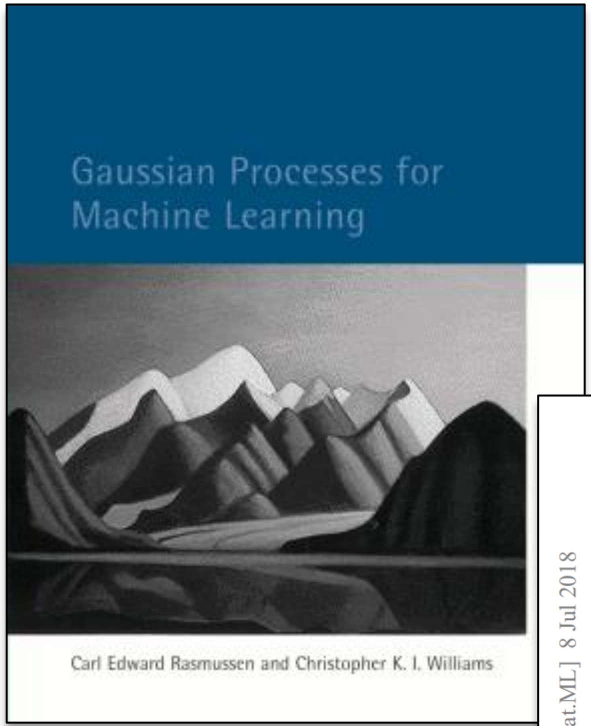
Gaussian Processes and Bayesian Optimization

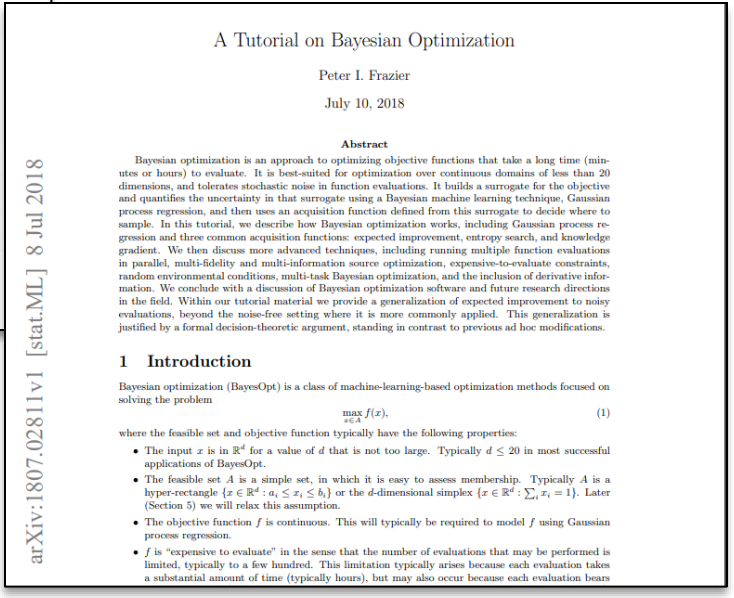
Assoc. Prof. Karl Ezra Pilario, Ph.D.
Process Systems Engineering Laboratory
Department of Chemical Engineering
University of the Philippines Diliman

Outline

- Motivation for GPs
- Bayesian Statistics
  - Bayes Theorem
  - Bayesians vs. Frequentists
- Gaussian Processes
  - Derivation
  - Kernels
- Bayesian Optimization
  - Algorithm
  - Acquisition Functions



Rasmussen and Williams (2006)
Gaussian Processes for Machine Learning.
MIT Press.
https://gaussianprocess.org/gpml/



A Tutorial on Bayesian Optimization

Peter I. Frazier

July 10, 2018

arXiv:1807.02811v1 [stat.ML] 8 Jul 2018

Abstract

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample. In this tutorial, we describe how Bayesian optimization works, including Gaussian process regression and three common acquisition functions: expected improvement, entropy search, and knowledge gradient. We then discuss more advanced techniques, including running multiple function evaluations in parallel, multi-fidelity and multi-information source optimization, expensive-to-evaluate constraints, random environmental conditions, multi-task Bayesian optimization, and the inclusion of derivative information. We conclude with a discussion of Bayesian optimization software and future research directions in the field. Within our tutorial material we provide a generalization of expected improvement to noisy evaluations, beyond the noise-free setting where it is more commonly applied. This generalization is justified by a formal decision-theoretic argument, standing in contrast to previous ad hoc modifications.

1 Introduction

Bayesian optimization (BayesOpt) is a class of machine-learning-based optimization methods focused on solving the problem

$$\max_{x \in A} f(x), \quad (1)$$

where the feasible set and objective function typically have the following properties:

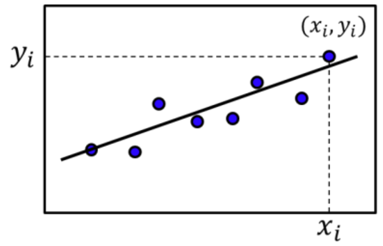
- The input x is in $\mathbb{R}^d$ for a value of d that is not too large. Typically $d \leq 20$ in most successful applications of BayesOpt.
- The feasible set A is a simple set, in which it is easy to assess membership. Typically A is a hyper-rectangle $\{x \in \mathbb{R}^d : a_i \leq x_i \leq b_i\}$ or the d-dimensional simplex $\{x \in \mathbb{R}^d : \sum_i x_i = 1\}$. Later (Section 5) we will relax this assumption.
- The objective function f is continuous. This will typically be required to model f using Gaussian process regression.
- f is "expensive to evaluate" in the sense that the number of evaluations that may be performed is limited, typically to a few hundred. This limitation typically arises because each evaluation takes a substantial amount of time (typically hours), but may also occur because each evaluation bears

Frazier (2018). A Tutorial on Bayesian Optimization
https://arxiv.org/pdf/1807.02811.pdf

Motivation for Gaussian Processes

Recall: From **Linear Regression** to **Kernel Ridge Regression**

How to compute for weights, w_i?



Linear Regression: $y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$

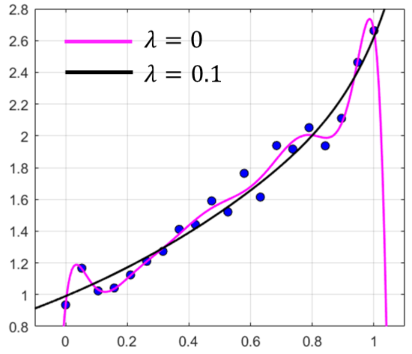
$$\boxed{\hat{\boldsymbol{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{y}}$$

Transform the X using nonlinear functions, ϕ_i

Linear Basis Function Model: $y = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \cdots + w_m\phi_m(x)$

$$\boxed{\hat{\boldsymbol{w}} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}}$$

Add regularization to penalize too large weights



Ridge Regression: $y = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \cdots + w_m\phi_m(x)$

$$\boxed{\hat{\boldsymbol{w}} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\boldsymbol{I})^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}}$$

Apply the kernel trick, similar to SVMs, using kernel functions

$$K = \langle \phi_i, \phi_j \rangle$$

Kernel Ridge Regression: $y = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \cdots + w_m\phi_m(x)$

$$\boxed{y_{\text{pred}} = \boldsymbol{k}(x)^T(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}}$$

Motivation for Gaussian Processes

*KRR only outputs a **mean** prediction.*

Kernel Ridge Regression (KRR):

$$y = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \cdots + w_m\phi_m(x)$$

$$\boxed{y_{\text{pred}} = \boldsymbol{k}(x)^T(\boldsymbol{K} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}}$$

*GPR outputs both a **mean and distribution** prediction.*

Gaussian Process Regression (GPR):

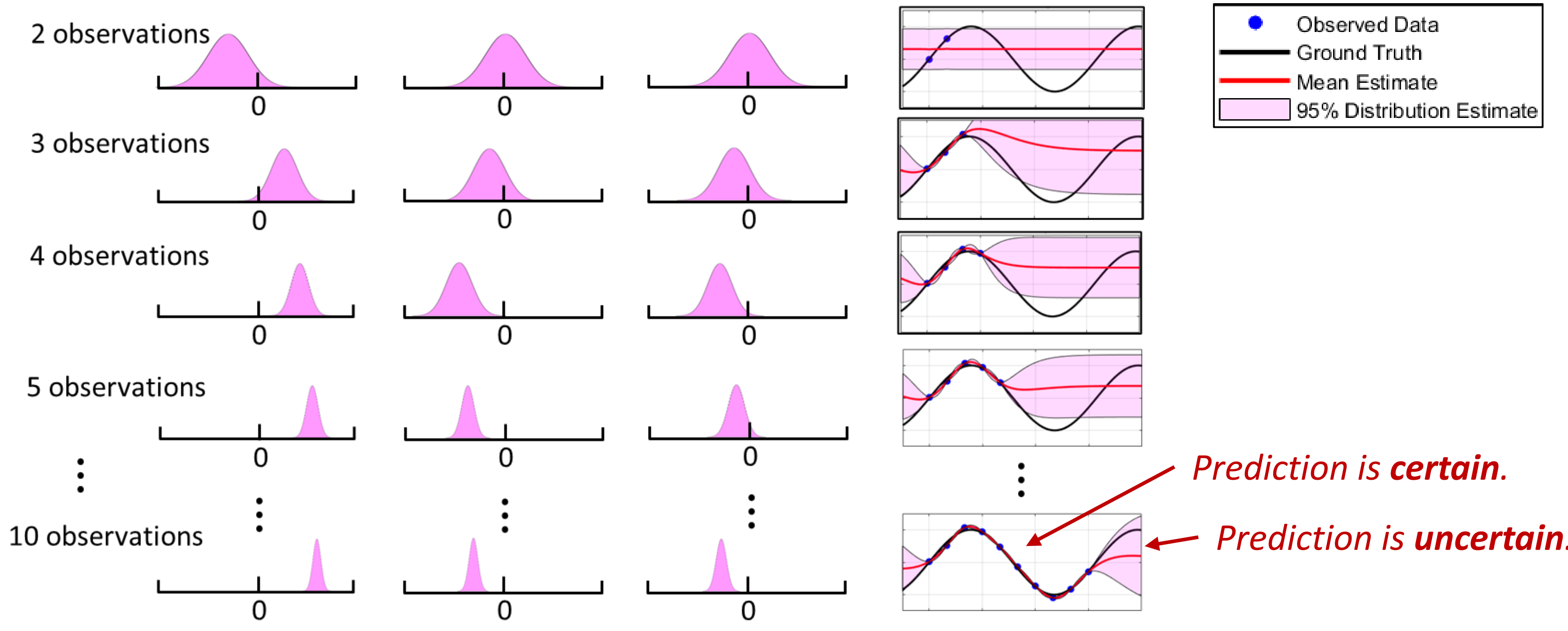
$$\boxed{\text{mean}(y^*|x^*) = k(x^*, \boldsymbol{x})^T[\boldsymbol{K} + \sigma^2\boldsymbol{I}]^{-1}\boldsymbol{y}}$$

$$\boxed{\text{var}(y^*|x^*) = k(x^*, x^*) + \sigma^2 - k(x^*, \boldsymbol{x})^T[\boldsymbol{K} + \sigma^2\boldsymbol{I}]^{-1}k(x^*, \boldsymbol{x})}$$

Derived using Bayes Theorem.

