A tutorial on Bayesian Optimization

Daniel Hernández–Lobato
Computer Science Department
Universidad Autónoma de Madrid

http://dhnzl.org, daniel.hernandez@uam.es

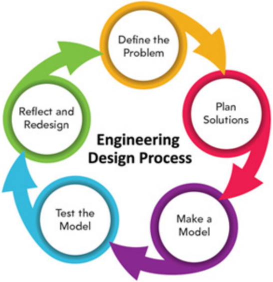
Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!

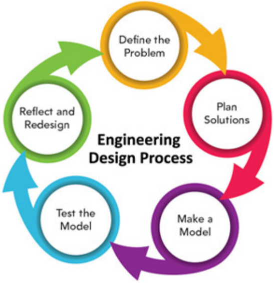
Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!



Challenges in Engineering Design

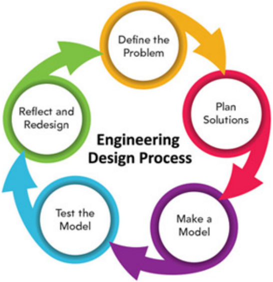
The society demands new products of better quality, functionality, usability, etc.!



- Many choices at each step.

Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!

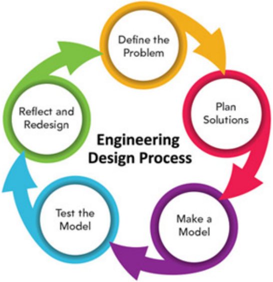


- Many choices at each step.
- Complicated and high dimensional.

Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!

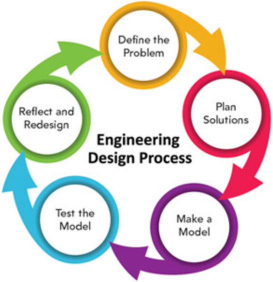


- Many choices at each step.
- Complicated and high dimensional.
- Difficult for individuals to reason about.

Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!

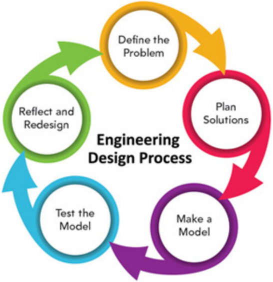


- Many choices at each step.
- Complicated and high dimensional.
- Difficult for individuals to reason about.
- Prone to human bias.

Challenges in Engineering Design

The society demands new products of better quality, functionality, usability, etc.!

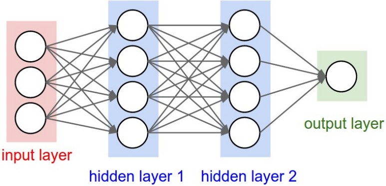
   

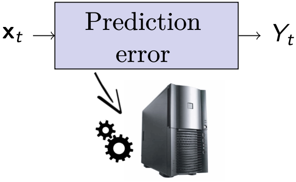


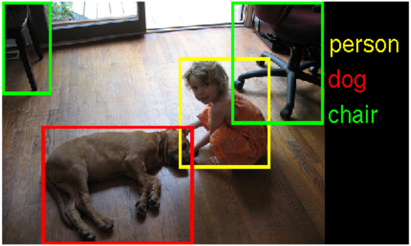
- Many choices at each step.
- Complicated and high dimensional.
- Difficult for individuals to reason about.
- Prone to human bias.

Optimization is a challenging task in new products design!

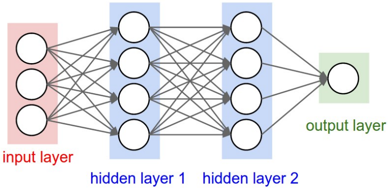
Example: **Deep Neural Network** for object recognition.

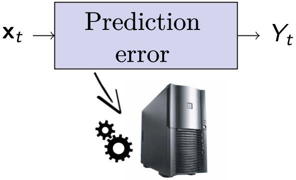


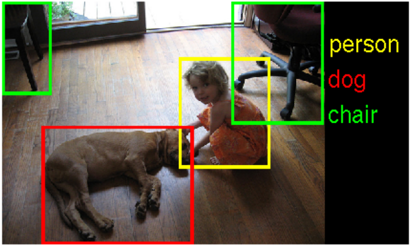




Example: **Deep Neural Network** for object recognition.

