Zero-Order Methods

Derivative-Free Search, Bayesian Optimization, Surrogate Modeling

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

Outline

- Introduction to NLP
- Necessary and Sufficient Conditions for Optimality
- Convex Programming
- Methods for Solving NLP
    - One-Dimensional, Unconstrained NLP
    - Multivariable, Unconstrained NLP
    - **Zero-order**, First-order, Second-order Methods
    - Constrained NLP

A Taxonomy of NLP Solvers

Zero-order Methods

First-order Methods

Second-order Methods

Derivative-free, black-box

- Grid Search / Exhaustive Search
- Random Search
- Nelder-Mead Simplex

- Metaheuristic Search
  - Genetic Algorithms
  - Particle Swarm
  - Simulated Annealing
  - Differential Evolution
  - CMAES

- **Bayesian Optimization / Surrogate-based**

- **Zero-order methods** are good for objective functions whose
  - exact expression is unknown, or
  - it is known but hard / impossible to differentiate.

- If we know the exact expression and it is differentiable, then it is better to use first-order / second-order methods.

- **Scenarios where zero-order methods are useful:**
  - Design of Experiments → Self-driving labs!
  - Fast prototyping of a product / material design → Materials Discovery!
  - Optimization of machine learning hyper-parameters or architectures
  - Surrogate optimization in chemical plants

ve information

d (BFGS)

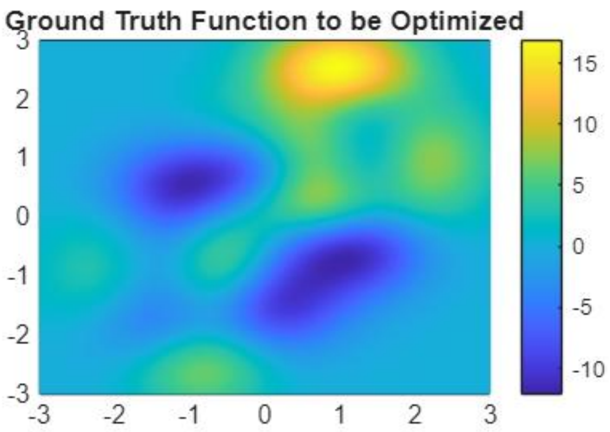
t

Methods

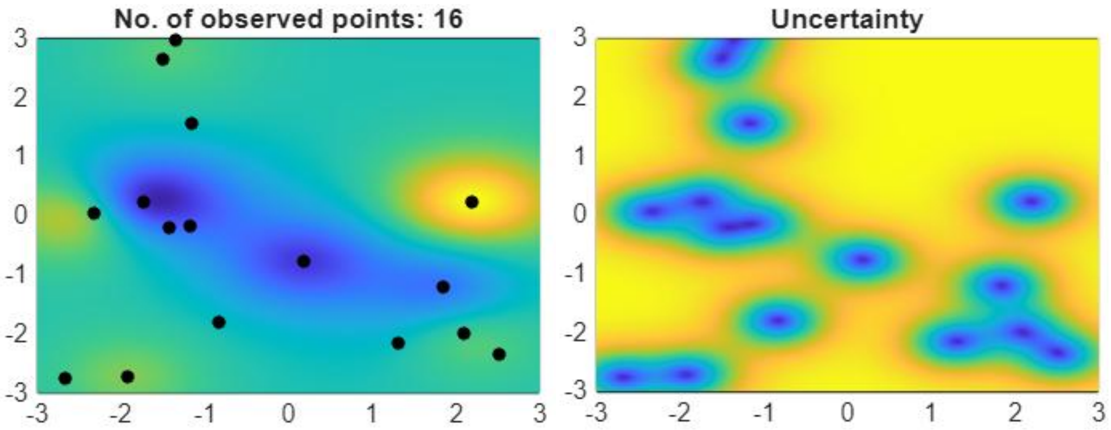
r-Point

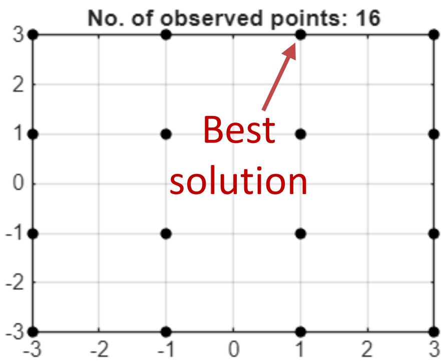
Zero-Order Methods

Find the maximum point in this surface. Assume it is unknown and you can only sample it 50x.

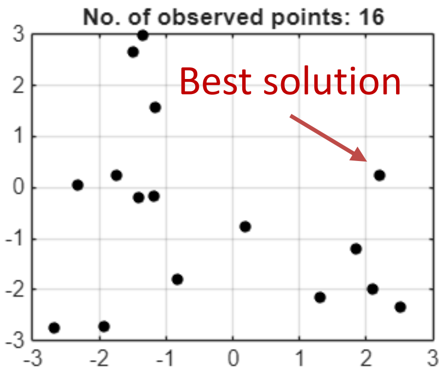


[Step 1] Surrogate Modelling





Grid Search



Random Search

Bayesian Optimization

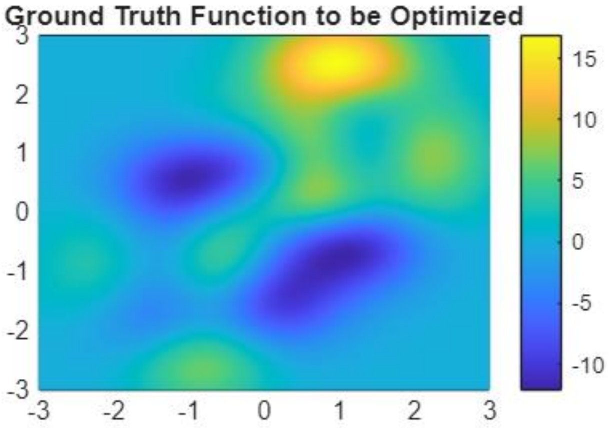
- Find next trials that either:
  - Explores unknown regions, or;
  - Exploits the best regions.
- Need: *Surrogate model* + *Acquisition function*