To appreciate GPR, imagine that data points only arrive one at a time…

$$y = \quad w_0 \quad + \quad w_1\phi_1(\boldsymbol{x}) \quad + \quad w_2\phi_2(\boldsymbol{x})$$



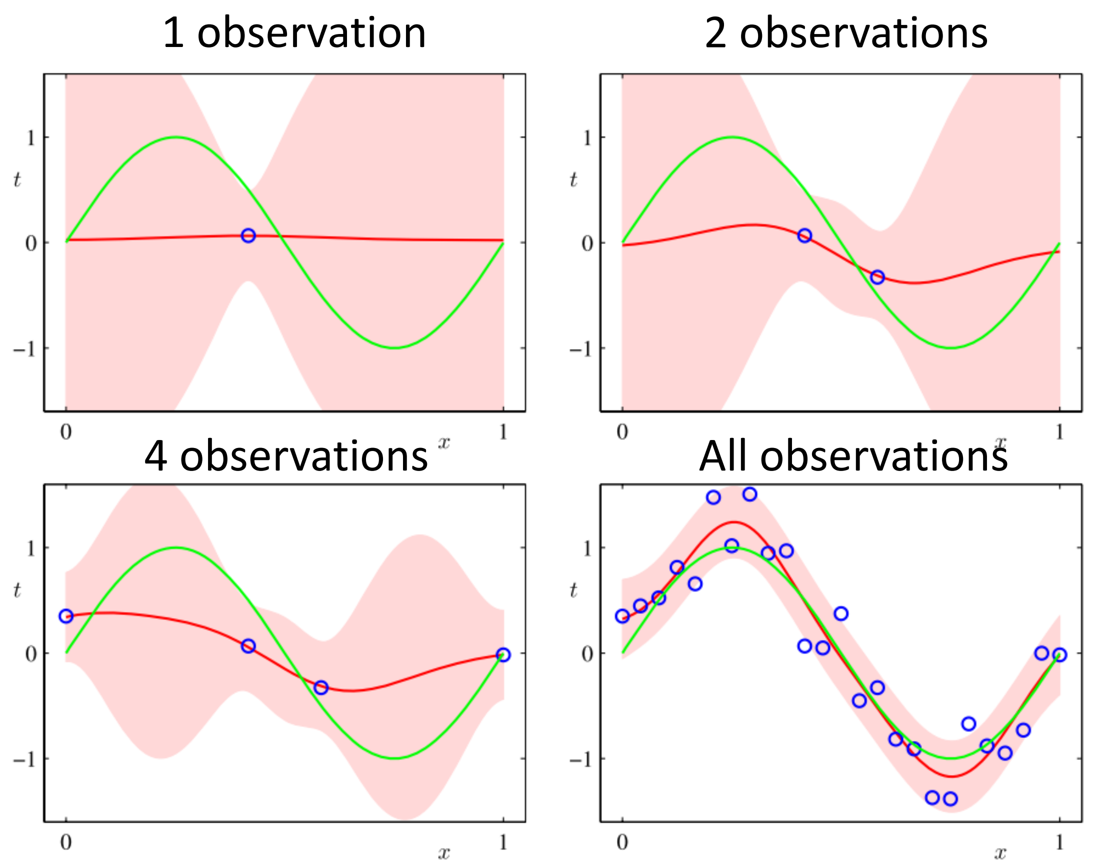
*Prediction is **certain**.*

*Prediction is **uncertain**.*

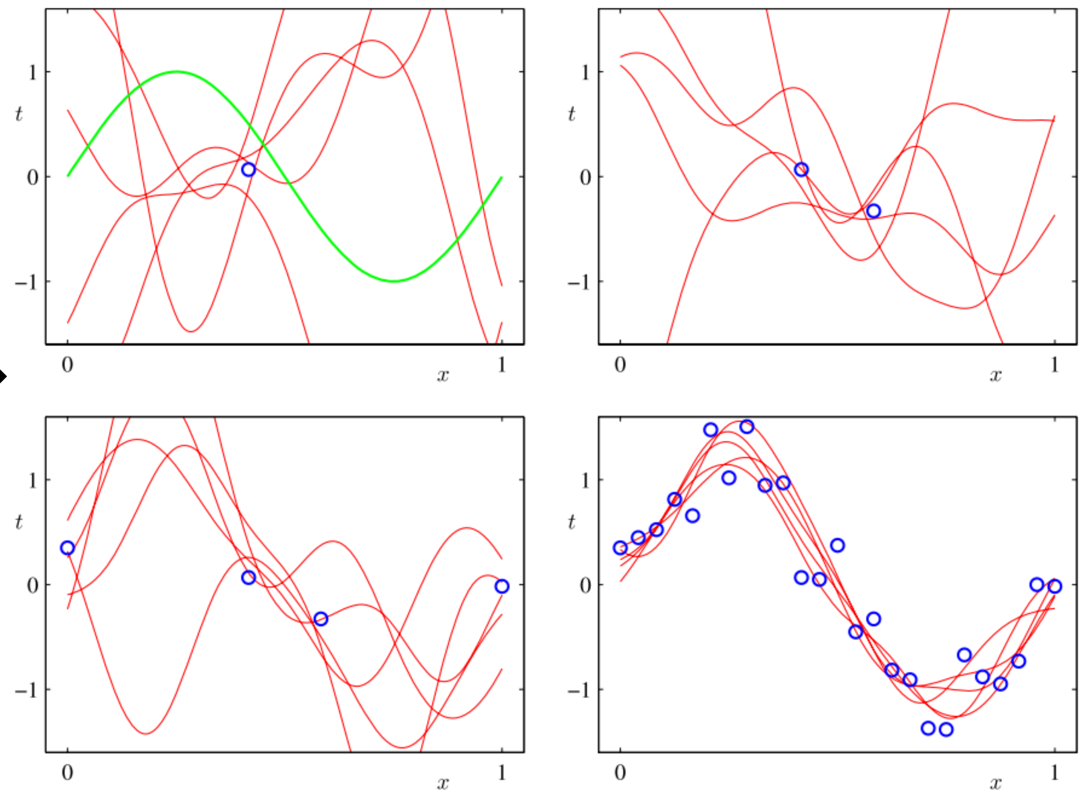
Motivation for Gaussian Processes

Having a *predictive distribution* is important in understanding the uncertainty of the model.

🟩	Ground truth
🟥	Predictions
▨	Uncertainty
○	Samples



Functions sampled from the distribution:



Bayesian Statistics

Notation:

If X is a random variable that is Gaussian or *normally* distributed, then:

$$p(X) = \mathcal{N}(\mu, \sigma^2)$$

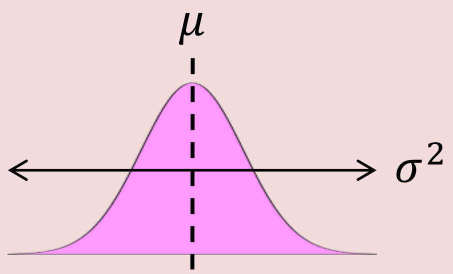
or

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

where $\mathcal{N}$ stands for "*normal*".

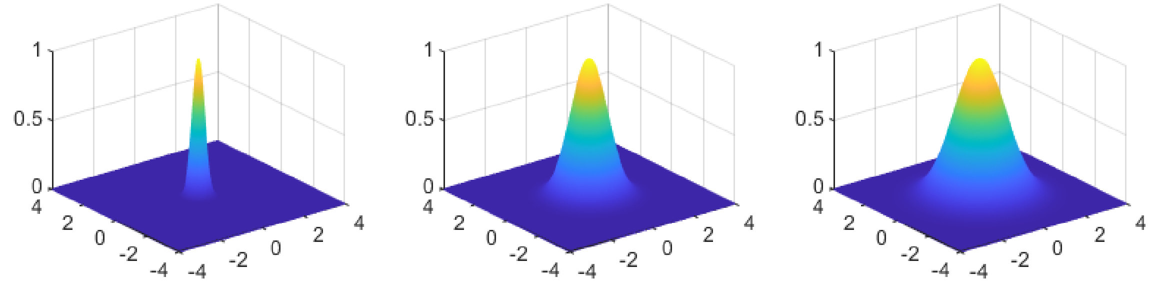
The normal distribution is parametrized by a mean, μ, and a variance σ^2:

$$\mathcal{N}(X|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(X-\mu)^2}{2\sigma^2}\right)$$



Multivariate Normal Distribution:

$$\mathcal{N}(\boldsymbol{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})\right)$$



$\boldsymbol{\mu} \in \Re^D$	mean vector (D-dimensional)
$\boldsymbol{\Sigma} \in \Re^{D \times D}$	covariance matrix (D-by-D)
$|\boldsymbol{\Sigma}| \in \Re$	determinant of the covariance matrix

In order to understand Bayesian statistics, let's turn to an example involving discrete events…

Bayesian Statistics

The Coffee Meet-up Example

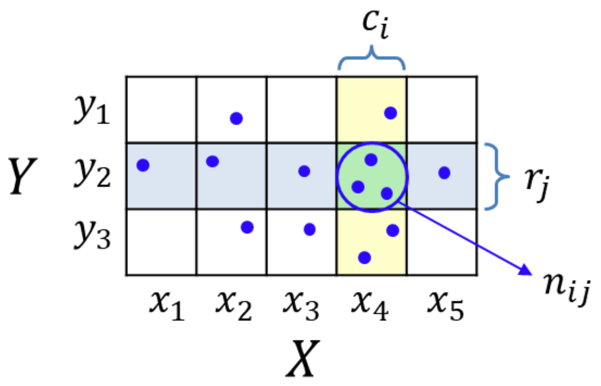
For the sake of example, let Y be the *weather condition* and X be the event that your friends *come* to meet you for coffee:

y_1 = the weather is sunny
y_2 = slightly raining
y_3 = a storm is coming

x_1 = Friend 1 comes
x_2 = Friend 2 comes
x_3 = Friend 3 comes
x_4 = Friend 4 comes
x_5 = Friend 5 comes

X and Y are two random variables that can take on values $\{y_j\}$ and $\{x_i\}$.

In this example, events were recorded from the *history of past coffee meet-ups*.



