternatives, some of which have provable theoretical guarantees, some which result in intractable expressions but have been shown empirically to perform well under various approximate inference techniques. Ultimately, the choice of acquisition function is simply a model hyperparameter that can be tuned to fit the specific task/goals of the practitioner.

For the following discussion, assume that we model the (black-box) objective function as a *noiseless* GP,  $f \mid \mathcal{D} \sim \mathcal{GP}(f; \mu_{\mathcal{D}}, k_{\mathcal{D}})$ , where  $\mathcal{D} = (\mathbf{X}, \mathbf{f})$  is a previously gathered set of observations of  $f$  and  $\mu_{\mathcal{D}}, k_{\mathcal{D}}$  are the posterior mean and covariance functions respectively.

### Knowledge Gradient

An alternative interpretation of EI is that it greedily maximizes the following utility function:

$$u'(\mathcal{D}) = \max_{\mathbf{x} \in \mathcal{D}} \mu_{\mathcal{D}}(\mathbf{x}).$$

*expected improvement*

In our noiseless setting,  $\mu_{\mathcal{D}}(\mathbf{x}) = f(\mathbf{x}) \forall \mathbf{x} \in \mathcal{D}$  so for a fixed dataset,  $u'(\mathcal{D}) = \max \mathbf{f} := f'$ .

Using this utility function, we can then recreate the point-wise utility we had previously defined as

$$u(\mathbf{x}) = u'(\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x}))) - u'(\mathcal{D}) = \max(0, f(\mathbf{x}) - f').$$

*derived as marginal gain*

Under this interpretation of EI, one might notice a bit of an oddity: why should our utility be defined over only points we have observed,  $\mathbf{x} \in \mathcal{D}$ ? Intuitively, this would be like saying that at the end of our budget, we are limited to returning a point in our observed dataset as our “guess” at the optimum of the function.

We can extend EI by instead allowing us to return any arbitrary location in the domain after our final guess, including ones that we never observed during experimentation. This gives rise to the **knowledge gradient** (KG) acquisition function. The implied utility function is

$$u(\mathbf{x}) = u''(\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x}))) - u''(\mathcal{D}) \text{ where } u''(\mathcal{D}) = \max_{\mathbf{x} \in \mathcal{X}} \mu_{\mathcal{D}}(\mathbf{x}).$$

*compare and contrast w/ EI*

Notably, the dataset utility,  $u''$ , is **defined over the entire domain,  $\mathbf{x} \in \mathcal{X}$** . Let  $\mu_{\mathcal{D}}^* = \max_{\mathbf{x} \in \mathcal{X}} \mu_{\mathcal{D}}(\mathbf{x})$ .

The resulting acquisition function is

$$\begin{aligned} a_{\text{KG}}(\mathbf{x}) &= \mathbb{E}[u(\mathbf{x}) \mid \mathbf{x}, \mathcal{D}] = \int \left( \max_{\mathbf{v} \in \mathcal{X}} \mu_{\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x}))}(\mathbf{v}) \right) p(f(\mathbf{x}) \mid \mathbf{x}, \mathcal{D}) df(\mathbf{x}) - \max_{\mathbf{v} \in \mathcal{X}} \mu_{\mathcal{D}}(\mathbf{v}) \\ &= \int \left( \max_{\mathbf{v} \in \mathcal{X}} \mu_{\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x}))}(\mathbf{v}) \right) \mathcal{N}(f(\mathbf{x}); \mu_{\mathcal{D}}(\mathbf{x}), k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})) df(\mathbf{x}) - \mu_{\mathcal{D}}^*. \end{aligned}$$