Evolutionary Search

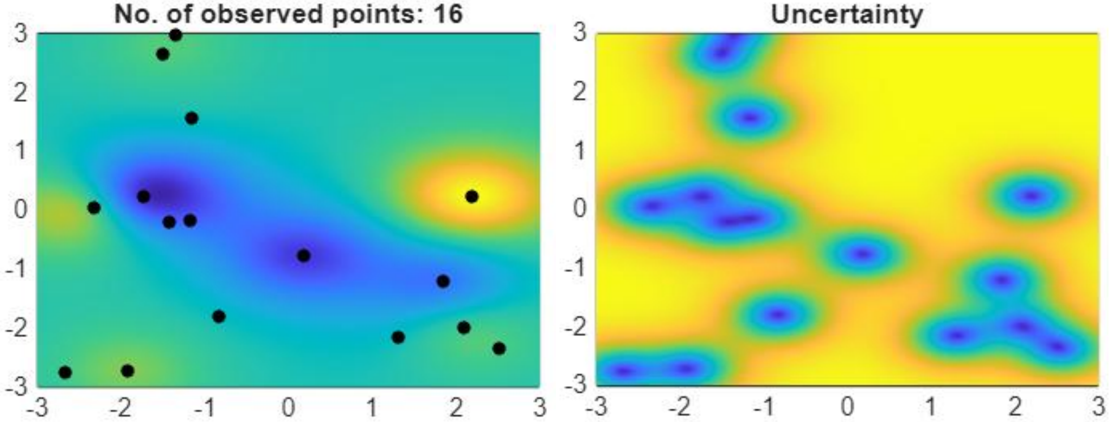
- Takes N x G no. of more samples
- N = no. of iterations
- G = population size

Bayesian Optimization

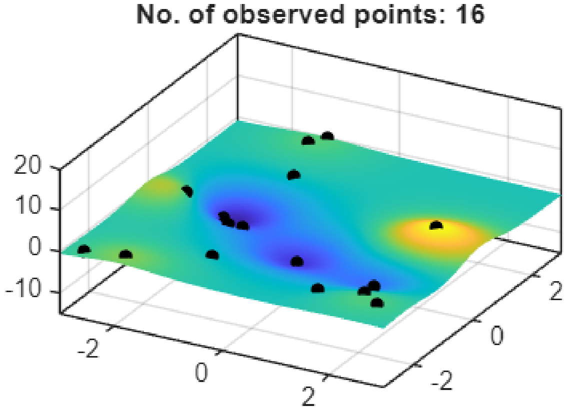
Find the maximum point in this surface. Assume it is unknown and you can only sample it 50x.

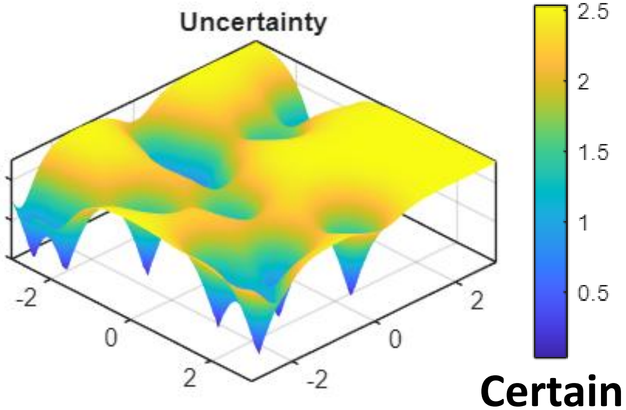


[Step 1] Surrogate Modelling



In another viewpoint…





Uncertain

Certain

Mean estimate

Uncertainty estimate

Surrogate Model

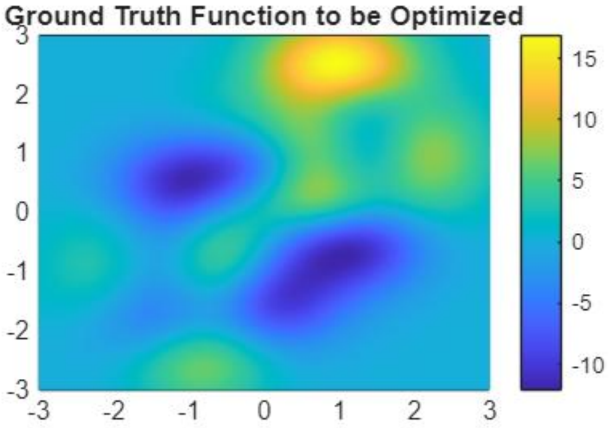
- Any regression model that can output a *mean* and *uncertainty* estimate over the search space.
- A **proxy** for the unknown *objective function* that is *sequentially fitted* to new samples using **Bayesian inference**.
- Typically, **Gaussian process regression** is used:

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

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

Bayesian Optimization

Find the maximum point in this surface. Assume it is unknown and you can only sample it 50x.

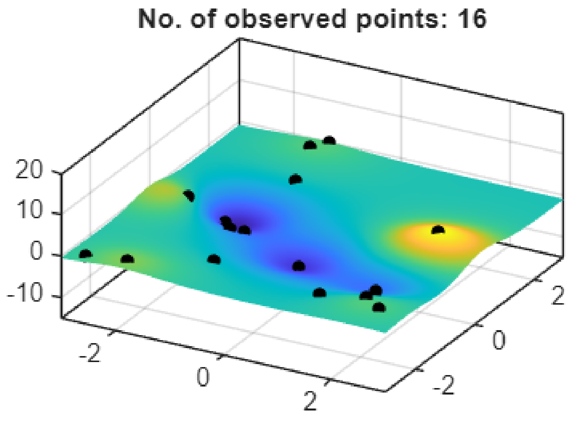


Acquisition Function

- A **policy** that evaluates the surrogate model output to find the best place to sample the objective function **next**.
- The next best trial is where the policy surface is **maximum**.
- Typically, **Upper Confidence Bound** (**UCB**) is used:

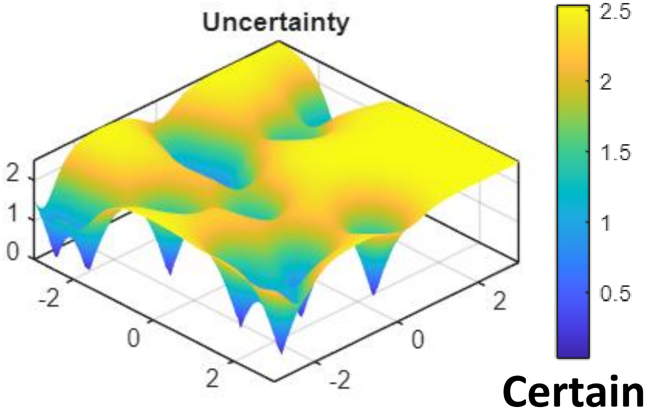
$$\mathrm{UCB}_n(x) = \underbrace{\mu_n(x)}_{\textit{Mean estimate}} + \beta^{1/2}\underbrace{\sigma_n(x)}_{\textit{Uncertainty estimate}}$$