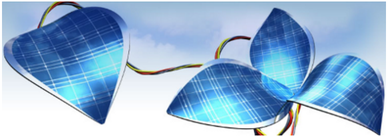


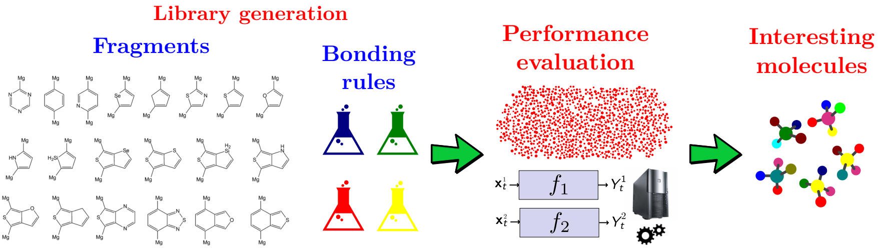




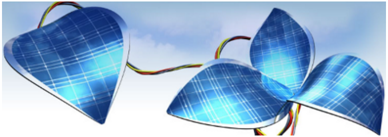
Parameters to tune: Number of neurons, number of layers, learning-rate, level of regularization, momentum, etc.

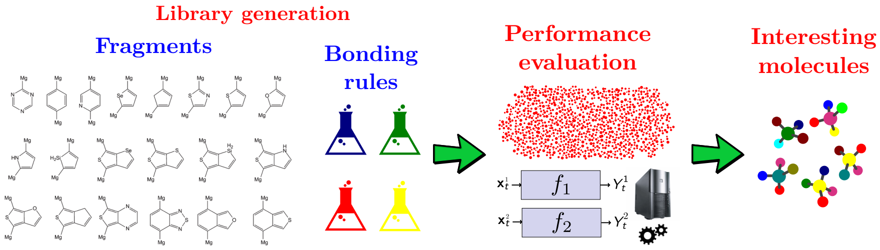
Example: new **plastic solar cells** for transforming light into electricity.





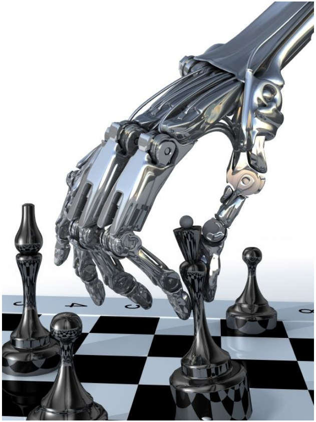
Example: new **plastic solar cells** for transforming light into electricity.





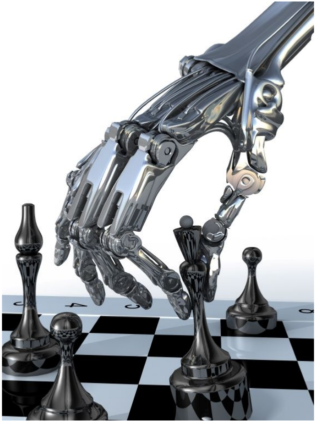
Explore **millions of candidate molecule structures** to identify the compounds with the best properties.

Example: **control system** for a robot that is able to grasp objects.



Finger Joint Trajectories

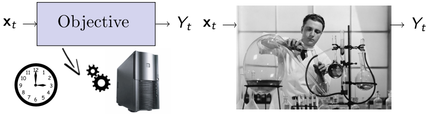
Example: **control system** for a robot that is able to grasp objects.



Parameters to tune: initial pose for the robot's hand and finger joint trajectories.

Optimization Problems: Common Features

- Very expensive evaluations.



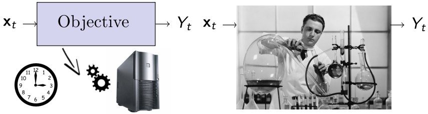
Optimization Problems: Common Features

- Very expensive evaluations.

- The objective is a black-box.

Optimization Problems: Common Features

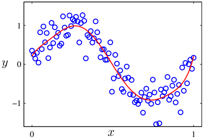
- Very expensive evaluations.



- The objective is a black-box.

- The evaluation can be noisy.

$$y = f(\mathbf{x}) + \epsilon$$



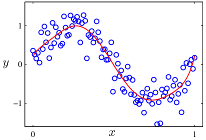
Optimization Problems: Common Features

- Very expensive evaluations.

- The objective is a black-box.

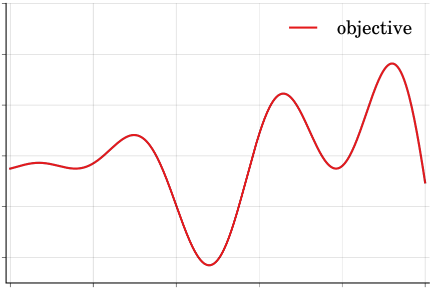
- The evaluation can be noisy.

$$y = f(\mathbf{x}) + \epsilon$$



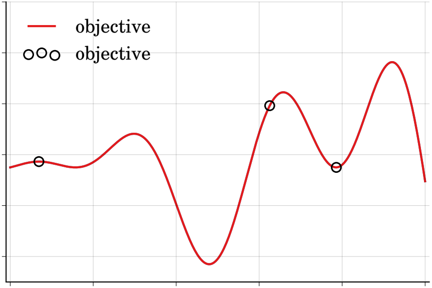
Bayesian optimization methods can be used to solve these problems!