Intuitively, this integral averages all possible updates to the posterior mean that could occur if we were to observe the function at location  $\mathbf{x}$ . Unfortunately, this integral is **intractable** in all but a handful of degenerate cases (e.g., when the domain is discrete or one-dimensional and specific covariance function is used). As such, numerical integration techniques must be used to approximate



this integral; we will discuss many of these at length shortly but for the sake of example, a simple approximation would be the **Monte Carlo estimate**

$$a_{\text{KG}}(\mathbf{x}) = \int \left( \max_{\mathbf{v} \in \mathcal{X}} \mu_{\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x}))}(\mathbf{v}) \right) \mathcal{N}(f(\mathbf{x}); \mu_{\mathcal{D}}(\mathbf{x}), k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})) df(\mathbf{x}) - \mu_{\mathcal{D}}^*$$

$$\approx \left( \frac{1}{S} \sum_{s=1}^S \left( \max_{\mathbf{v} \in \mathcal{X}} \mu_{\mathcal{D} \cup (\mathbf{x}, f_s)}(\mathbf{v}) \right) \right) - \mu_{\mathcal{D}}^* \text{ where } f_s \sim \mathcal{N}(f(\mathbf{x}); \mu_{\mathcal{D}}(\mathbf{x}), k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})),$$

which simply draws (finitely many) samples from the posterior belief about  $f(\mathbf{x})$ , treats each sample as if it were the true observed function value and computes what the maximum of the posterior mean would be under each sample.

Empirically, it has been found that the knowledge gradient acquisition function outperforms EI, particularly in settings with noise (which admittedly, we have explicitly excluded from this discussion).

### Upper confidence bound

The Bayesian optimization literature draws heavily from the work done in multi-armed bandit settings, which can be viewed as a noisy, discrete-domain formulation of the Bayesian optimization problem. One relatively famous acquisition function inspired by this line of research is known as the UCB or **upper confidence bound** acquisition function. This acquisition function is somewhat difficult to define in terms of an implied utility function for Bayesian decision theory (although in certain cases, it is possible to construct one); instead, we will simply consider its functional form:

$$a_{\text{UCB}}(\mathbf{x}; \beta) = \underbrace{\mu_{\mathcal{D}}(\mathbf{x})}_{\text{exploitation}} + \beta \underbrace{\sqrt{k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})}}_{\text{exploration}}.$$

We refer to this (somewhat erroneously) as an upper confidence bound as it corresponds to an upper bound on the  $\alpha$ -credible interval of our posterior belief about the function's value at  $\mathbf{x}$ , where  $\alpha = \Phi(\beta)$ .

Similar to the  $\varepsilon$  parameter for PI,  $\beta > 0$  is a parameter that controls the exploration-exploitation tradeoff; like EI, that tradeoff can be explicitly seen in the functional form of  $a_{\text{UCB}}(\mathbf{x}; \beta)$ , where the  $\mu(x)$  term favors exploitation and the  $\sqrt{k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})}$  prioritizes exploration.

Figure 1 shows the effect of the tradeoff parameter (in terms of  $\alpha$  rather than  $\beta$ ): as  $\alpha$  increases, UCB tends to prefer exploration over exploitation and prioritizes observations farther from previously observed locations. We can see this in the sample run also depicted in Figure 1: relative to EI, this acquisition function tends to space out its observations more evenly over the domain (at least for this setting of  $\alpha$ ), even going so far as to depart from the neighborhood of the true optimal value after discovering it (the two tick marks circled in red in Figure 1).