[Step 1] Surrogate Modelling



Mean estimate

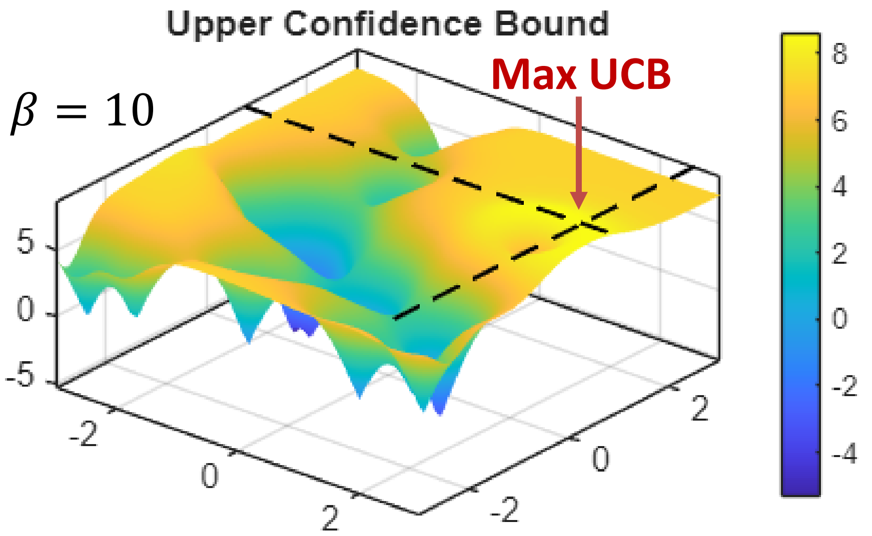
Uncertain



Certain

Uncertainty estimate

[Step 2] Calculate Acquisition Function



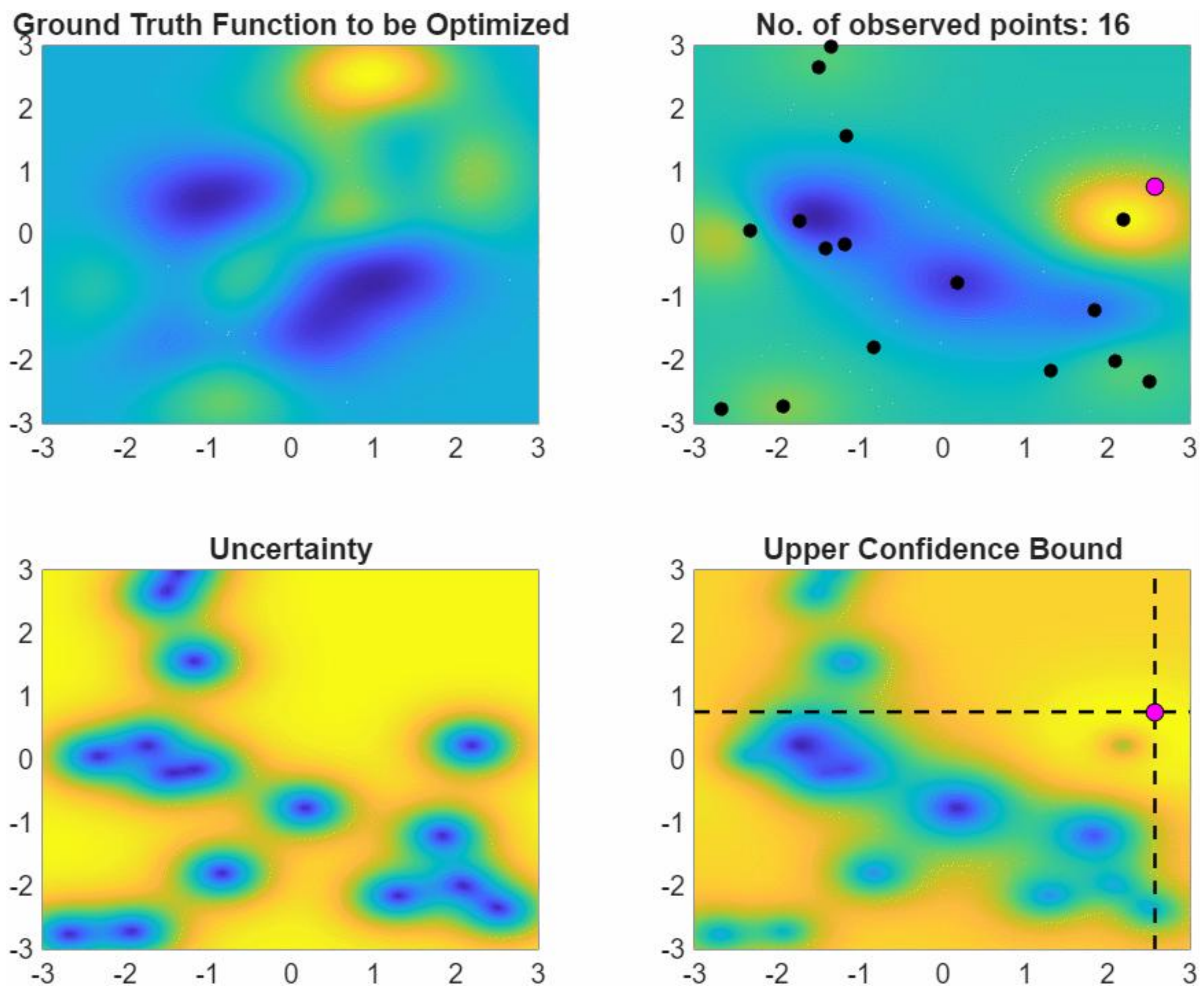
Bayesian Optimization

Find the maximum point in this surface. Assume it is unknown and you can only sample it 50x.

Observations:

1. Maximum already found in the 30th trial.
2. Non-promising regions may not be explored anymore.
3. Next best trial is guided by knowledge from all past trials. **Sample-efficient!**

Solution: (Top View)