Bayesian Optimization in Practice



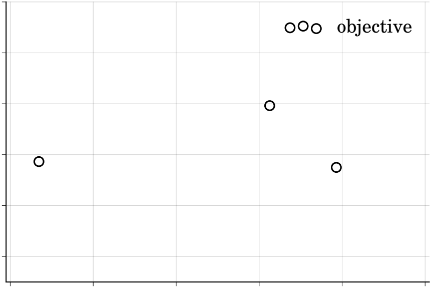
1. Get initial sample.

Bayesian Optimization in Practice



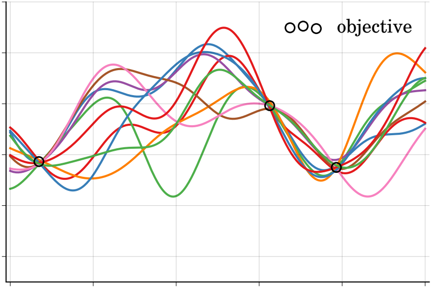
1. Get initial sample.

Bayesian Optimization in Practice



1. **Get initial sample.**
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$

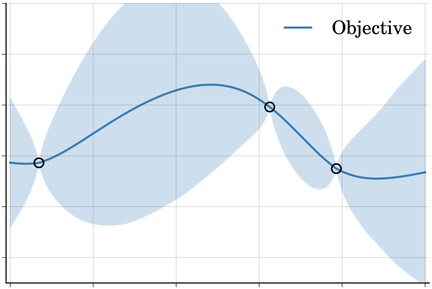
Bayesian Optimization in Practice



1. Get initial sample.
2. Fit a model to the data:

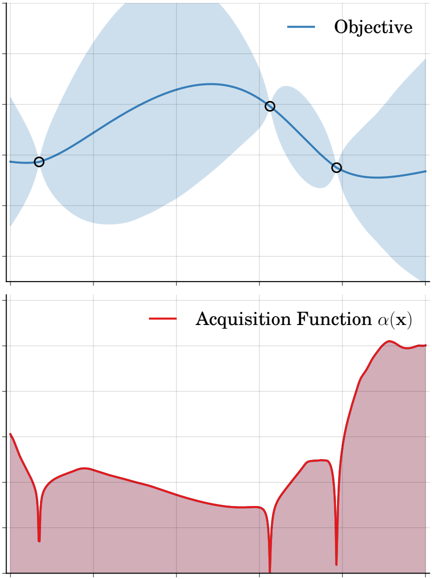
$$p(y|\mathbf{x}, \mathcal{D}_n) \, .$$

Bayesian Optimization in Practice



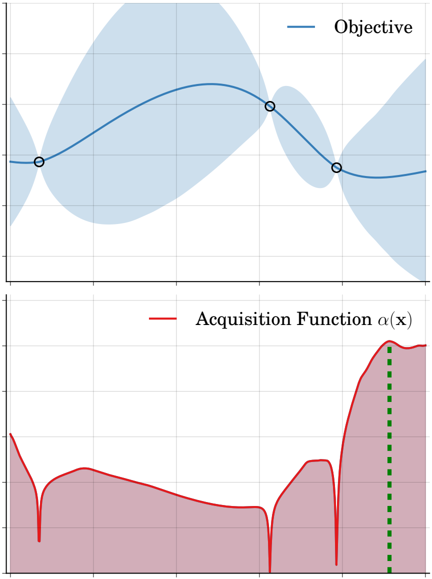
1. Get initial sample.
2. **Fit a model to the data:**
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. Select data collection strategy:
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$

Bayesian Optimization in Practice



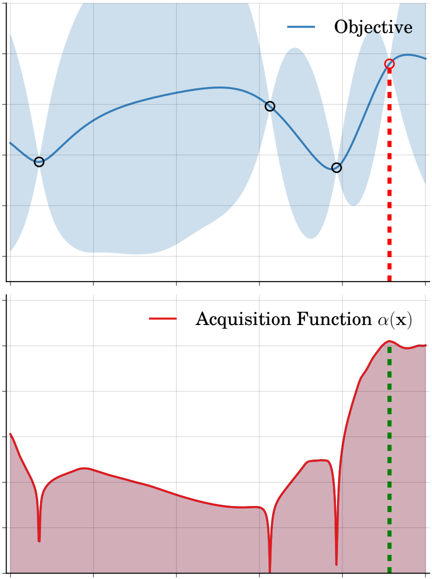
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. Select data collection strategy:
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. Optimize acquisition function $\alpha(\mathbf{x})$.