$p(X) = \left[\frac{1}{13}, \frac{3}{13}, \frac{2}{13}, \frac{6}{13}, \frac{1}{13}\right]$

$p(Y) = \left[\frac{2}{13}, \frac{7}{13}, \frac{4}{13}\right]$

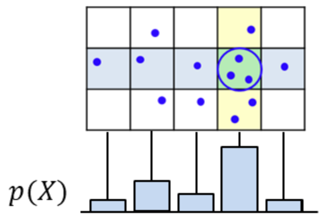
Things that we can say for the next meet-up:

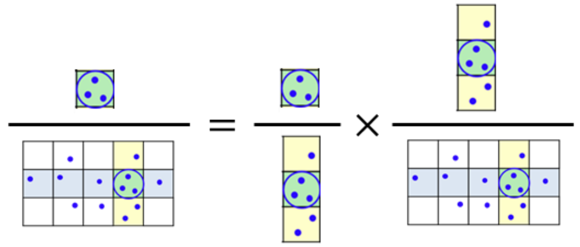
- The probability that $X = x_4$ is $p(X = x_4) = \frac{c_i}{N} = \frac{6}{13}$
- The joint probability, $p(x_4, y_2)$, is $p(X = x_4, Y = y_2) = \frac{n_{ij}}{N} = \frac{3}{13}$
- The conditional probability, $p(y_2|x_4)$ is $p(Y = y_2|X = x_4) = \frac{n_{ij}}{c_i} = \frac{3}{6}$
- The conditional probability, $p(x_4|y_2)$ is $p(X = x_4|Y = y_2) = \frac{n_{ij}}{r_i} = \frac{3}{7}$

Rules of Probability:

$$\boxed{p(X) = \sum_Y p(X,Y)}$$

$$\boxed{p(X,Y) = p(Y|X)p(X)}$$





A note in notation:
The quantity $p(X = x_4)$ denotes the specific probability for event x_4, but the quantity $p(X)$ is a distribution of probabilities on all events of X.

Bayesian Statistics

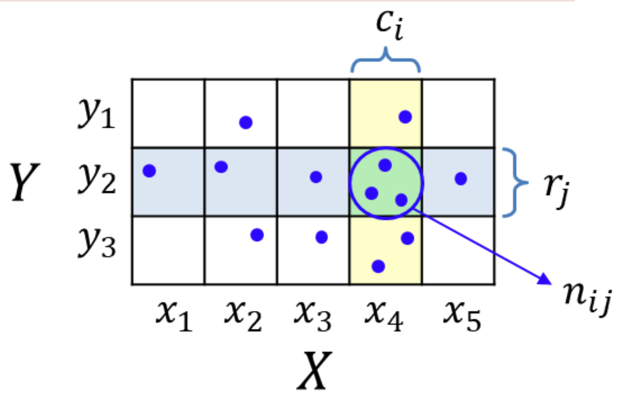
Rules of Probability: $\boxed{p(X) = \sum_Y p(X,Y)}$ $\boxed{p(X,Y) = p(Y|X)p(X)}$

(Sum Rule) (Product Rule)

Based on the product rule and the symmetry of joint probabilities, i.e. $p(X,Y) = p(Y,X)$, we can derive the following:

Bayes' Theorem $\boxed{p(Y|X) = \dfrac{p(X|Y)p(Y)}{p(X)}}$

Example:



$$\left.\begin{aligned}
p(Y = y_2|X = x_4) &= \frac{3}{6}\\
p(X = x_4|Y = y_2) &= \frac{3}{7}\\
p(Y = y_2) &= \frac{7}{13}\\
p(X = x_4) &= \frac{6}{13}
\end{aligned}\right\} \frac{3}{6} = \frac{{}^{3}\!/_{7} \times {}^{7}\!/_{13}}{{}^{6}\!/_{13}}$$

If our random variables can take continuous values rather than discrete, then:

(Sum Rule) $p(x) = \int p(x,y)\ dy$

(Product Rule) $p(x,y) = p(y|x)p(x)$

Together, we have: $p(x) = \int p(y|x)p(x)\ dy$

Upon substituting into Bayes' Theorem:

$$\underset{\text{posterior}}{p(y|x)} = \frac{\overset{\text{likelihood}}{p(x|y)}\ \overset{\text{prior}}{p(y)}}{\underbrace{\int p(y|x)p(x)\,dy}_{\text{Marginal likelihood}}}$$

Bayesian Statistics: A Change of Perspective

Bayes Theorem:

$$\overset{\text{posterior}}{p(y|x)} = \underbrace{\frac{\overset{\text{likelihood}}{p(x|y)}\ \overset{\text{prior}}{p(y)}}{\int p(y|x)p(x)\,dy}}_{\text{Marginal likelihood}}$$

Frequentists vs. Bayesians

In Frequentist statistics, $p(x)$ refers to the **frequency** of occurrence of an event. In Bayesian statistics, $p(x)$ refers to the **belief** that event x is true.

For Frequentists, $p(x)$ is **fixed**, and its correct value can be approached with more and more data (evidence).

For Bayesians, beliefs $p(x)$ **can change**, and there is no single correct value to aim for. New data will always cause the belief to change.

Bayesian Inference in machine learning models:

Let: H = hypothesis (current belief of the world)
D = new data

$$p(H|D) = \frac{p(D|H)\ p(H)}{p(D)}$$

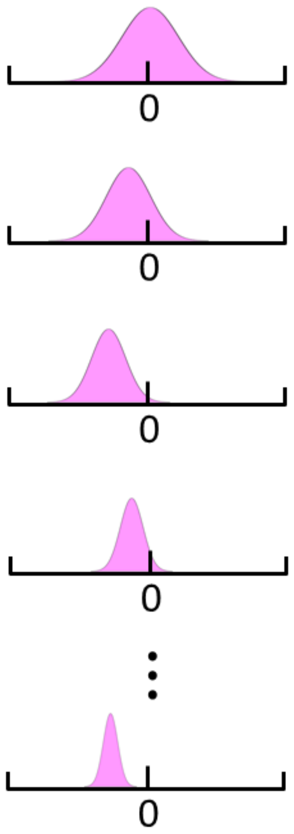
Belief that *H* is true, GIVEN the new data *D*

Belief that data *D* can exist GIVEN that *H* is true

Prior belief that *H* is true

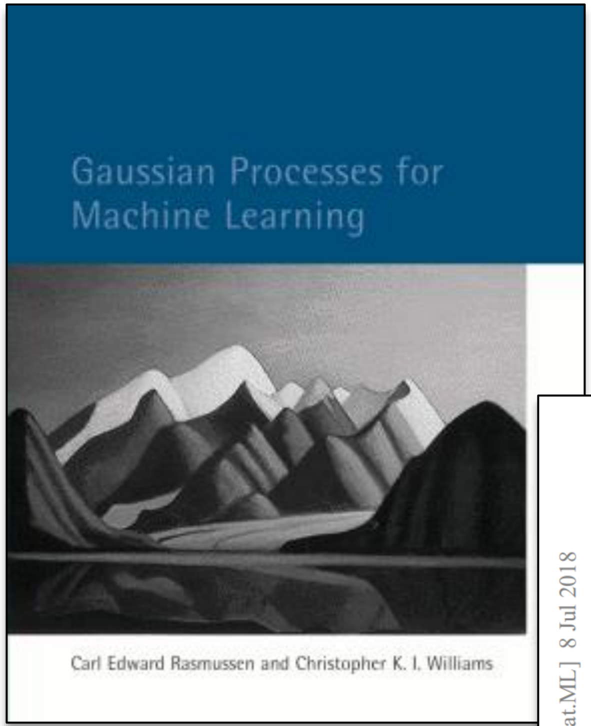
Total belief that data *D* can exist considering ALL possible hypotheses

$p(H) = p(\mathbf{w})$



Outline

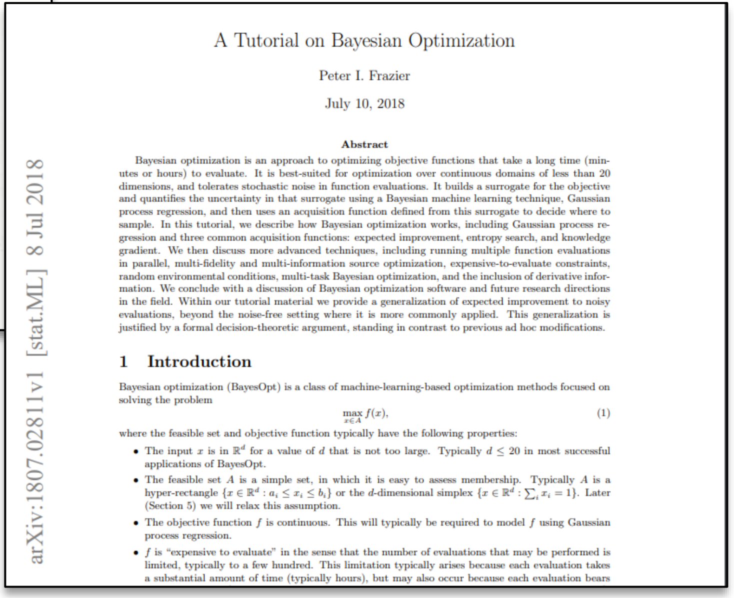
- Motivation for GPs
- Bayesian Statistics
  - Bayes Theorem
  - Bayesians vs. Frequentists
- **Gaussian Processes**
  - **Derivation**
  - **Kernels**
- Bayesian Optimization
  - Algorithm
  - Acquisition Functions



Gaussian Processes for Machine Learning

Carl Edward Rasmussen and Christopher K. I. Williams

Rasmussen and Williams (2006)
Gaussian Processes for Machine Learning.
MIT Press.
https://gaussianprocess.org/gpml/



A Tutorial on Bayesian Optimization

Peter I. Frazier

July 10, 2018

arXiv:1807.02811v1 [stat.ML] 8 Jul 2018

Abstract

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample. In this tutorial, we describe how Bayesian optimization works, including Gaussian process regression and three common acquisition functions: expected improvement, entropy search, and knowledge gradient. We then discuss more advanced techniques, including running multiple function evaluations in parallel, multi-fidelity and multi-information source optimization, expensive-to-evaluate constraints, random environmental conditions, multi-task Bayesian optimization, and the inclusion of derivative information. We conclude with a discussion of Bayesian optimization software and future research directions in the field. Within our tutorial material we provide a generalization of expected improvement to noisy evaluations, beyond the noise-free setting where it is more commonly applied. This generalization is justified by a formal decision-theoretic argument, standing in contrast to previous ad hoc modifications.