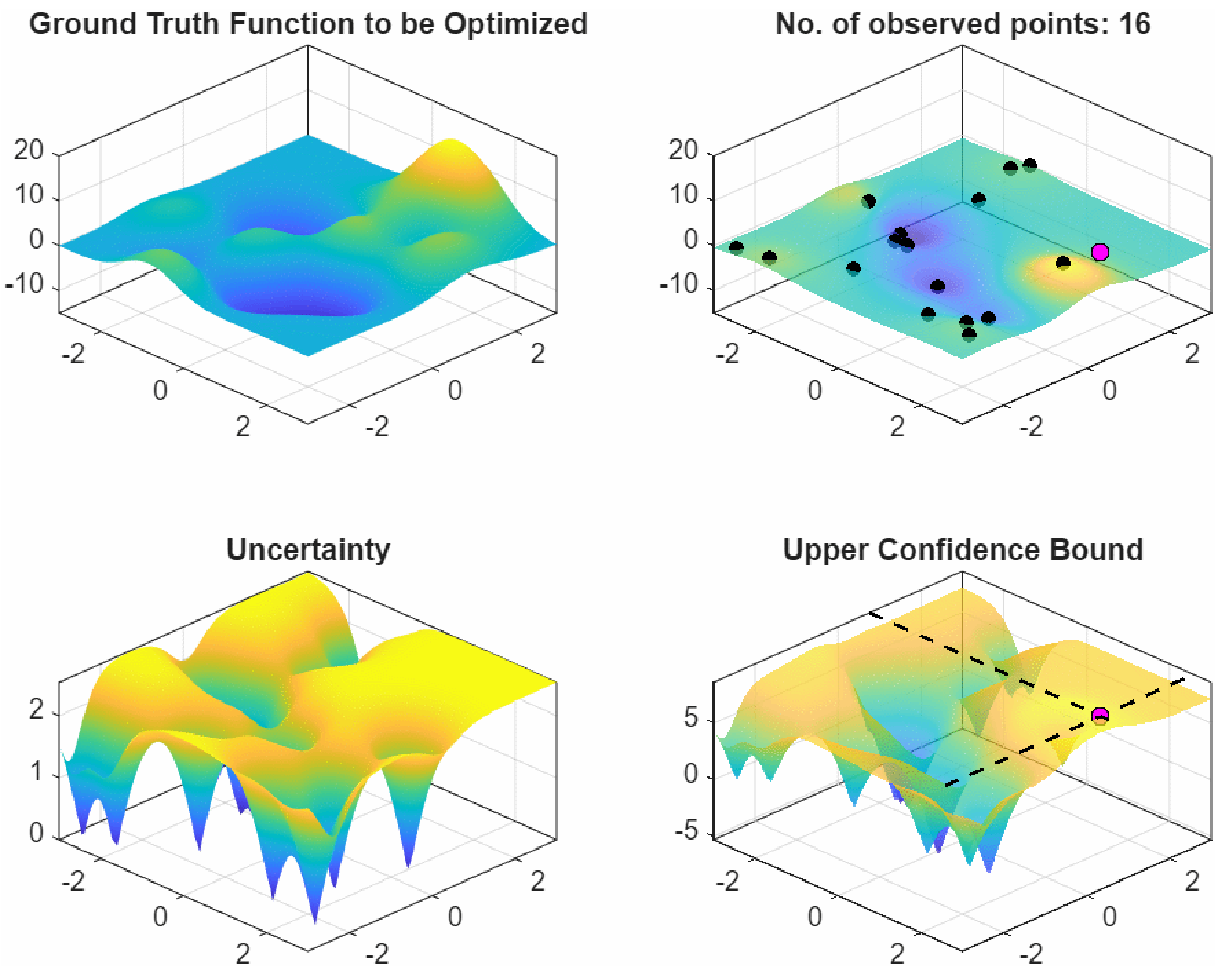
Bayesian Optimization

Solution: **(Side View)**

Find the maximum point in this surface. Assume it is unknown and you can only sample it 50x.

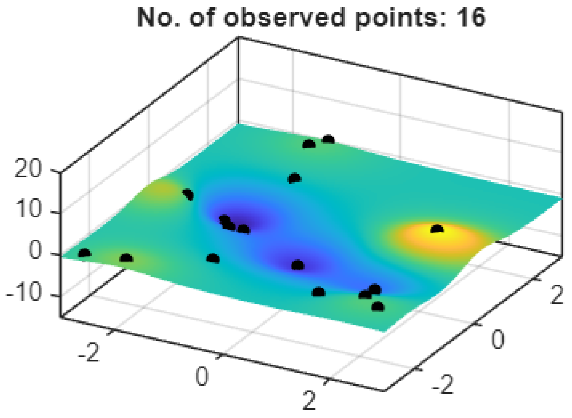
Observations:

1. Maximum already found in the 30^{th} trial.
2. Non-promising regions may not be explored anymore.
3. Next best trial is guided by knowledge from all past trials. **Sample-efficient!**



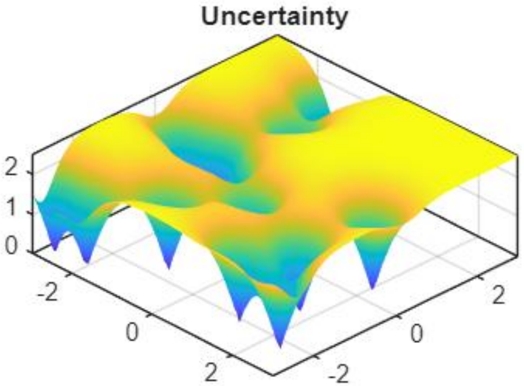
Bayesian Optimization: Acquisition Functions

Surrogate Model



Mean estimate

$\mu_n(x)$



Uncertainty estimate

$\sigma_n(x)$

Acquisition Function

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

Policy:
Sample where UCB is maximum.

High $\mu_n(x)$
Exploitation

High $\sigma_n(x)$
Exploration

β = exploration / exploitation parameter

Other Acquisition Functions:

Upper Confidence Bound

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

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

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

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:

$\mu_n(x) = \mathrm{mean}(y|x) =$ surrogate mean estimate at x

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

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

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

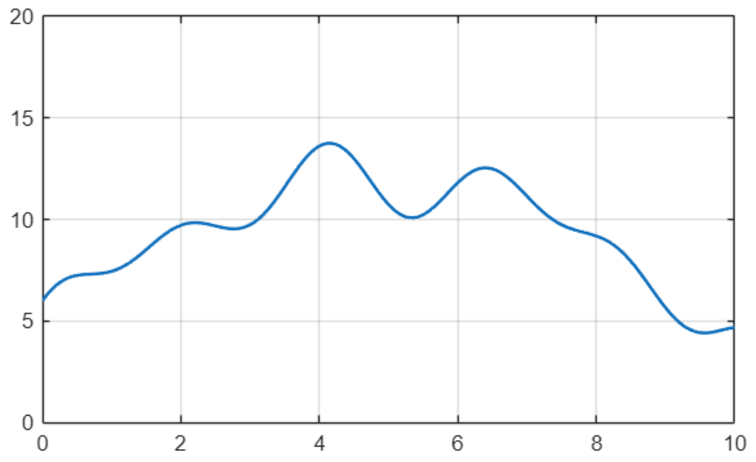
$\phi(\cdot) =$ normal cumulative density function (CDF)