Bayesian Optimization in Practice



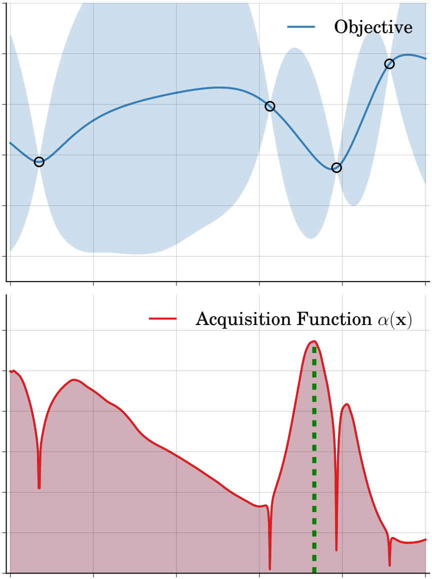
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. Select data collection strategy:
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. Optimize acquisition function $\alpha(\mathbf{x})$.
5. Collect data and update model.

Bayesian Optimization in Practice



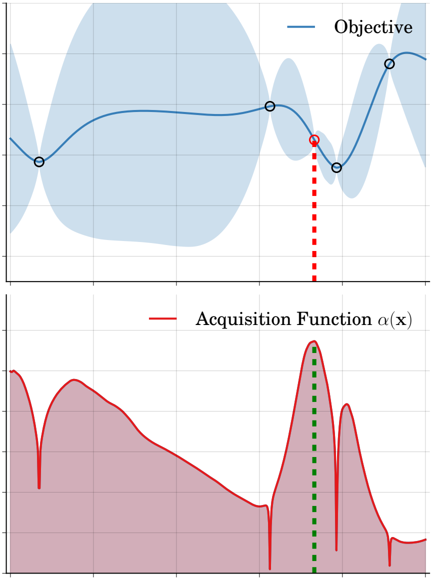
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. Select data collection strategy:
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. Optimize acquisition function $\alpha(\mathbf{x})$.
5. Collect data and update model.
6. Repeat!

Bayesian Optimization in Practice



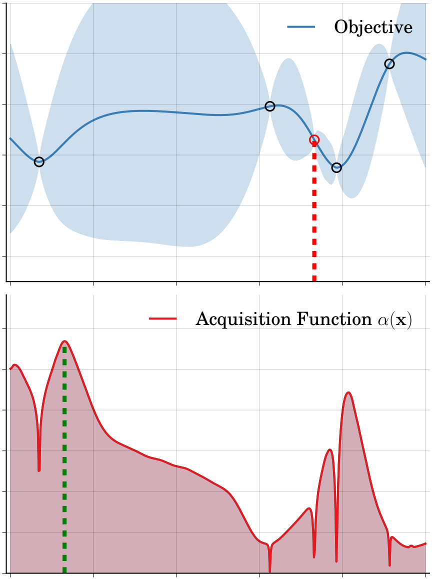
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



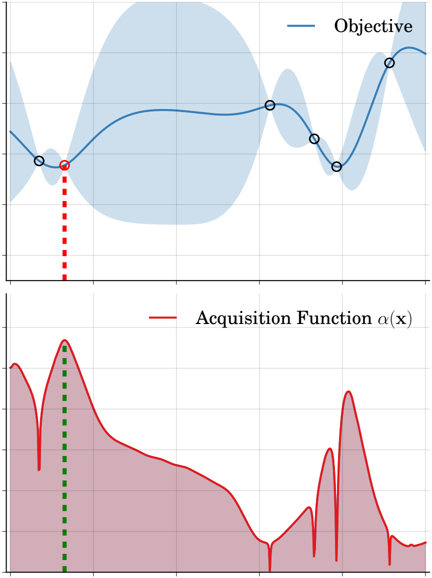
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x},\mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



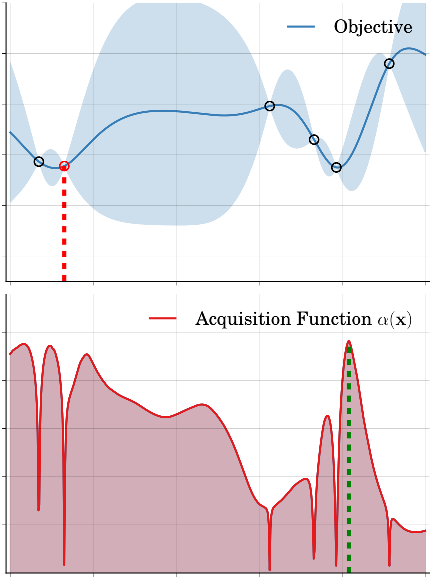
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x},\mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