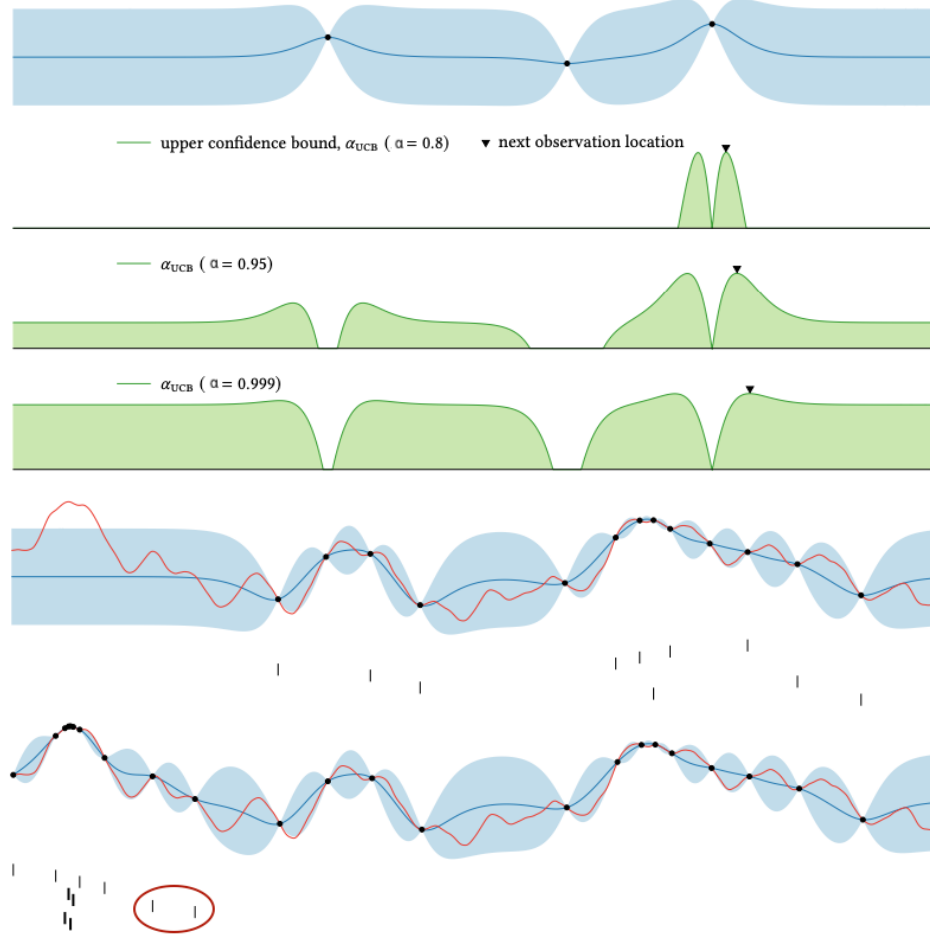


Figure 1: The upper confidence bound acquisition function evaluated on the running example from the previous notes (top), presented in terms of three different values of  $\alpha$ , as well as the first ten observation locations selected by this policy when  $\alpha = 0.999$  (middle) and the next ten observation locations (bottom). The height of the tick marks indicates the relative order that points are selected to be observed in, with higher tick marks being selected earlier; the thicker tick marks indicate observations with 0.2 units of the true optimum.

## Regret Bounds

One major advantage of this acquisition function is that it enjoys strong theoretical guarantees under certain assumptions about the quality of the GP’s fit to the underlying objective function. These theoretical guarantees are defined in terms of the *regret* of an acquisition function:

$$R_T = \sum_{t=1}^T f(\mathbf{x}^*) - f(\mathbf{x}_t)$$

where  $T$  is our budget of observations,  $f(\mathbf{x}^*)$  is the optimal objective function value, and  $\mathbf{x}_t$  is the  $t^{\text{th}}$  observation selected by the acquisition function.

Srinivas et al. (2010) proved that the regret of the UCB acquisition function is bounded according to

$$p\left(R_T \leq O^*\left(\sqrt{T\beta_T\gamma_T}\right)\right) \geq 1 - \delta$$

for some  $\delta \in (0, 1)$ , where  $O^*$  is equivalent to  $O$ -notation with polylogarithmic factors suppressed. The bound above holds if the following conditions are satisfied; the conditions are somewhat technical if rigorously defined so we provide an intuitive description of the conditions instead:

- the domain  $\mathcal{X}$  is compact and convex,
- the objective function is “well-modeled” by the GP belief,
- $\beta_t$  follows a schedule that increases as  $t$  increases and/or  $\delta$  decreases,
- $\gamma_t$  is some measure of how “informative” point-wise observations of the function are about the function as a whole.

## Thompson Sampling

Another acquisition function with its roots in the multi-armed bandit literature is known as **Thompson sampling** (TS). Unlike the other acquisition functions we will consider, TS is an inherently stochastic policy.<sup>1</sup>

We can define the TS acquisition function as

$$a_{\text{TS}}(\mathbf{x}) = g(\mathbf{x}) \text{ where } g \sim \mathcal{GP}(f; \mu_{\mathcal{D}}, k_{\mathcal{D}}).$$

In words, Thompson sampling simply operates by drawing a sample path from the posterior GP belief on the objective function and observing the location where the sample is maximized.