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

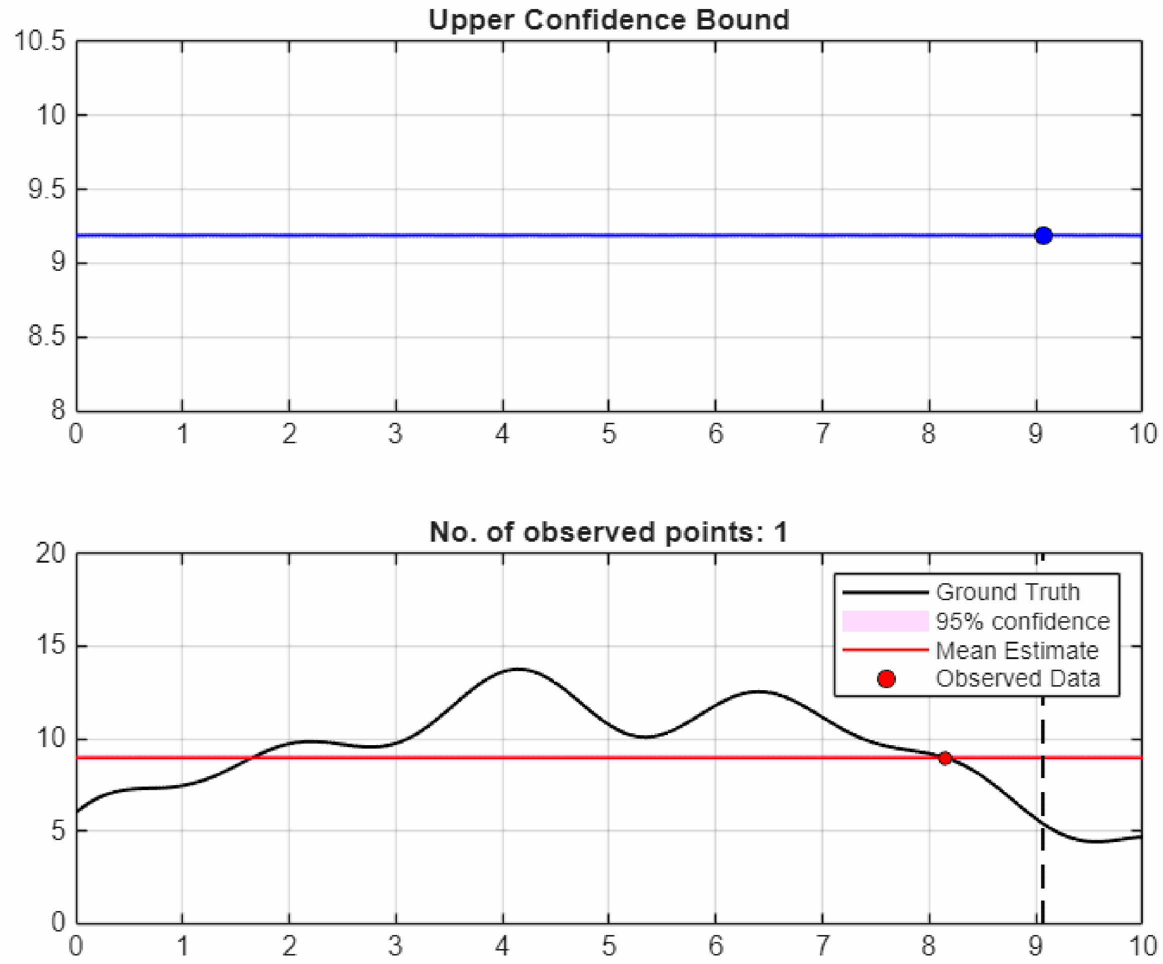
Bayesian Optimization

Find the maximum point in this unknown function within [0, 10] using only 20 trials.

- Start from only 1 random trial.
- Use the **UCB** with $\beta = 1$.



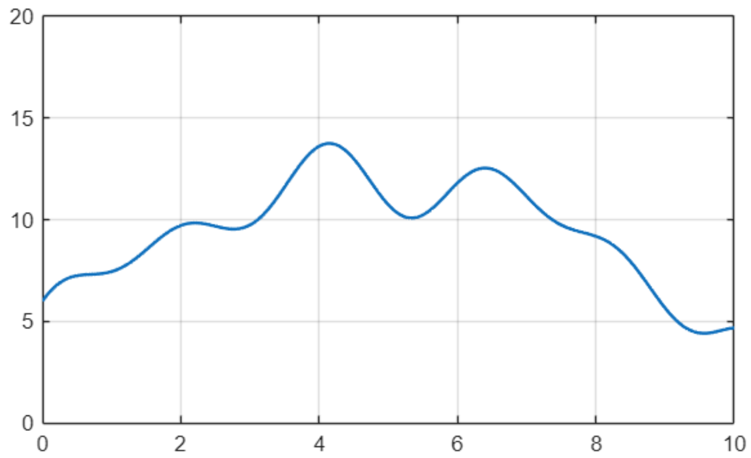
Solution:



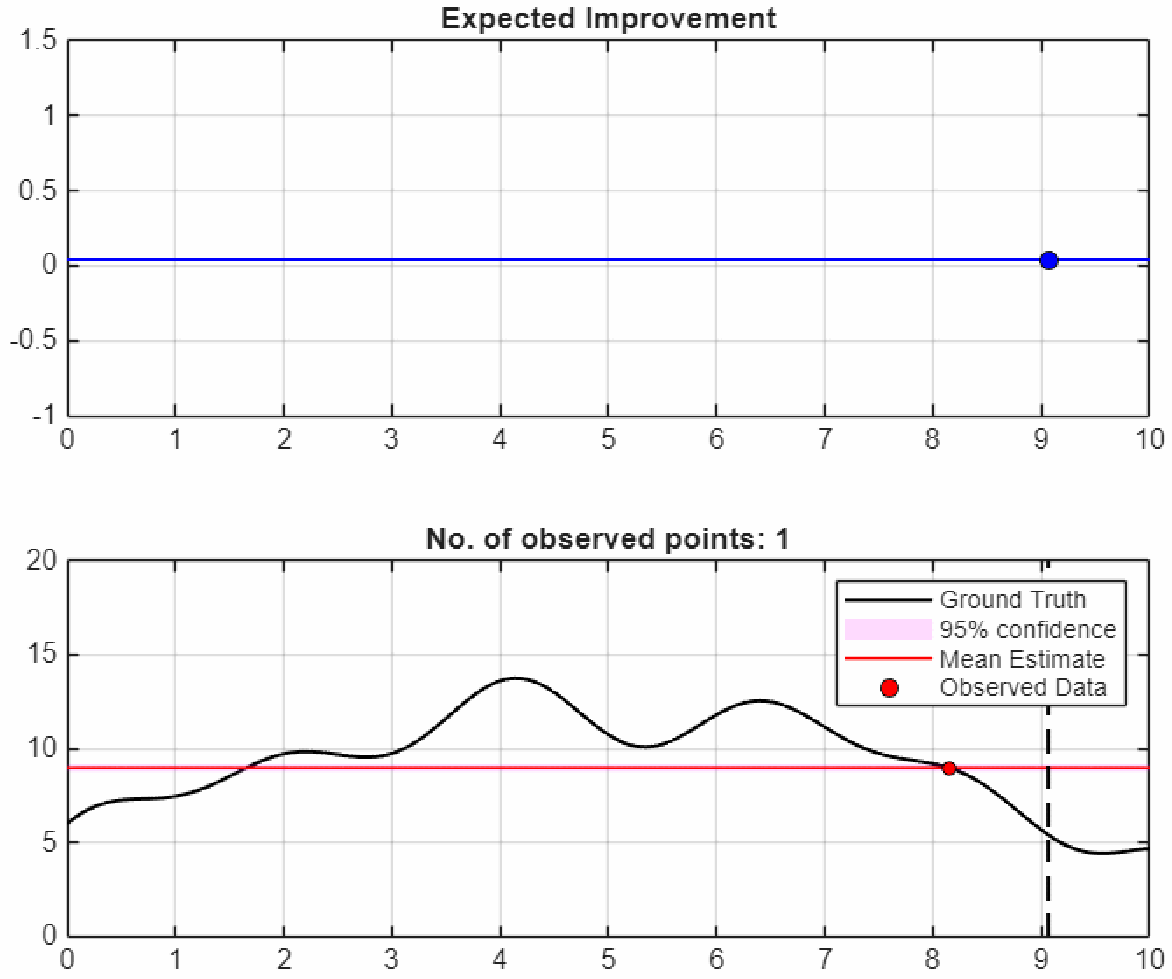
Bayesian Optimization

Find the maximum point in this unknown function within [0, 10] using only 20 trials.

- Start from only 1 random trial.
- Use the **Expected Improvement** Policy with $\xi = 0$.



Solution:



Bayesian Optimization: Gaussian Processes

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

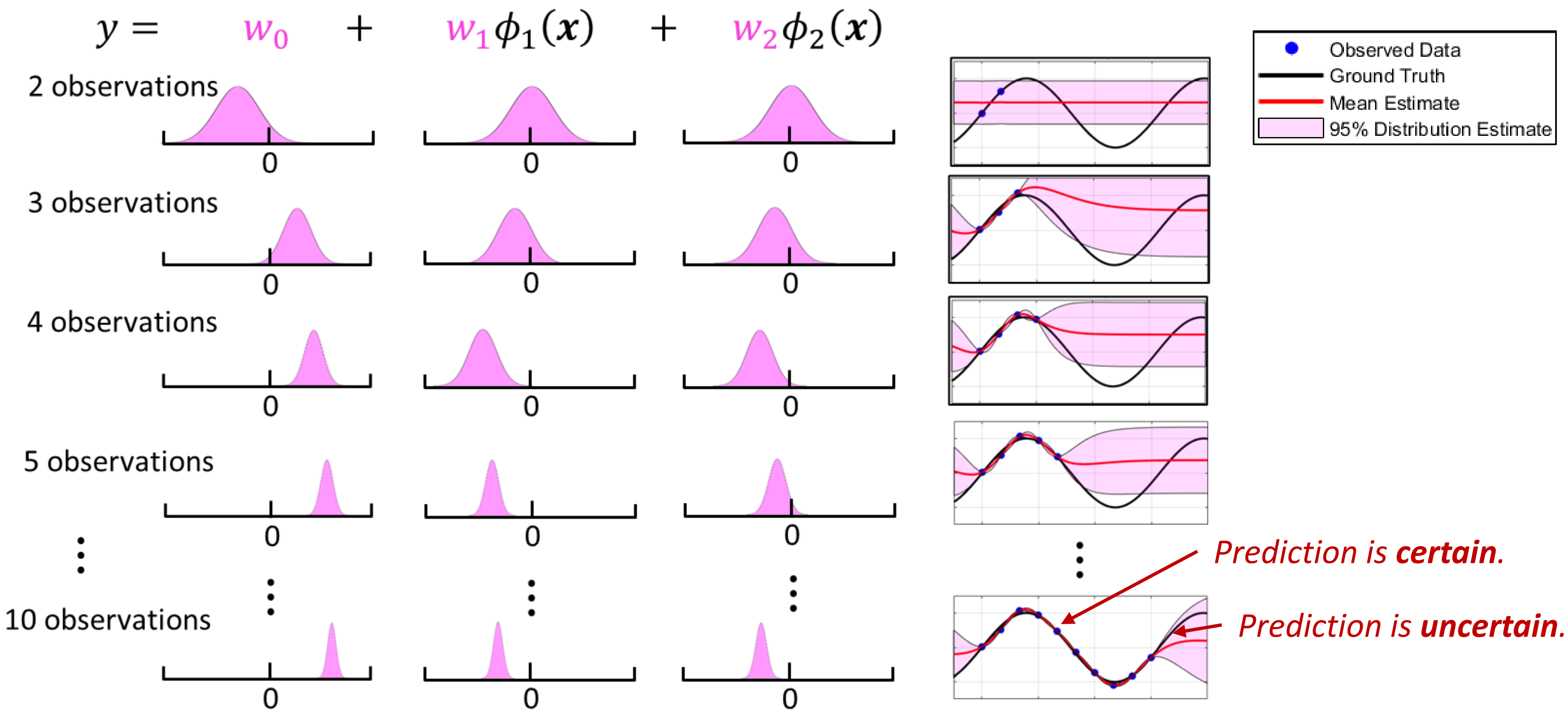
Gaussian Process Regression (GPR):

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

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

Derived using Bayes Theorem.

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



Bayesian Optimization: Gaussian Processes

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

Gaussian Process Regression (GPR):

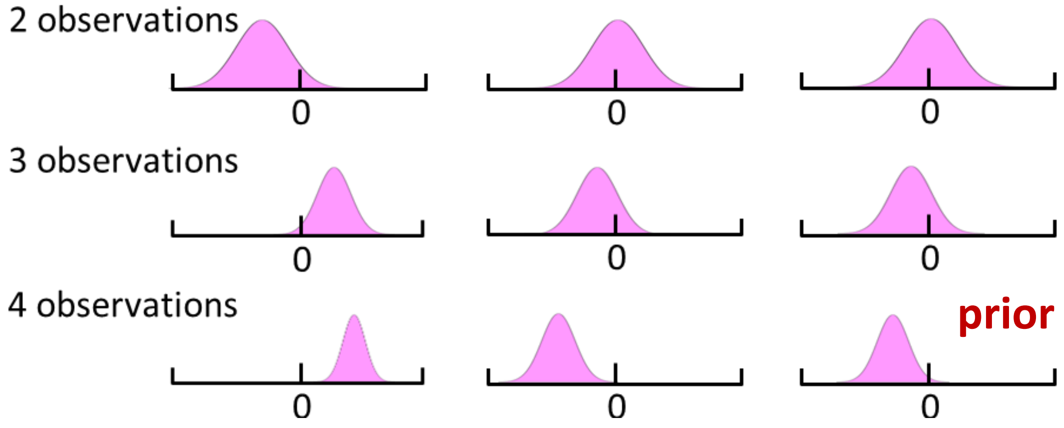
$$\boxed{\mathrm{mean}(y^*|x^*) = k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}\boldsymbol{y}}$$