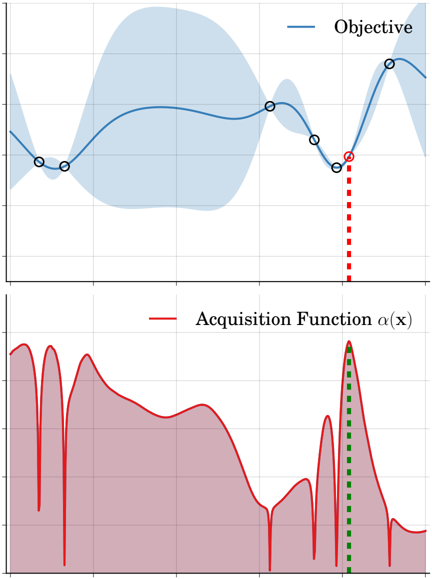
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



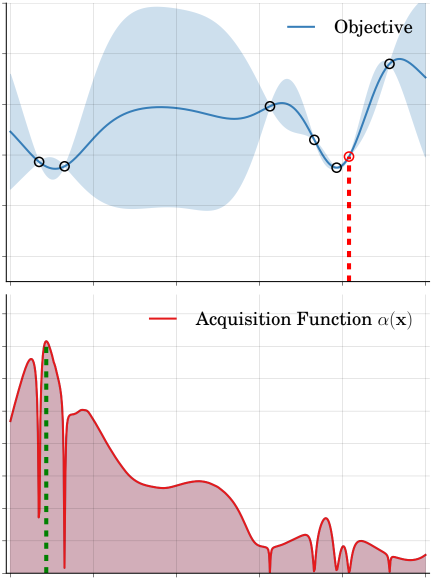
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x},\mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



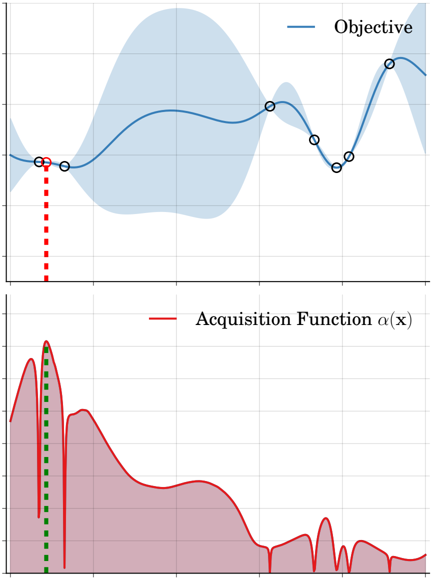
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



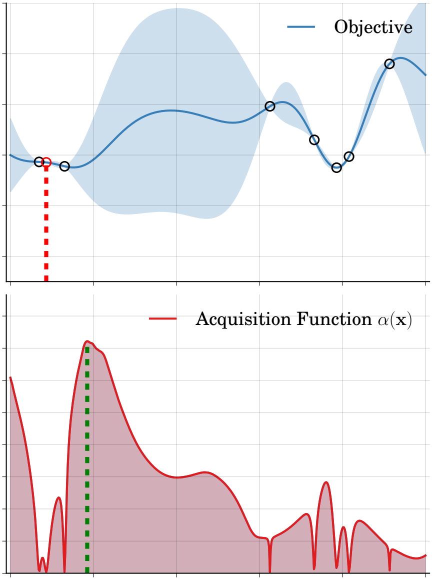
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



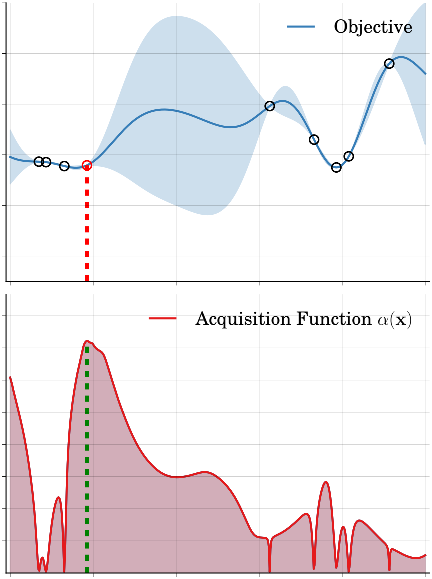
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



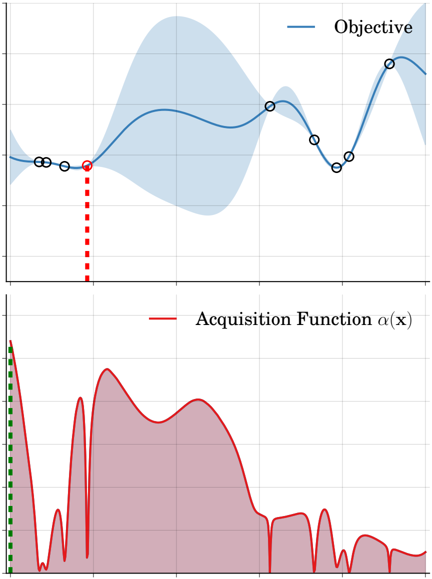
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