1 Introduction

Bayesian optimization (BayesOpt) is a class of machine-learning-based optimization methods focused on solving the problem

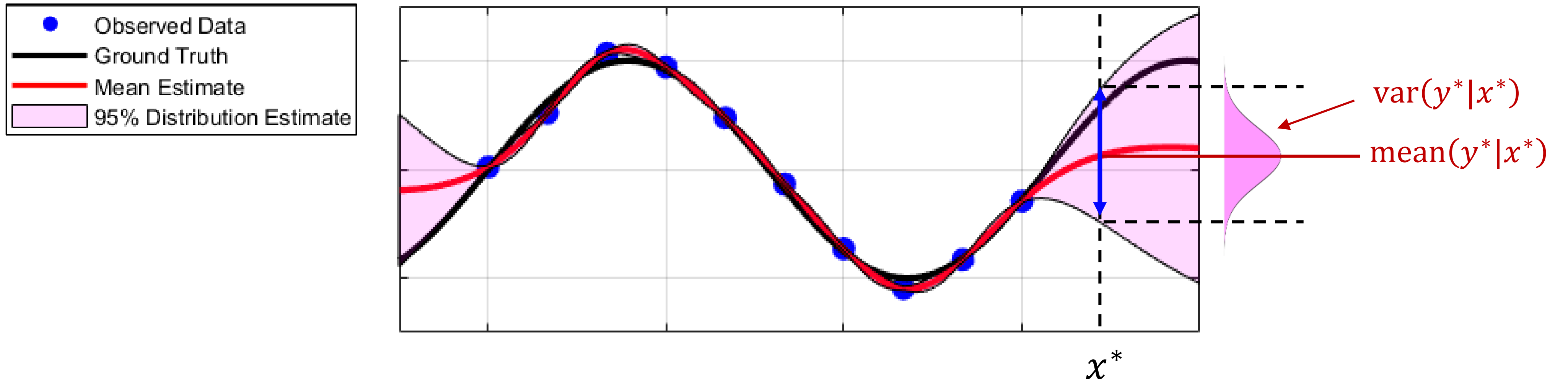
$$\max_{x \in A} f(x), \qquad (1)$$

where the feasible set and objective function typically have the following properties:

- The input x is in $\mathbb{R}^d$ for a value of d that is not too large. Typically $d \leq 20$ in most successful applications of BayesOpt.
- The feasible set A is a simple set, in which it is easy to assess membership. Typically A is a hyper-rectangle $\{x \in \mathbb{R}^d : a_i \leq x_i \leq b_i\}$ or the d-dimensional simplex $\{x \in \mathbb{R}^d : \sum_i x_i = 1\}$. Later (Section 5) we will relax this assumption.
- The objective function f is continuous. This will typically be required to model f using Gaussian process regression.
- f is "expensive to evaluate" in the sense that the number of evaluations that may be performed is limited, typically to a few hundred. This limitation typically arises because each evaluation takes a substantial amount of time (typically hours), but may also occur because each evaluation bears

Frazier (2018). A Tutorial on Bayesian Optimization
https://arxiv.org/pdf/1807.02811.pdf

Gaussian Process Regression



These are the quantities that we need to derive.

Training Data: $\{\boldsymbol{x}, \boldsymbol{y}\}, \quad \boldsymbol{x} \in \Re^{N\times D}, \quad \boldsymbol{y} \in \Re^{N}$

New Data: $\boldsymbol{x}^*$

Model (M): $\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon}$

Desired prediction: $y^* \to \mathrm{mean}(y^*|x^*)$, $y^* \to \mathrm{var}(y^*|x^*)$

Assumptions:

1. The model is linear: $\boldsymbol{y} = \boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{\varepsilon}$
2. The noise is Gaussian: $p(\boldsymbol{\varepsilon}) = \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$
3. The prior on $\boldsymbol{w}$ is Gaussian: $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$
4. The likelihood is Gaussian: $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}^T\boldsymbol{x}, \sigma^2\boldsymbol{I})$

Bayes Theorem:

$$\underbrace{p(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})}^{\text{likelihood}}\ \overbrace{p(\boldsymbol{w})}^{\text{prior}}}{\int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{w})\, d\boldsymbol{w}}$$

$$\overset{\text{Prediction, } y^*}{p(y^*|x^*, \boldsymbol{x}, \boldsymbol{y})} = \int \overset{y^* = \boldsymbol{w}^T x^* + \boldsymbol{\varepsilon}}{p(y^*|\boldsymbol{w}, x^*)}\ \overset{\text{posterior}}{p(\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{y})}\ d\boldsymbol{w}$$

Gaussian Process Regression

Bayes Theorem:

$$\overset{\text{posterior}}{p(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y})} = \frac{\overset{\text{likelihood}}{p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})}\,\overset{\text{prior}}{p(\boldsymbol{w})}}{\int p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{w})\,p(\boldsymbol{w})\,d\boldsymbol{w}}$$

$$\overset{\text{Prediction, } y^*}{p(y^*|x^*,\boldsymbol{x},\boldsymbol{y})} = \int \overset{y^* = \boldsymbol{w}^T x^* + \boldsymbol{\varepsilon}}{p(y^*|\boldsymbol{w},x^*)}\,\overset{\text{posterior}}{p(\boldsymbol{w}|\boldsymbol{x},\boldsymbol{y})}\;d\boldsymbol{w}$$

Using a Gaussian distribution makes the integration tractable because the result of multiplying two Gaussians is still a Gaussian.

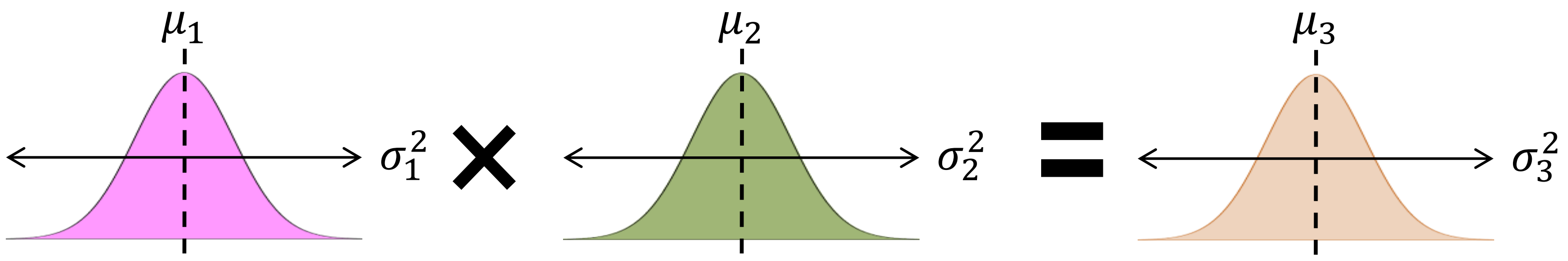
Prediction in GPR:

These are the main equations in GPR.

$$\boxed{\begin{aligned} \text{mean}(y^*|x^*) &= k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}\boldsymbol{y} \\ \text{var}(y^*|x^*) &= k(x^*,x^*)+\sigma^2 - k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}k(x^*,\boldsymbol{x}) \end{aligned}}$$

$\{\boldsymbol{x},\boldsymbol{y}\}$ = Training data
$\boldsymbol{x}^*$ = New Data
y^* = Desired prediction
σ^2 = variance of noise, ε

$\boldsymbol{K}$ = prior covariance kernel on $\boldsymbol{w}$
$k(\cdot,\cdot)$ = kernel function



$$\boxed{\mu_3 = \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}}$$

$$\boxed{\sigma_3^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}}$$

Gaussian Process Regression

Definition of a GP: A Gaussian Process is a collection of random variables any finite number of which have (consistent) joint Gaussian distributions.

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{\varepsilon}$$

$$\boxed{f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}) = \boldsymbol{0}, k(\boldsymbol{x}, \boldsymbol{x}'))}$$

$$\text{mean}(y^*|x^*) = k(x^*, \boldsymbol{x})^T [\boldsymbol{K} + \sigma^2 \boldsymbol{I}]^{-1} \boldsymbol{y}$$

$$\text{var}(y^*|x^*) = k(x^*, x^*) + \sigma^2 - k(x^*, \boldsymbol{x})^T [\boldsymbol{K} + \sigma^2 \boldsymbol{I}]^{-1} k(x^*, \boldsymbol{x})$$

Hyper-parameters in GPR: $k(x, x'), \theta_i, \sigma^2$

To optimize hyper-parameters, GPR packages use gradient descent. The gradient can be calculated by:

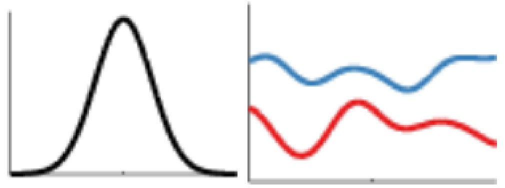
$$\frac{\partial}{\partial \theta_i} \ln p(\mathrm{y}|\boldsymbol{\theta}) = -\frac{1}{2}\mathrm{tr}\left(\mathrm{C}^{-1}\frac{\partial \mathrm{C}}{\partial \theta_i}\right) + \frac{1}{2}\mathrm{y}^T \mathrm{C}^{-1}\frac{\partial \mathrm{C}}{\partial \theta_i}\mathrm{C}^{-1}\mathrm{y}$$

where $\boldsymbol{C} = \boldsymbol{K} + \sigma^2 \boldsymbol{I}$.

Kernel Functions in GPR:

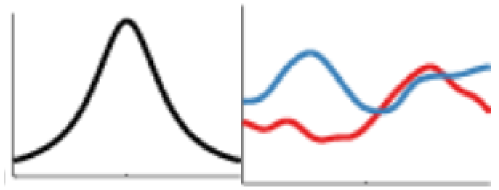
Squared Exponential

$$k_{SE} = \sigma^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right)$$



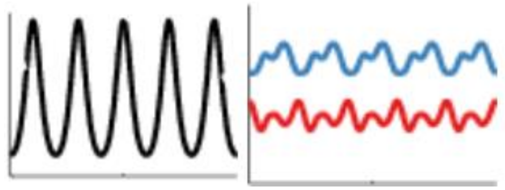
Rational Quadratic

$$k_{RQ} = \sigma^2 \left(1 + \frac{(x - x')^2}{2\alpha l^2}\right)$$



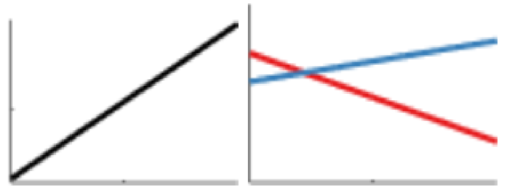
Periodic

$$k_{PER} = \sigma^2 \exp\left(-\frac{2}{l^2}\sin^2\frac{\pi|x - x'|}{p}\right)$$



Linear

$$k_{LIN} = \sigma_b^2 + \sigma_v^2 (x - c)(x' - c)$$



- Matern 3/2 kernel
- Matern 5/2 kernel
- Automatic Relevance Determination kernels

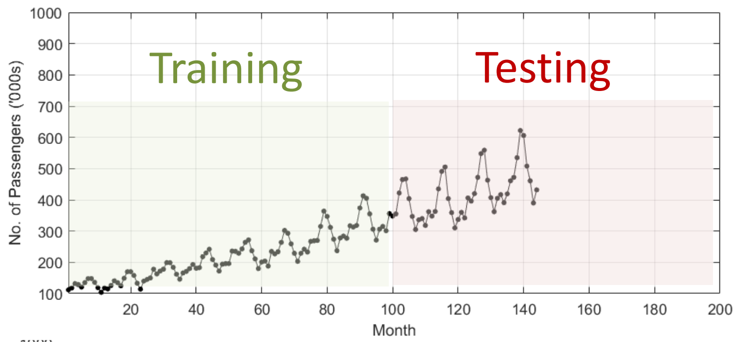
Gaussian Process Regression

Example:

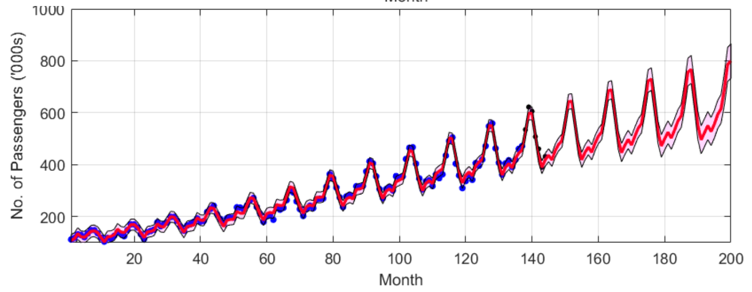
Find the mean and variance prediction of the Airline Passengers Data Set using the first 100 out of 144 data, and the following kernel function (SE x Periodic):

$$K(x,x') = \theta_1 \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\theta_2^2}\right) \times \exp\left(\frac{-2\sin^2(\pi|\boldsymbol{x}-\boldsymbol{x}'|/\theta_3)}{\theta_4^2}\right)$$

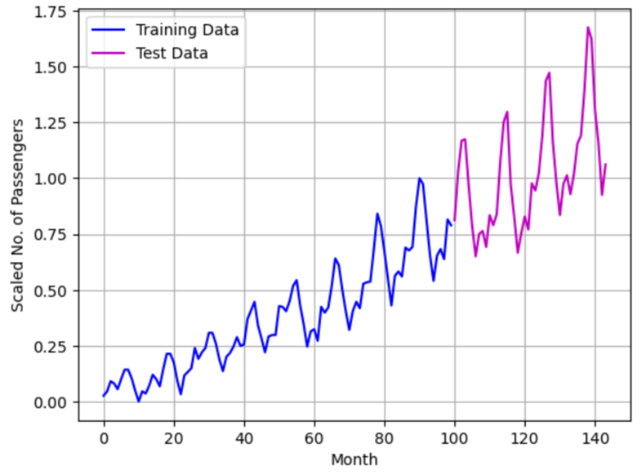
Given Data Set



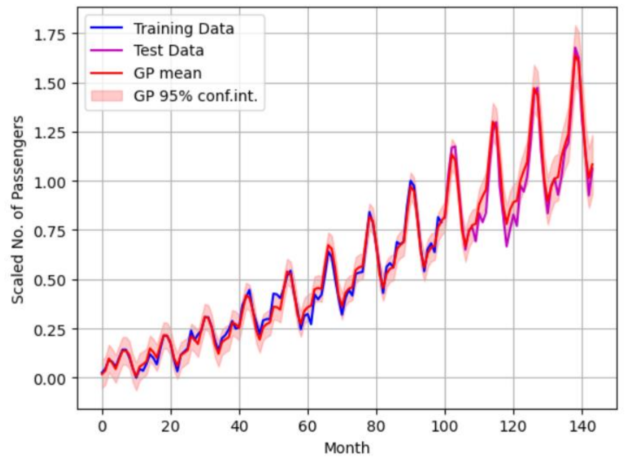
Given Data Set + GPR Predictions **(MATLAB)**



Given Data Set



Given Data Set + GPR Predictions **(Python)**



Optimized kernel:
RBF(length_scale=179) * ExpSineSquared(length_scale=1.57, periodicity=12) + WhiteKernel(noise_level=0.000967)

Training MSE: 0.0007858767229793949
Testing MSE: 0.004509114222736137

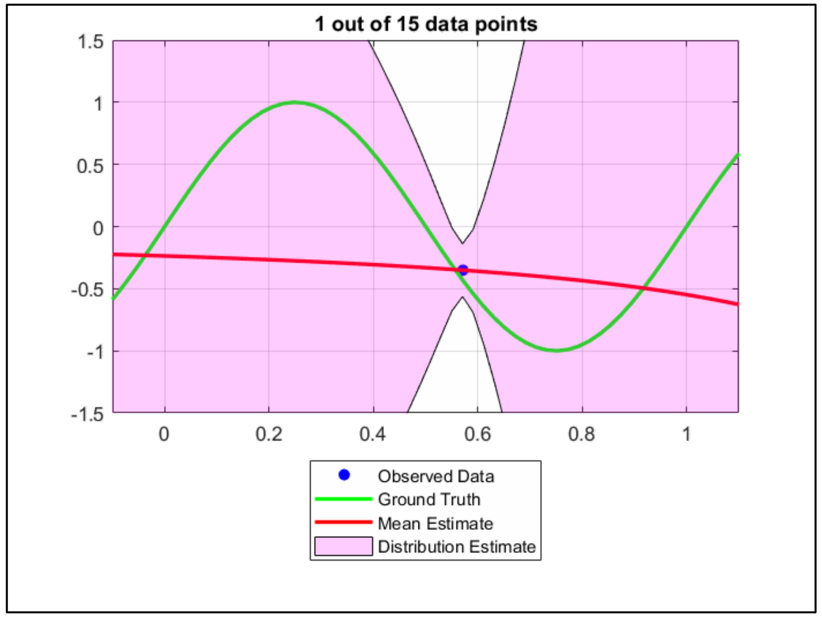
Gaussian Process Regression

Side Note on GPR:

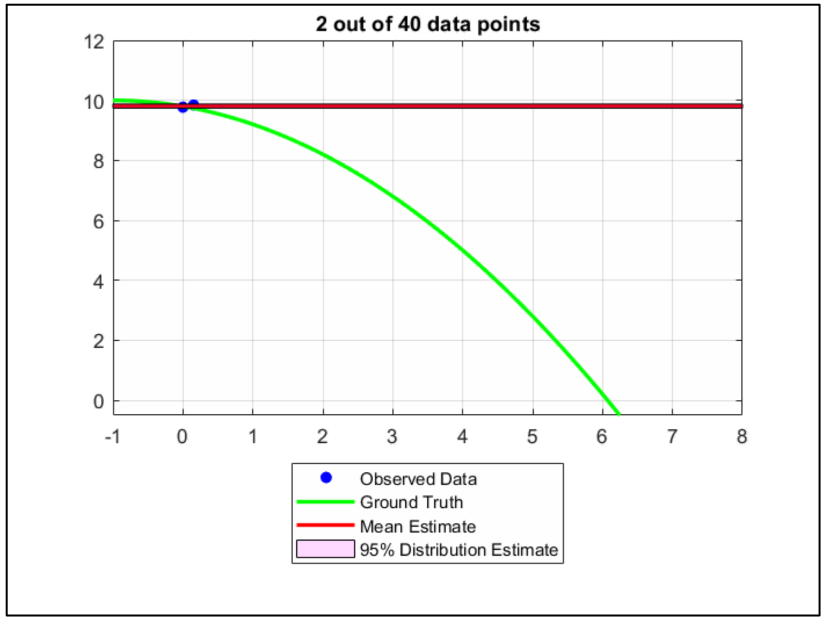
Neal (1996) has shown that for a broad class of prior distributions over $\boldsymbol{w}$, the distribution of functions generated by a single-hidden-layer *neural network* will tend to a Gaussian Process as the number of hidden neurons tend to infinity.

Other Notable Examples:

An example where data points from a sine wave arrive one at a time at random positions. The predictive distributions show which parts are uncertain to the model.

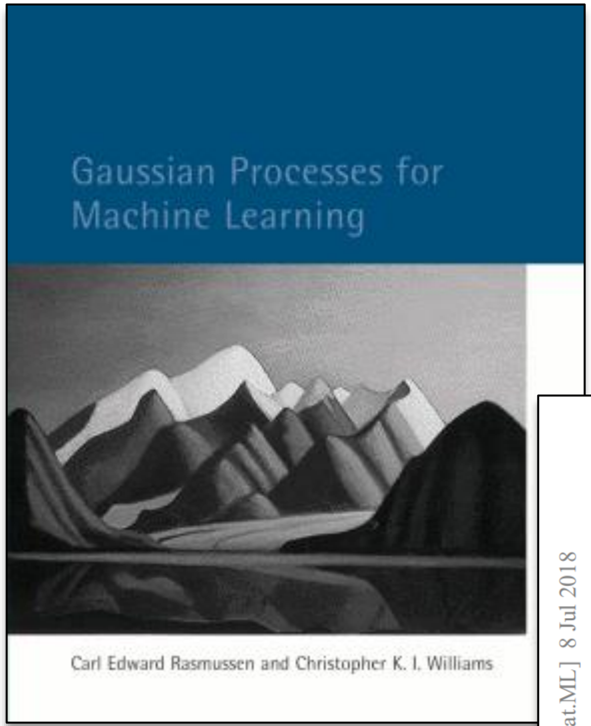


Prediction of Battery Degradation based on NASA data sets.



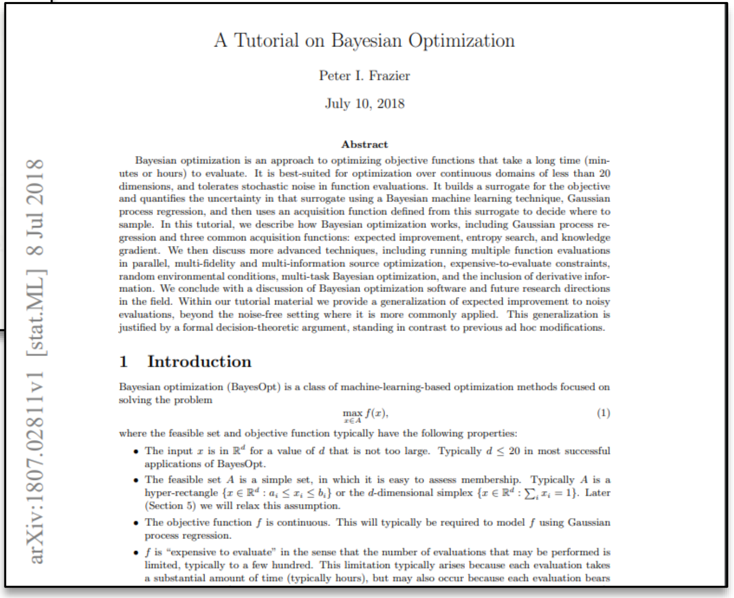
Outline

- Motivation for GPs
- Bayesian Statistics
  - Bayes Theorem
  - Bayesians vs. Frequentists
- Gaussian Processes
  - Derivation
  - Kernels
- **Bayesian Optimization**
  - **Algorithm**
  - **Acquisition Functions**


Gaussian Processes for Machine Learning

Carl Edward Rasmussen and Christopher K. I. Williams

Rasmussen and Williams (2006)
Gaussian Processes for Machine Learning.
MIT Press.
https://gaussianprocess.org/gpml/


arXiv:1807.02811v1 [stat.ML] 8 Jul 2018

A Tutorial on Bayesian Optimization

Peter I. Frazier

July 10, 2018

Abstract

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample. In this tutorial, we describe how Bayesian optimization works, including Gaussian process regression and three common acquisition functions: expected improvement, entropy search, and knowledge gradient. We then discuss more advanced techniques, including running multiple function evaluations in parallel, multi-fidelity and multi-information source optimization, expensive-to-evaluate constraints, random environmental conditions, multi-task Bayesian optimization, and the inclusion of derivative information. We conclude with a discussion of Bayesian optimization software and future research directions in the field. Within our tutorial material we provide a generalization of expected improvement to noisy evaluations, beyond the noise-free setting where it is more commonly applied. This generalization is justified by a formal decision-theoretic argument, standing in contrast to previous ad hoc modifications.