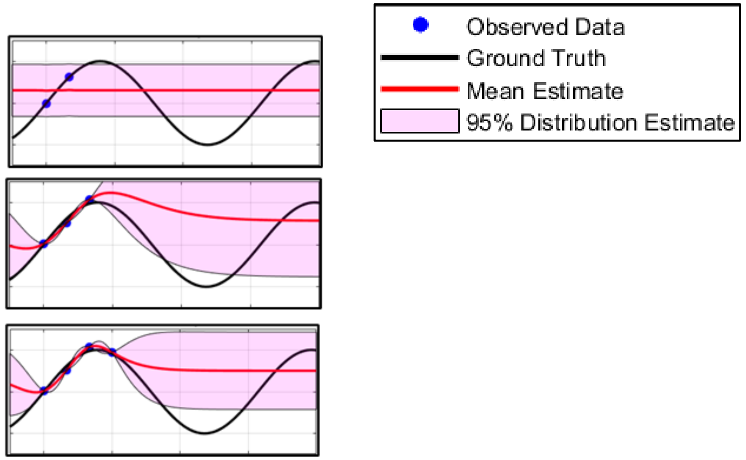
$$\boxed{\mathrm{var}(y^*|x^*) = k(x^*,x^*)+\sigma^2 - k(x^*,\boldsymbol{x})^T[\boldsymbol{K}+\sigma^2\boldsymbol{I}]^{-1}k(x^*,\boldsymbol{x})}$$

Derived using Bayes Theorem.

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

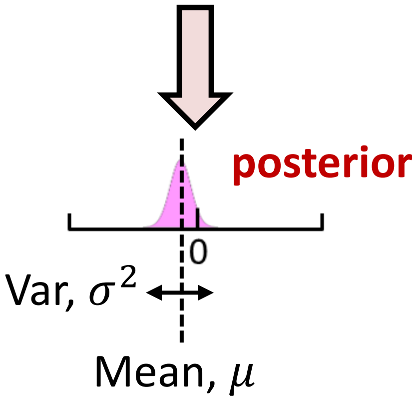
$$y = w_0 + w_1\phi_1(\boldsymbol{x}) + w_2\phi_2(\boldsymbol{x})$$





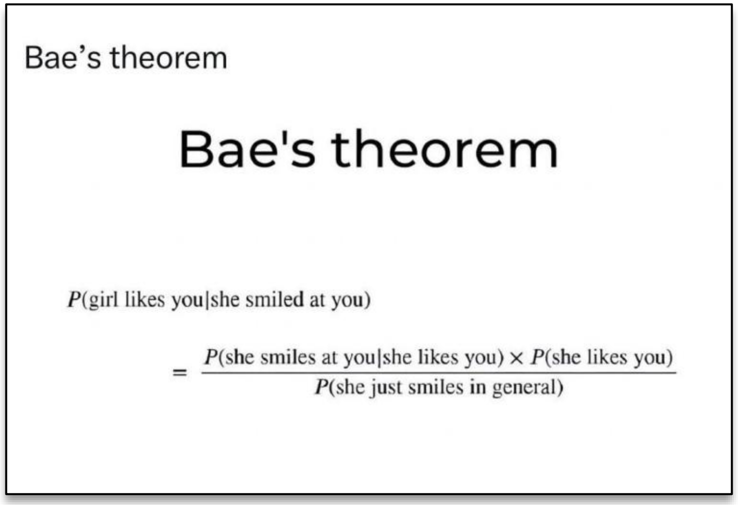
Bayes' Theorem:

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



- In GPR, the prior, posterior, and likelihood are all **Gaussian** distributed.
- If they are not assumed Gaussian, Bayesian inference may be intractable.

Bayesian Statistics: A Change of Perspective



Bayes' Theorem:

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

Bae's Theorem:

$$p(\text{girl likes you}|\text{she smiled at you}) = \frac{p(\text{she smiled at you}|\text{she likes you})\ \ p(\text{she likes you})}{P(\text{she just smiles in general})}$$

Bayesian Modeling:

Given what I just observed, how likely is this new model?

If this model were true, how likely is it that I would see this new observation?

Before observing anything new, how plausible did I think my current model was?

$$p(\text{model}|\text{observation}) = \frac{p(\text{observation}|\text{model})\ \ p(\text{model})}{P(\text{observation})}$$

Across all models I thought were possible, how likely was it to see this new observation?

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.

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

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

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

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

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:

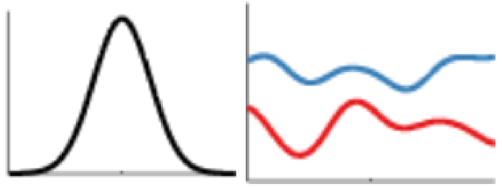
$$\frac{\partial}{\partial\theta_i}\ln p(\mathrm{y}|\boldsymbol{\theta}) = -\frac{1}{2}\mathrm{tr}\left(\mathrm{C}^{-1}\frac{\partial \mathrm{C}}{\partial\theta_i}\right) + \frac{1}{2}\mathrm{y}^T\mathrm{C}^{-1}\frac{\partial \mathrm{C}}{\partial\theta_i}\mathrm{C}^{-1}\mathrm{y}$$

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

Kernel Functions in GPR:

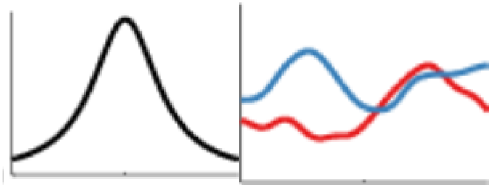
Squared Exponential

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



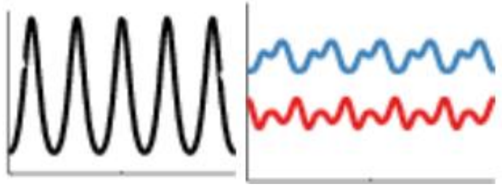
Rational Quadratic

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



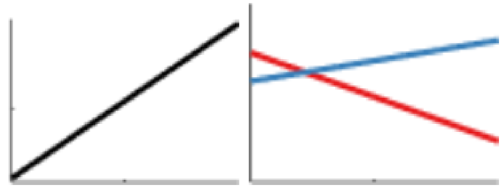
Periodic

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



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

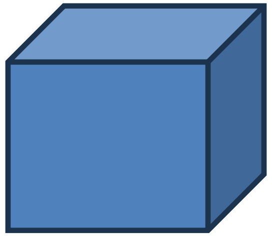
Bayesian Optimization: Initial Sampling

Depending on the constraints, various initial sampling methods are available.