While seemingly straightforward, **exactly optimizing a sample path from a GP belief is non-trivial**. Two common approaches are described below.

### “Brute-force” or Exhaustive Sampling

For sufficiently low-dimensional (or ideally, discrete) domains, a reasonable approximation is to sample from the GP belief at a dense grid of locations, typically evenly spaced according to some low-discrepancy sequence, and then observe the location with the highest sampled function value. This amounts to simply sampling from a multivariate Gaussian distribution, which can be done

<sup>1</sup>Some of the approximations we will introduce for the intractable acquisition functions introduce randomness in the approximation but the functions we are approximately are themselves deterministic quantities.

efficiently for a small-ish number of locations (i.e.,  $< 100,000$ ) but high-dimensional domains or objective functions with very short relative length-scales, this may not give sufficient coverage of the possible inputs to accurately estimate the sample path's maximum.

Figure 2 shows the (approximate) distribution of the optimal location to observe,

$$p(x^* | \mathcal{D}) \text{ where } x^* = \arg \max_{x \in \mathcal{X}} f(x),$$

on our 1-dimensional running example.

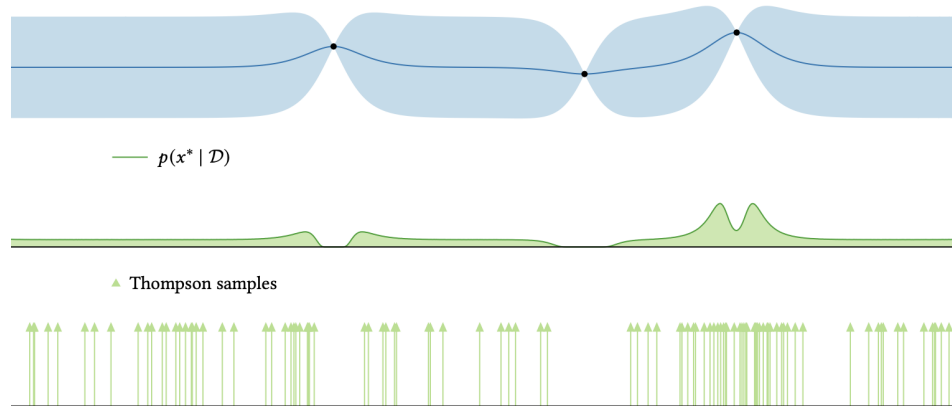


Figure 2: Thompson sampling used to approximate the distribution of the objective function's arg max: to generate the distribution (middle), 100 sample paths were drawn from the posterior GP belief (top) over a grid of 1000 evenly spaced locations; the location of the arg max of each sample path is shown as well (bottom).

### On-demand Sampling

If exhaustive sampling is prohibitively expensive, we can rely on iterative optimization algorithms (e.g., gradient descent) which only query a single point from our sample path at a time. Whenever our optimizer requires a new observation from our sample path, we simply draw a univariate sample from the Gaussian posterior belief at the requested location, conditioned on all the previous samples. It turns out that with our GP belief, it is even possible to sample a gradient of our sample path at any given location!

We will discuss this property and its implications at a later point but one key consideration is that incorporating gradient samples can significantly increase computational costs as we are effectively going from a single Gaussian sample in each iteration to  $d + 1$  samples, where  $d$  is the dimensionality of the domain (recall that the gradient contains an element for each dimension of the input).

### Entropy search

Finally, a recently proposed class of acquisitions functions leverages the notion of *mutual information* to guide the selection of observations; these are typically called some variant of the phrase **entropy search**. Entropy search acquisition functions seek to minimize the uncertainty we have about some quantity of interest. In the context of Bayesian optimization, there are two obvious quantities that we might wish to gain information about:

1. the optimal objective function value  $f^* = \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$
2. the location of the optimal objective function value  $\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$

Entropy search treats these quantities as random variables and seeks to evaluate points so as to maximize the mutual information between them and the selected observations. Recall that the mutual information between two random variables,  $X$  and  $Y$ , is given by

$$I(X; Y) = H(X) - \mathbb{E}_Y[H(X | Y)]$$

where  $H(X) = \mathbb{E}_X[\log(X)]$  is the entropy of  $X$ . One key property of mutual information that we exploit shortly is symmetry:  $I(X; Y) = I(Y; X)$ .