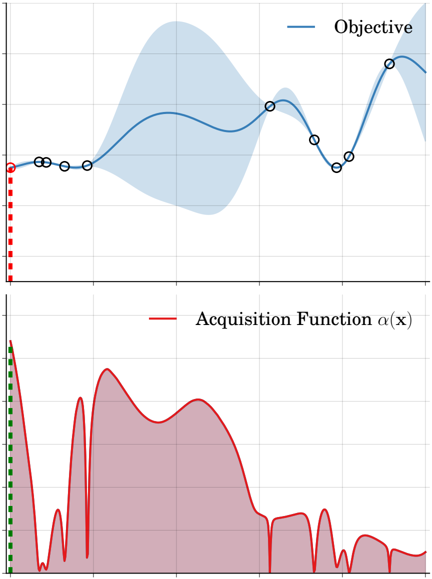
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



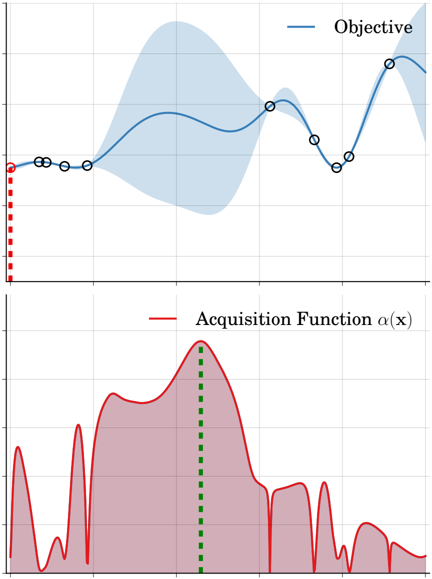
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



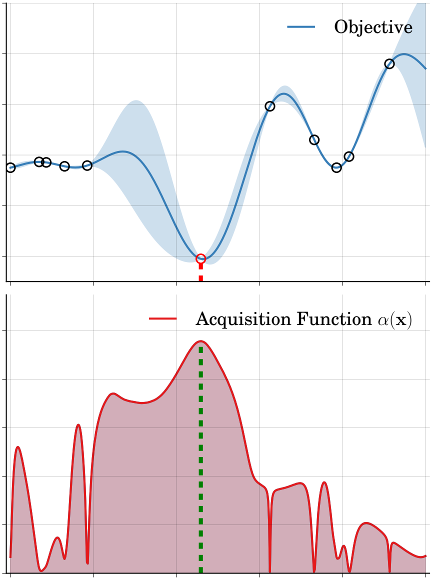
1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



1. Get initial sample.
2. Fit a model to the data:
$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$
3. **Select data collection strategy:**
$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x},\mathcal{D}_n)]\,.$$
4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