Independent Uniform Sampling

Used when we have box constraints:

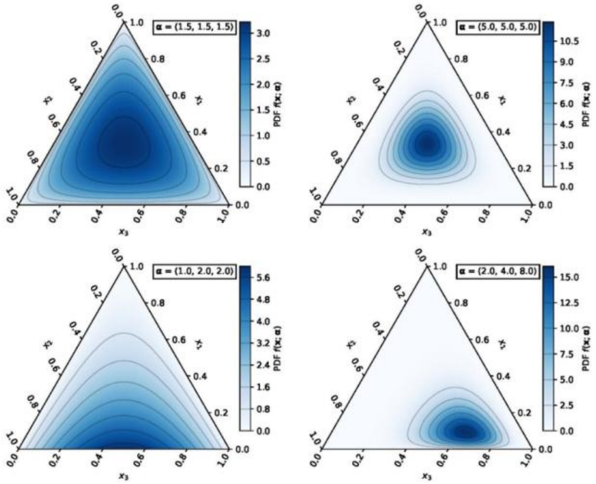
$$0 \leq x_1 \leq 1$$
$$0 \leq x_2 \leq 1$$
$$0 \leq x_3 \leq 1$$



Dirichlet Sampling

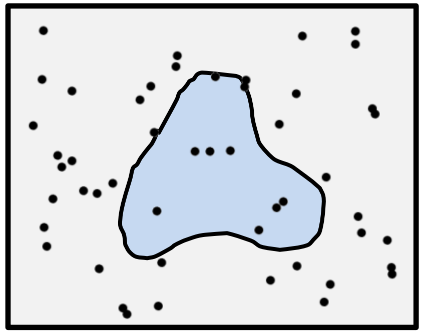
Used when the constraint is a regular simplex:

$$0 \leq x_1 \leq 1$$
$$0 \leq x_2 \leq 1$$
$$0 \leq x_3 \leq 1$$
$$x_1 + x_2 + x_3 = 1$$

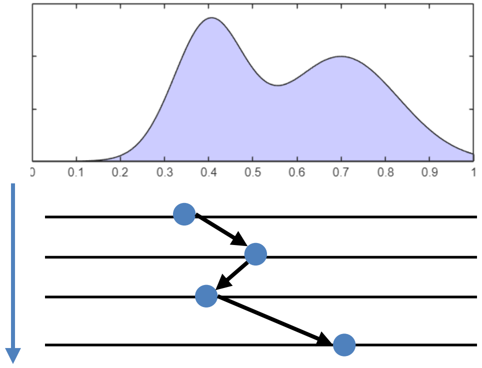


Rejection Sampling

When constraints are nonlinear, just sample anywhere and reject if the constraints are violated.



Markov Chain Monte Carlo (MCMC)



Specify a probability distribution, then draw samples using Markov chains, e.g. Metropolis-Hastings algorithm.

Reference: https://en.wikipedia.org/wiki/Dirichlet_distribution

Bayesian Optimization: Summary

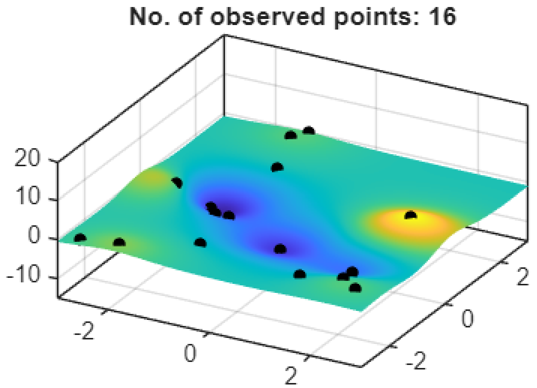
Algorithm:

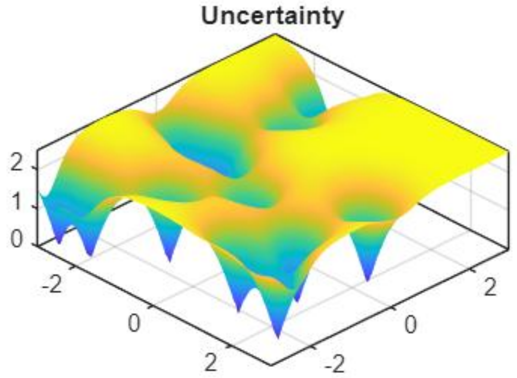
Step 1. Sample a few initial points from the objective function.

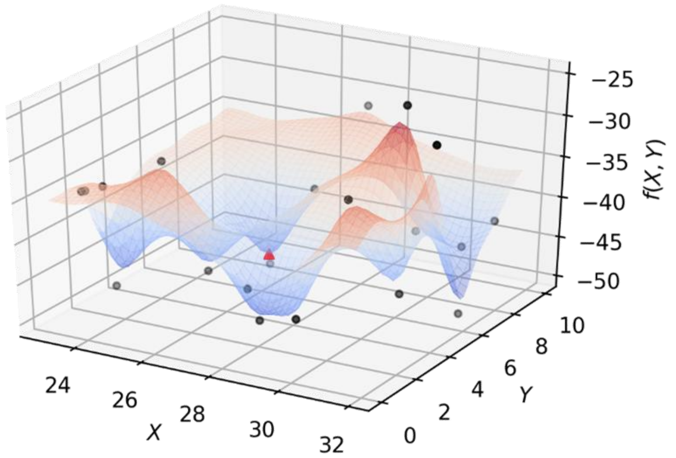
Step 2. Fit a surrogate on the current samples.

Step 3. Compute the acquisition function then find its maximum.

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







Surrogate-based Optimization

- The idea of replacing a hard-to-evaluate objective function with a surrogate model which is faster to evaluate.
- Initially, surrogate model is trained on a few samples of the objective function.
- Surrogate can be updated with more samples.

Reference: https://sksurrogate.readthedocs.io/en/latest/surrogate.html

A Taxonomy of NLP Solvers

Zero-order Methods

Derivative-free, black-box

- Grid Search / Exhaustive Search
- Random Search
- Nelder-Mead Simplex

- Metaheuristic Search
  - Genetic Algorithms
  - Particle Swarm
  - Simulated Annealing
  - Differential Evolution
  - CMAES

- **Bayesian Optimization / Surrogate-based**

First-order Methods

- **Zero-order methods** are good for objective functions whose
  - exact expression is unknown, or
  - it is known but hard / impossible to differentiate.

- If we know the exact expression and it is differentiable, then it is better to use first-order / second-order methods.

- **Scenarios where zero-order methods are useful:**
  - Design of Experiments → Self-driving labs!
  - Fast prototyping of a product / material design → Materials Discovery!
  - Optimization of machine learning hyper-parameters or architectures
  - Surrogate optimization in chemical plants

Second-order Methods

ve information

d (BFGS)

t

Methods

r-Point