Let  $\omega$  be either of the two quantities of interest above. The implied utility function of these acquisition functions is then

$$u'(\mathcal{D}) = H(\omega) - H(\omega | \mathcal{D})$$

which gives rise to the point-wise utility function

$$\begin{aligned} u(\mathbf{x}) &= \mathbb{E}_{f(\mathbf{x})}[u'(\mathcal{D} \cup (\mathbf{x}, f(\mathbf{x})))] - u'(\mathcal{D}) \\ &= H(\omega | \mathcal{D}) - \mathbb{E}_{f(\mathbf{x})}[H(\omega | \mathcal{D} \cup (\mathbf{x}, f(\mathbf{x})))] = I(\omega; f(\mathbf{x}) | \mathbf{x}, \mathcal{D}). \end{aligned}$$

Figure 3 compares these two acquisition functions for both  $\omega = \mathbf{x}^*$  and  $\omega = f^*$ .

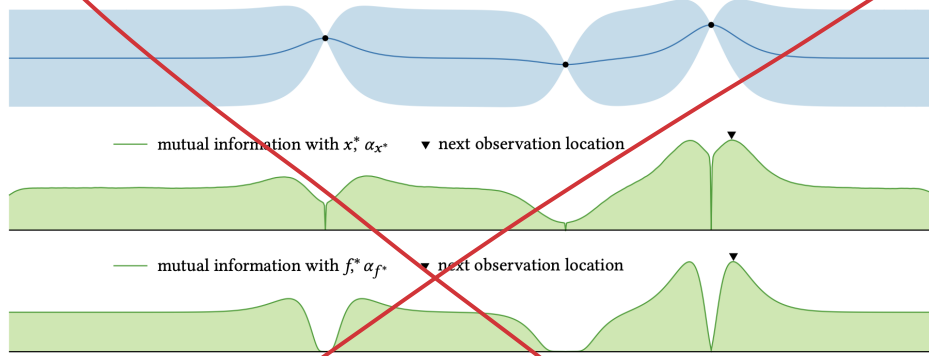


Figure 3: Entropy search for mutual information with respect to  $\mathbf{x}^*$  (middle) and  $f^*$  (bottom). Note the dips in the  $\text{ES}_{\mathbf{x}^*}$  acquisition function at the boundary: these dips are absent in all of the acquisition functions we’ve explored to date. This indicates that  $\text{ES}_{\mathbf{x}^*}$  is less interested in observing the boundary because it can only decrease our uncertainty about the optimal location “in one direction”: observations further from the boundary reduce the uncertainty over a larger portion of the domain.

Unfortunately, this acquisition function is intractable for both  $\mathbf{x}^*$  and  $f^*$ ; instead, a somewhat complicated sequence of approximations must be made in either case to estimate these acquisition functions. For brevity, we will present just an (abridged) approximation for  $\text{ES}_{\mathbf{x}^*}$ ; many corresponding approximations for  $\text{ES}_{f^*}$  follow a similar structure to the one presented below.

### Mutual Information with $\mathbf{x}^*$

Instead of computing  $I(\mathbf{x}^*; f(\mathbf{x}), \mathcal{D})$ , we will instead consider approximating the equivalent

$$I(f(\mathbf{x}); \mathbf{x}^* | \mathbf{x}, \mathcal{D}) = H(f(\mathbf{x}) | \mathbf{x}, \mathcal{D}) - \mathbb{E}_{\mathbf{x}^*} [H(f(\mathbf{x}) | \mathbf{x}^*, \mathbf{x}, \mathcal{D})].$$

Luckily, the first term is simple under the GP belief:

$$H(f(\mathbf{x}) | \mathbf{x}, \mathcal{D}) = \frac{1}{2} \log (2\pi e k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})).$$

The second term is harder to approximate. We begin by approximating the expectation via Monte Carlo integration:

$$\mathbb{E}_{\mathbf{x}^*} [H(f(\mathbf{x}) | \mathbf{x}^*, \mathbf{x}, \mathcal{D})] \approx \frac{1}{S} \sum_{s=1}^S H(f(\mathbf{x}) | \mathbf{x}_s^*, \mathbf{x}, \mathcal{D}) \text{ where } \mathbf{x}_s^* \sim p(\mathbf{x}^* | \mathcal{D}).$$

The necessary samples of  $\mathbf{x}^*$  can be drawn using Thompson sampling as detailed above.