1 Introduction

Bayesian optimization (BayesOpt) is a class of machine-learning-based optimization methods focused on solving the problem

$$\max_{x \in A} f(x), \tag{1}$$

where the feasible set and objective function typically have the following properties:

- The input x is in $\mathbb{R}^d$ for a value of d that is not too large. Typically $d \leq 20$ in most successful applications of BayesOpt.
- The feasible set A is a simple set, in which it is easy to assess membership. Typically A is a hyper-rectangle $\{x \in \mathbb{R}^d : a_i \leq x_i \leq b_i\}$ or the d-dimensional simplex $\{x \in \mathbb{R}^d : \sum_i x_i = 1\}$. Later (Section 5) we will relax this assumption.
- The objective function f is continuous. This will typically be required to model f using Gaussian process regression.
- f is "expensive to evaluate" in the sense that the number of evaluations that may be performed is limited, typically to a few hundred. This limitation typically arises because each evaluation takes a substantial amount of time (typically hours), but may also occur because each evaluation bears

Frazier (2018). A Tutorial on Bayesian Optimization
https://arxiv.org/pdf/1807.02811.pdf

Bayesian Optimization

- An efficient algorithm for finding the optimum (minimum / maximum) of a function globally.

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \qquad \max_{\boldsymbol{x}} f(\boldsymbol{x})$$

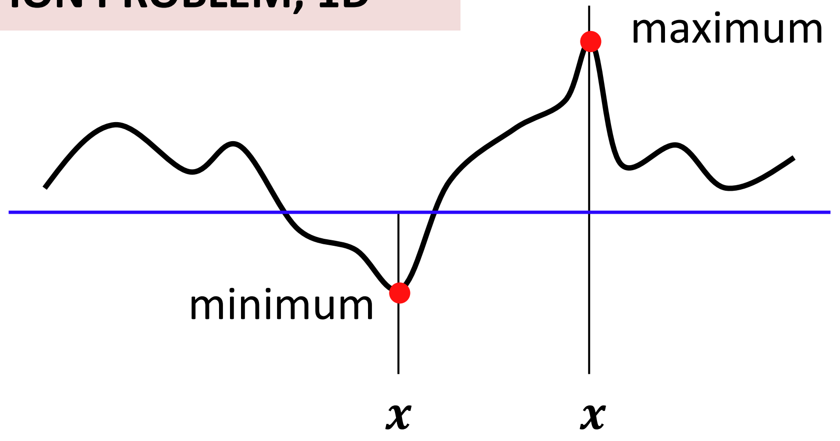
- First popularized as **EGO** (Efficient Global Optimization) algorithm by Jones et al. (1998).

- Now widely used for optimizing hyper-parameters in machine learning algorithms.

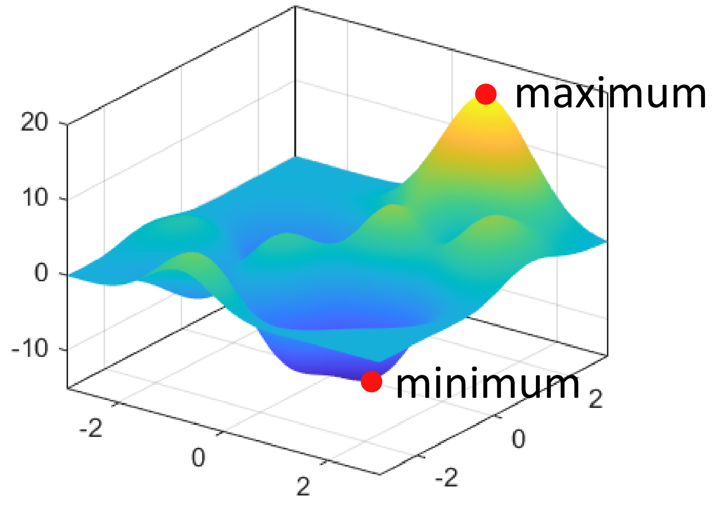
- **Advantage:** Sample-efficient, gradient-free!
- **Disadvantage:** Good for low-dimensional $\boldsymbol{x}$ only

- Built on two basic components:
    - **Surrogate Model**
    - **Acquisition Function**

OPTIMIZATION PROBLEM, 1D



OPTIMIZATION PROBLEM, 2D



Bayesian Optimization

Surrogate Model (Gaussian Process)

A **proxy** for the unknown objective function that is *sequentially fitted* to every incoming sample. It must provide both mean and variance estimates.

$$\mathrm{mean}(y|x^*) = k(x^*, \boldsymbol{x})^T[\boldsymbol{K} + \sigma^2\boldsymbol{I}]^{-1}\boldsymbol{y}$$

$$\mathrm{var}(y|x^*) = k(x^*, x^*) + \sigma^2 - k(x^*, \boldsymbol{x})^T[\boldsymbol{K} + \sigma^2\boldsymbol{I}]^{-1}k(x^*, \boldsymbol{x})$$

Acquisition Functions

A **policy** that evaluates the surrogate model output to determine the best place to sample the objective function next.

Expected Improvement

$$\mathrm{EI}_n(x) = \Delta_n(x)\Phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right) + \sigma\phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right)$$

Probability of Improvement

$$\mathrm{PI}_n(x) = \phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right)$$

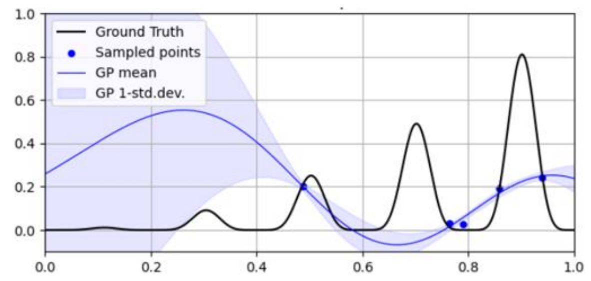
Upper Confidence Bound

$$\mathrm{UCB}_n(x) = \mu_n(x) + \beta^{1/2}\sigma_n(x)$$

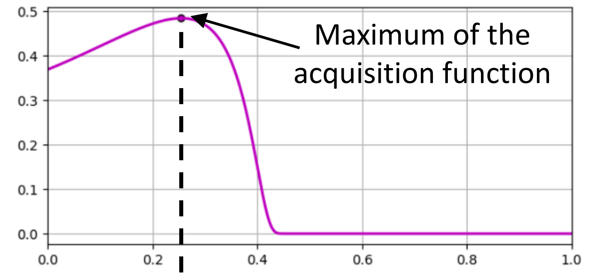
Algorithm:

Step 1. Sample a few initial points from the objective function.

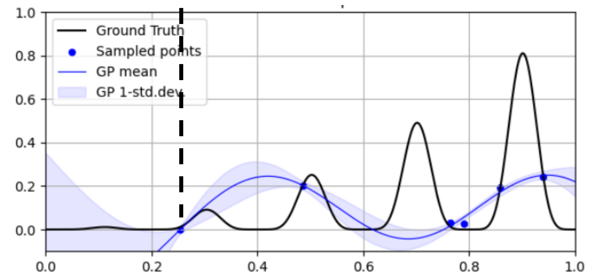
Step 2. Fit a GP surrogate on the current samples.



Step 3. Compute the acquisition function then find its maximum. This is where we should sample next, according to BayesOpt.



Step 4. Sample the objective function at the best point, then go back to **Step 2**.



Bayesian Optimization

Surrogate Model (Gaussian Process)

$$\boxed{\begin{aligned}\text{mean}(y|x^*) &= k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}\boldsymbol{y}\\ \text{var}(y|x^*) &= k(x^*,x^*)+\sigma^2-k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}k(x^*,\boldsymbol{x})\end{aligned}}$$

Acquisition Functions

Expected Improvement

$$\text{EI}_n(x) = \Delta_n(x)\Phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right) + \sigma\phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right)$$

Probability of Improvement

$$\text{PI}_n(x) = \phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right)$$

Upper Confidence Bound

$$\text{UCB}_n(x) = \mu_n(x) + \beta^{1/2}\sigma_n(x)$$

Definitions:

$$\Delta_n(x) = \begin{cases}\mu_n(x) - y_n^* - \xi & \text{if maximization}\\ y_n^* - \mu_n(x) - \xi & \text{if minimization}\end{cases}$$

At the nth iteration:

$\mu_n(x) = \text{mean}(y|x) =$ surrogate value at x

$\sigma_n(x) = \sqrt{\text{var}(y|x)} =$ uncertainty at x (std. dev.)

$y_n^* =$ max/min observed so far (nth iteration)

$\xi =$ exploration/exploitation parameter (higher ξ, more exploration)

$\phi(a) =$ normal cumulative density function (CDF)

$\Phi(a) =$ normal probability density function (PDF)

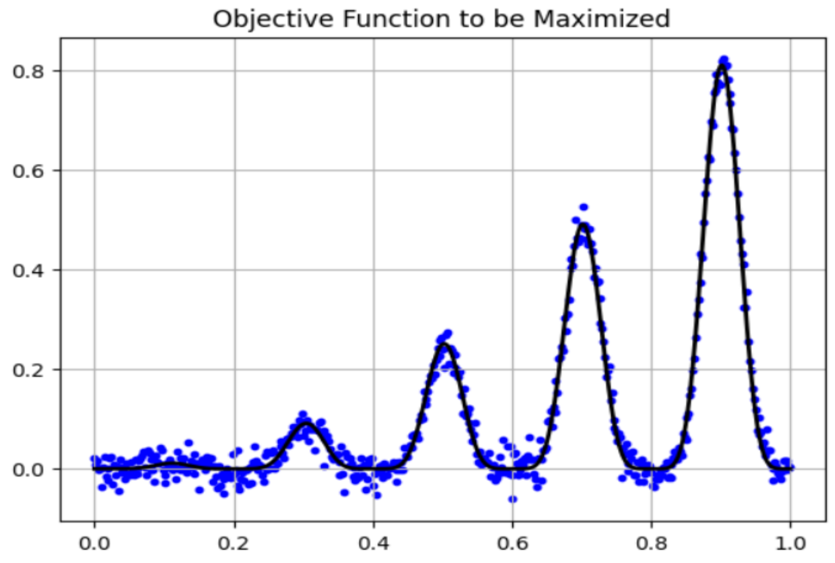
Bayesian Optimization

Example:

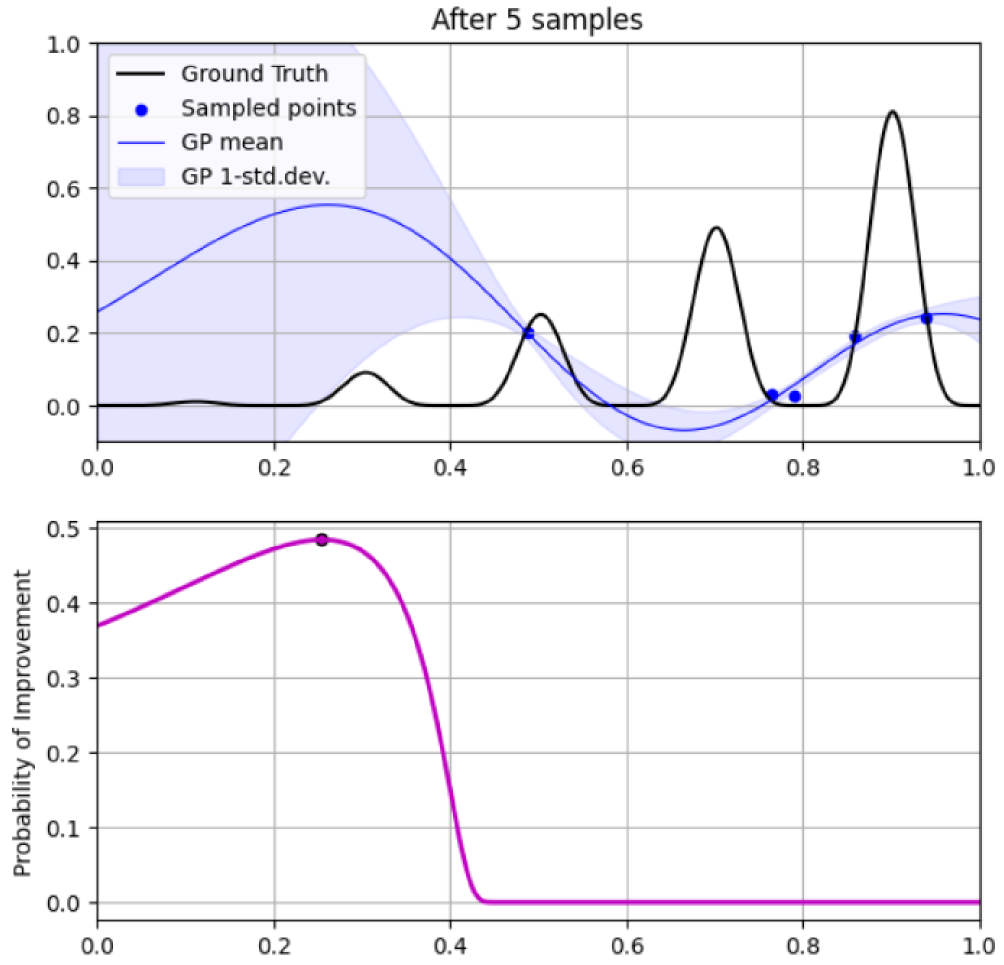
Find x within $x \in [0, 1]$ such that

$$x^2 \sin^6(5\pi x) + \varepsilon$$

is maximum. The noise is distributed as $\varepsilon \sim \mathcal{N}(0, 0.02)$. Use BayesOpt with the *Probability of Improvement* acquisition function.

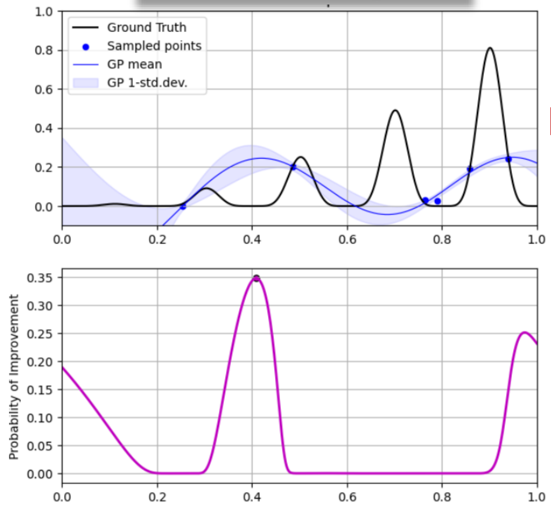


Initialization:

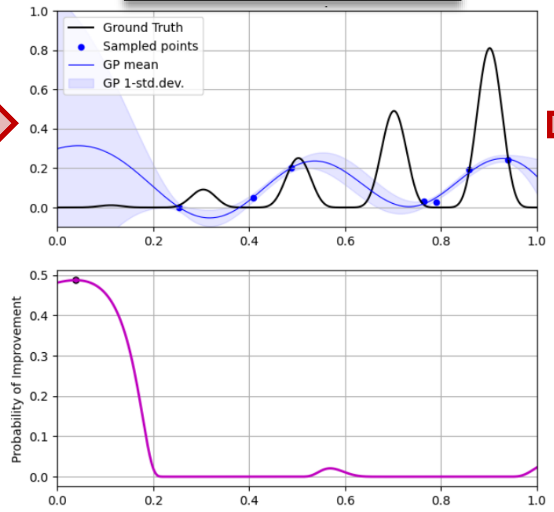


Bayesian Optimization

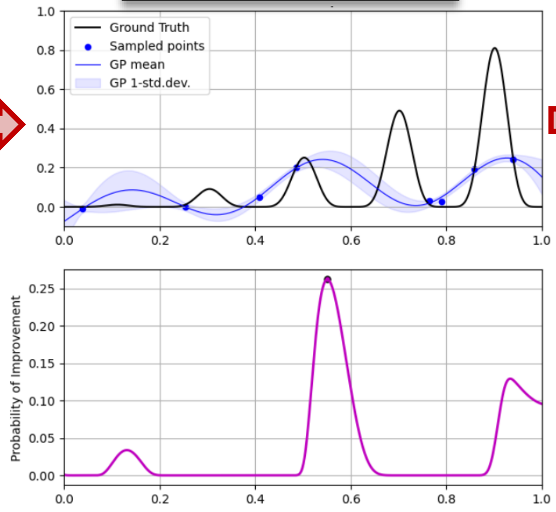
After 6 samples



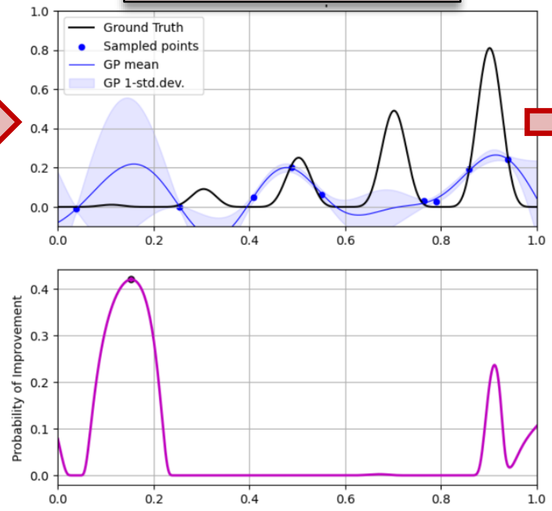
After 7 samples



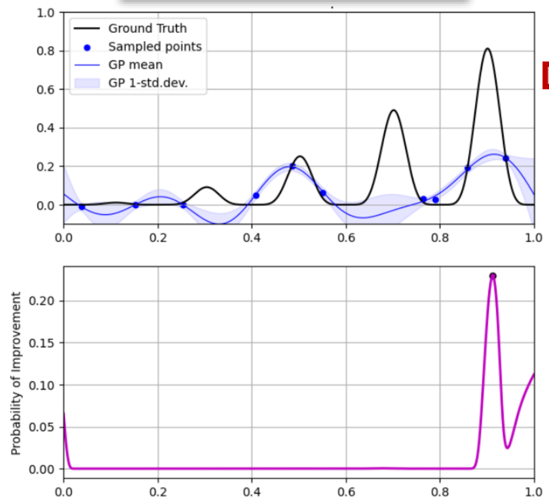
After 8 samples



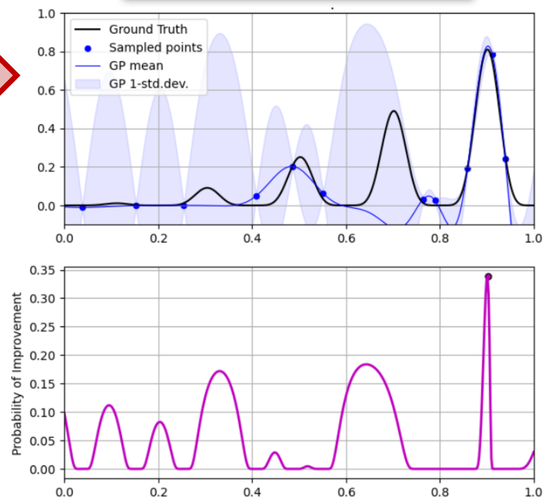
After 9 samples



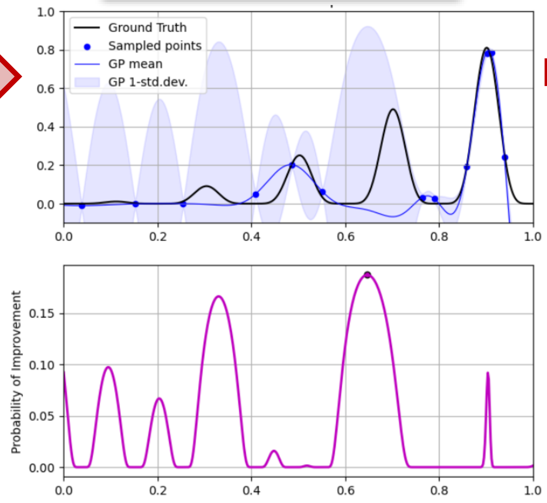
After 10 samples



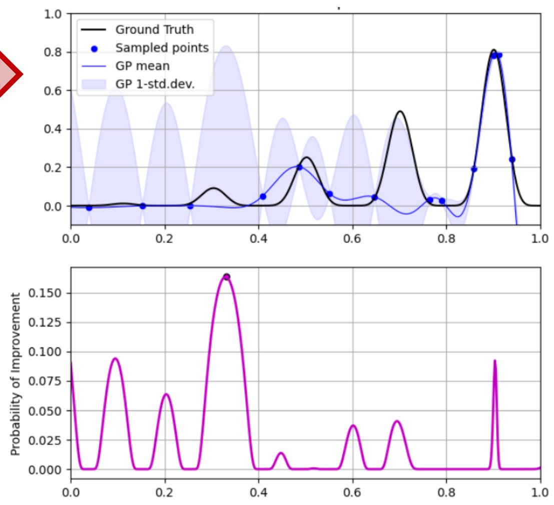
After 11 samples



After 12 samples



After 13 samples



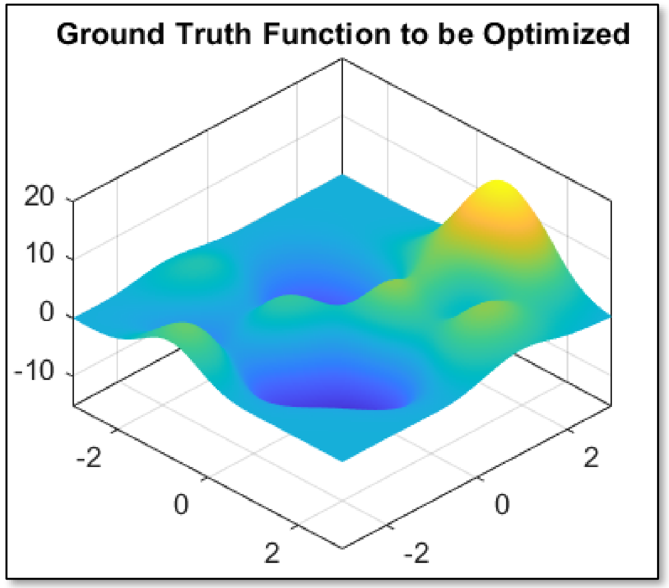
Bayesian Optimization

Example:

Find the point (x_1, x_2) that maximizes an *unknown* objective function given by the following surface.