1. Get initial sample.
2. Fit a model to the data:

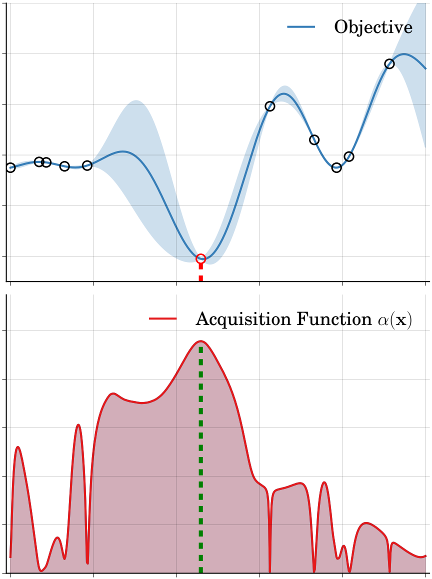
$$p(y|\mathbf{x}, \mathcal{D}_n).$$

3. **Select data collection strategy:**

$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)].$$

4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

Bayesian Optimization in Practice



1. Get initial sample.
2. Fit a model to the data:

$$p(y|\mathbf{x}, \mathcal{D}_n)\,.$$

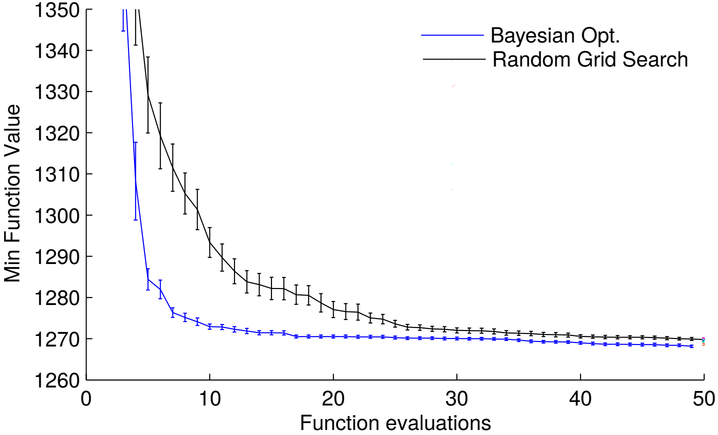
3. **Select data collection strategy:**

$$\alpha(\mathbf{x}) = \mathbf{E}_{p(y|\mathbf{x},\mathcal{D}_n)}[U(y|\mathbf{x}, \mathcal{D}_n)]\,.$$

4. **Optimize acquisition function $\alpha(\mathbf{x})$.**
5. **Collect data and update model.**
6. **Repeat!**

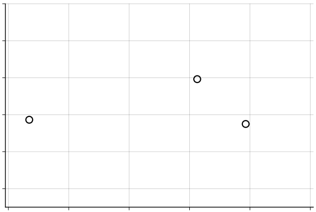
The model guides the search focusing on the most-promising regions of the input space!

Bayesian Optimization vs. Uniform Exploration

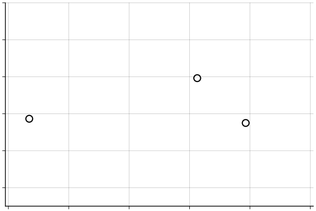


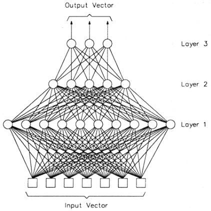
Tuning LDA on a collection of Wikipedia articles (Snoek *et al.*, 2012).

Fitting a Model to the Data



Fitting a Model to the Data

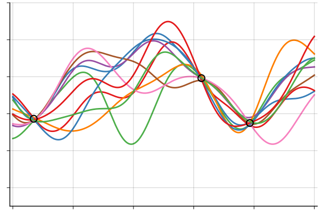


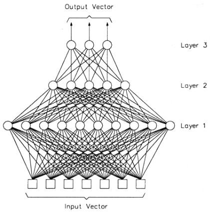


$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

Fitting a Model to the Data

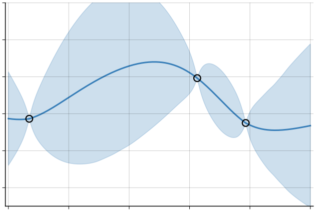


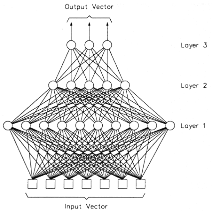


$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

Fitting a Model to the Data

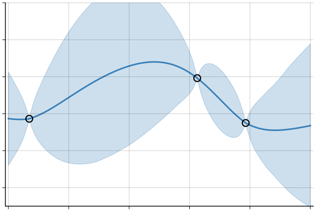


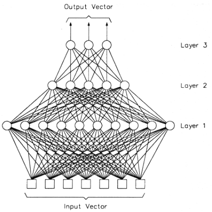


$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

Fitting a Model to the Data





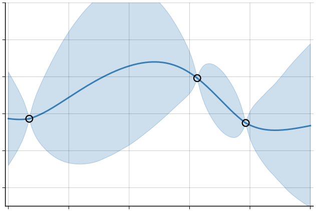
$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

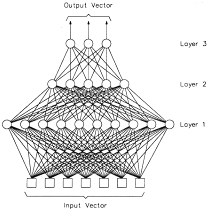
$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

Posterior Dist. $\quad p(\mathbf{W}|\mathrm{Data}) = p(\mathbf{W})p(\mathrm{Data}|\mathbf{W})/p(\mathrm{Data})$

Predictive Dist. $\quad p(y|\mathrm{Data}, x) = \int p(y|\mathbf{W}, x)p(\mathbf{W}|\mathrm{Data})d\mathbf{W}$

Fitting a Model to the Data





$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

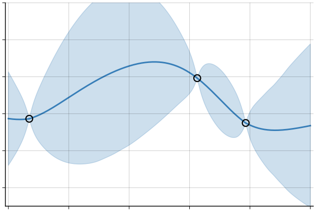
$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

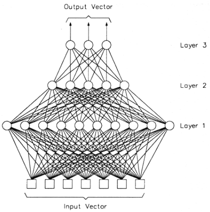
Posterior Dist. $p(\mathbf{W}|\text{Data}) = p(\mathbf{W})p(\text{Data}|\mathbf{W})/p(\text{Data})$

Predictive Dist. $p(y|\text{Data}, x) = \int p(y|\mathbf{W}, x)p(\mathbf{W}|\text{Data})d\mathbf{W}$

Challenges: The model should be non-parametric (the world is complicated) and computing $p(\text{Data})$ is intractable!