The distribution  $p(f(\mathbf{x}) | \mathbf{x}^*, \mathbf{x}, \mathcal{D})$  is a strange one: intuitively, this distribution poses the question “how would knowing the *location* of the objective function’s optimum affect my belief about the functions value at some other location?”

If we could approximate this distribution to be Gaussian,

$$p(f(\mathbf{x}) | \mathbf{x}^*, \mathbf{x}, \mathcal{D}) \approx \mathcal{N}(f(\mathbf{x}); m^*, \sigma^{2*}),$$

then we would once again have a simple expression for the entropy in question:

$$H(f(\mathbf{x}) | \mathbf{x}_s^*, \mathbf{x}, \mathcal{D}) \approx \frac{1}{2} \log (2\pi e \sigma_s^{2*})$$

Putting this together with the previous term gives the final approximation for the  $\text{ES}_{\mathbf{x}^*}$  acquisition function:

$$a_{\text{ES}} \approx \frac{1}{2} \log (2\pi e k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})) - \frac{1}{S} \sum_{s=1}^S \frac{1}{2} \log (2\pi e \sigma_s^{2*}) = \log \left( \sqrt{k_{\mathcal{D}}(\mathbf{x}, \mathbf{x})} \right) - \frac{1}{S} \sum_{s=1}^S \log (\sigma_s^*).$$

Of course, the key unanswered question is how to approximate  $p(f(\mathbf{x}) | \mathbf{x}^*, \mathbf{x}, \mathcal{D})$  as a Gaussian. [Hernández-Lobato et al. \(2014\)](#) proposed using expectation propagation (EP) to do so, which gives rise to the **predictive entropy search** or PES acquisition function.

At a high level, their proposed EP approximation begins with  $p(f(\mathbf{x}) | \mathbf{x}, \mathcal{D})$  as the “prior”,  $p_0(f(\mathbf{x}))$ . They then multiply this prior by a sequence of “likelihood” terms which correspond to enforcing that  $\mathbf{x}^*$  satisfies certain properties of a global maximum. At a high level, there is

- a term that enforces the gradient at  $\mathbf{x}^*$  is  $\mathbf{0}$ ,
- a term that enforces the Hessian at  $\mathbf{x}^*$  is negative semi-definite,
- a term that enforces  $f^* > f'$  (recall that  $f'$  is the maximum observed function value in  $\mathcal{D}$ ), and
- a term that enforces  $f^* > f(\mathbf{x})$ .

The first two terms correspond to  $\mathbf{x}^*$  being a *local* maximum whereas the second two terms loosely enforce that  $\mathbf{x}^*$  is globally optimal: they are necessary but not sufficient conditions although given that this is already an approximation, the somewhat loose correspondence can be forgiven.

Figure 4 shows the EP approximated posterior of  $f$  using this set of implied constraints: the effect is minimal to non-existent at locations far from  $\mathbf{x}^*$  but the (approximate) conditioning on  $\mathbf{x}^*$  has a clear impact on the posterior belief of nearby function values.

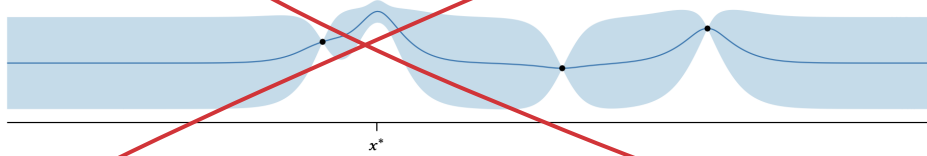


Figure 4: The approximate posterior belief on the objective function after using the PES EP approximation. The location being conditioned on was sampled using Thompson sampling.