Use Bayesian Optimization with the Expected Improvement policy and $\xi = 0.2$.

You can only sample the function 50 times.



Answer: Maximum (estimated):

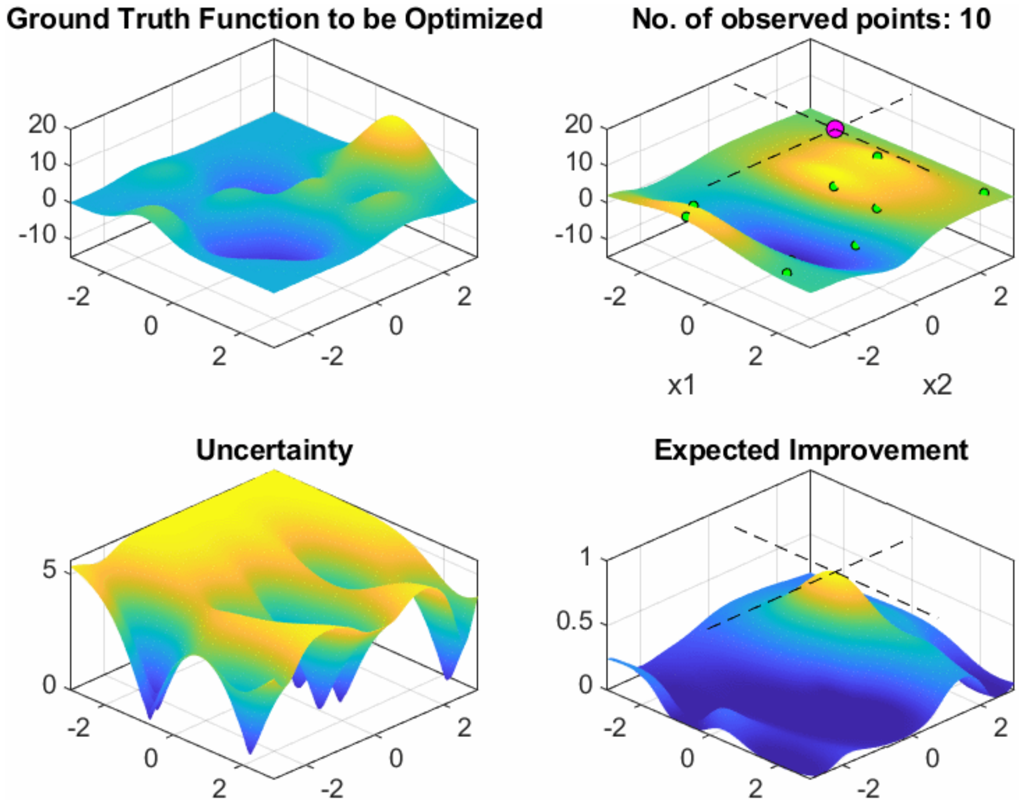
$$(x_1, x_2) = (0.939394, 2.515152)$$
$$f(x_1, x_2) = 16.795755$$

Max point of the surrogate model

Maximum (observed):

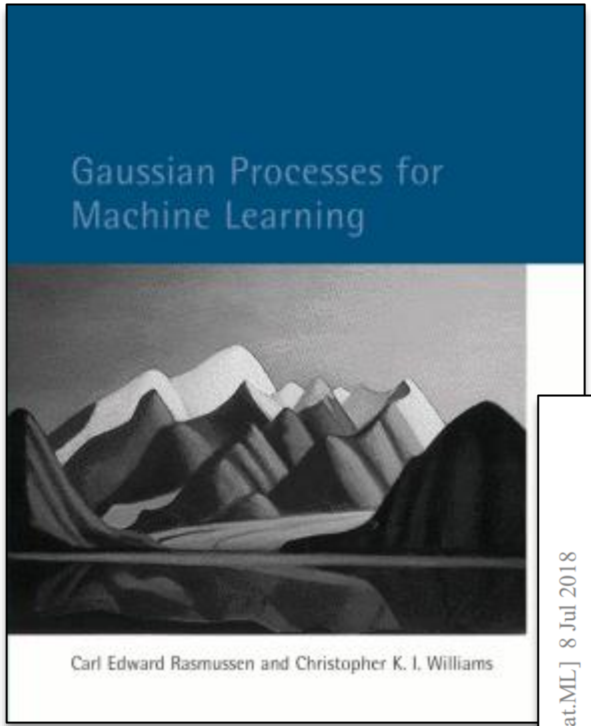
$$(x_1, x_2) = (1.060606, 2.575758)$$
$$f(x_1, x_2) = 16.619848$$

Max point among all 50 samples observed

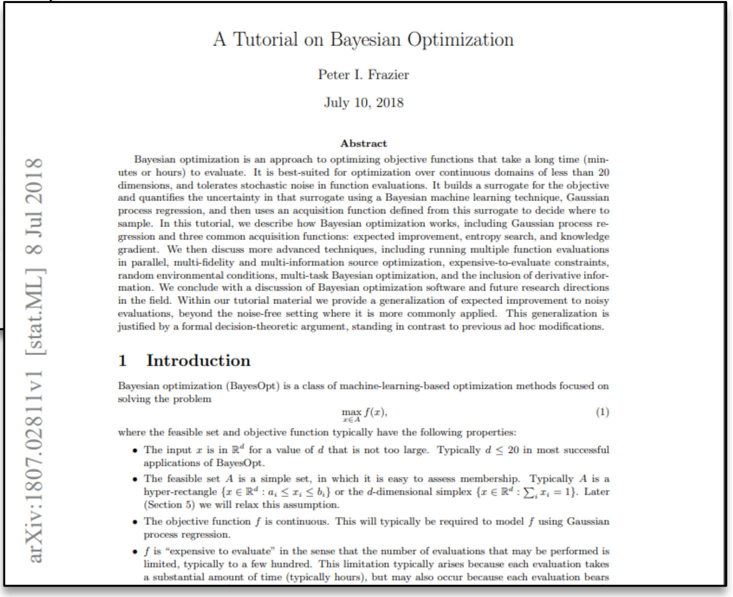


Outline

- Motivation for GPs
- Bayesian Statistics
  - Bayes Theorem
  - Bayesians vs. Frequentists
- Gaussian Processes
  - Derivation
  - Kernels
- Bayesian Optimization
  - Algorithm
  - Acquisition Functions


Gaussian Processes for Machine Learning
Carl Edward Rasmussen and Christopher K. I. Williams

Rasmussen and Williams (2006)
Gaussian Processes for Machine Learning.
MIT Press.
https://gaussianprocess.org/gpml/



A Tutorial on Bayesian Optimization

Peter I. Frazier

July 10, 2018

arXiv:1807.02811v1 [stat.ML] 8 Jul 2018

Abstract

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample. In this tutorial, we describe how Bayesian optimization works, including Gaussian process regression and three common acquisition functions: expected improvement, entropy search, and knowledge gradient. We then discuss more advanced techniques, including running multiple function evaluations in parallel, multi-fidelity and multi-information source optimization, expensive-to-evaluate constraints, random environmental conditions, multi-task Bayesian optimization, and the inclusion of derivative information. We conclude with a discussion of Bayesian optimization software and future research directions in the field. Within our tutorial material we provide a generalization of expected improvement to noisy evaluations, beyond the noise-free setting where it is more commonly applied. This generalization is justified by a formal decision-theoretic argument, standing in contrast to previous ad hoc modifications.

1 Introduction

Bayesian optimization (BayesOpt) is a class of machine-learning-based optimization methods focused on solving the problem

$$\max_{x \in A} f(x), \tag{1}$$

where the feasible set and objective function typically have the following properties:

- The input x is in $\mathbb{R}^d$ for a value of d that is not too large. Typically $d \leq 20$ in most successful applications of BayesOpt.
- The feasible set A is a simple set, in which it is easy to assess membership. Typically A is a hyper-rectangle $\{x \in \mathbb{R}^d : a_i \leq x_i \leq b_i\}$ or the d-dimensional simplex $\{x \in \mathbb{R}^d : \sum_i x_i = 1\}$. Later (Section 5) we will relax this assumption.
- The objective function f is continuous. This will typically be required to model f using Gaussian process regression.
- f is "expensive to evaluate" in the sense that the number of evaluations that may be performed is limited, typically to a few hundred. This limitation typically arises because each evaluation takes a substantial amount of time (typically hours), but may also occur because each evaluation bears

Frazier (2018). A Tutorial on Bayesian Optimization
https://arxiv.org/pdf/1807.02811.pdf

Further Reading

- C.E. Rasmussen and C. K. I. Williams. Gaussian Processes for Machine Learning. The MIT Press, 2006. ISBN 0-262-18253-X.
- https://www.cs.toronto.edu/~duvenaud/cookbook/
- Deringer et al. (2021). Gaussian Process Regression for Materials and Molecules. Chemical Reviews. doi: 10.1021/acs.chemrev.1c00022.
- Duvenaud (2014). Automatic Model Construction with Gaussian Processes. PhD Thesis. https://www.cs.toronto.edu/~duvenaud/thesis.pdf
- Roman Garnett (2023). Bayesian Optimization. Cambridge University Press. https://bayesoptbook.com/
- Pilario et al. (2022). Predicting Drying Curves in Algal Biorefineries using Gaussian Process Autoregressive Models. Digital Chemical Engineering, vol. 4, 100036. https://doi.org/10.1016/j.dche.2022.100036
- Shahriari et al. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE. Doi: 10.1109/JPROC.2015.2494218
- Frazier, Wang (2015). Bayesian Optimization for Materials Design. https://arxiv.org/pdf/1506.01349.pdf
- Micchelli et al. (2006). **Universal Kernels**. Journal of Machine Learning Research 7, pp. 2651-2667.
- Jakkala (2021). Deep Gaussian Processes: A Survey. https://arxiv.org/pdf/2106.12135.pdf
- Neal, R. (1996). Bayesian Learning for Neural Networks. Lecture Notes in Statistics. Springer New York.
- Titsias, M. and Lawrence, N. D. (2010). Bayesian Gaussian Process Latent Variable Model. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, volume 9, pages 844–851, Chia Laguna Resort, Sardinia, Italy. JMLR Workshop and Conference Proceedings.
- Cowen-Rivers et al. (2022). HEBO: Pushing The Limits of Sample-Efficient Hyper-parameter Optimisation. Journal of Artificial Intelligence Research. https://doi.org/10.1613/jair.1.13643