Fitting a Model to the Data





$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

Posterior Dist. $p(\mathbf{W}|\mathrm{Data}) = p(\mathbf{W})p(\mathrm{Data}|\mathbf{W})/p(\mathrm{Data})$

Predictive Dist. $p(y|\mathrm{Data}, x) = \int p(y|\mathbf{W}, x)p(\mathbf{W}|\mathrm{Data})d\mathbf{W}$

Challenges: The model should be non-parametric (the world is complicated) and computing $p(\mathrm{Data})$ is intractable!

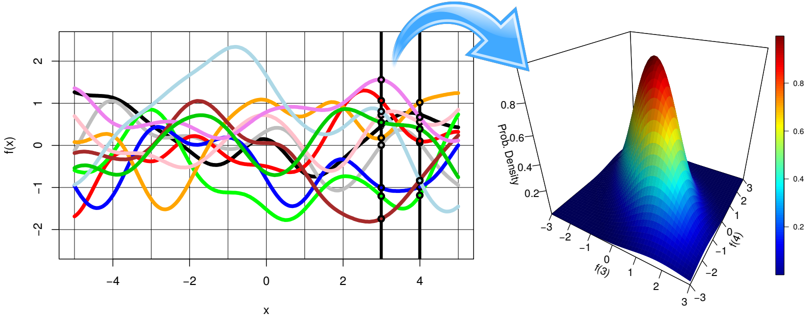
Solved by setting $p(\mathbf{W}) = \prod_{ij} \mathcal{N}(w_{ji}|0, \sigma^2 H^{-1})$ and letting $H \to \infty$!

Gaussian Processes

Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^N$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an N-dimensional Gaussian distribution.

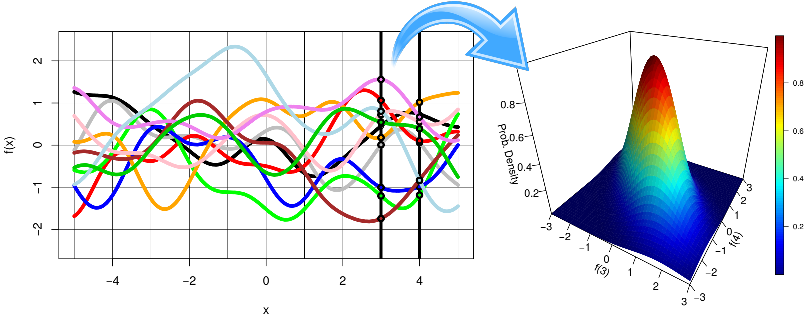
Gaussian Processes

Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^N$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\top$ follows an N-dimensional Gaussian distribution.



Gaussian Processes

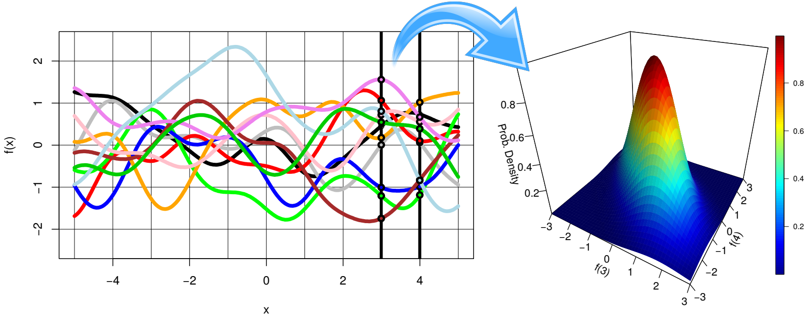
Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^{N}$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an N-dimensional Gaussian distribution.



When $H \to \infty$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an N-dimensional Gaussian where $\mathbb{E}[f(\mathbf{x}_i)f(\mathbf{x}_k)] = \sigma^2\mathbb{E}[h_j(\mathbf{x}_i)h_j(\mathbf{x}_k)]$ by the **central limit theorem**.

Gaussian Processes

Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^N$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an N-dimensional Gaussian distribution.



When $H \to \infty$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an N-dimensional Gaussian where $\mathbb{E}[f(\mathbf{x}_i)f(\mathbf{x}_k)] = \sigma^2 \mathbb{E}[h_j(\mathbf{x}_i)h_j(\mathbf{x}_k)]$ by the **central limit theorem**.

Due to Gaussian form, there are closed-form solutions for many useful questions about finite data.

Gaussian Processes

- The **joint distribution** for $\mathbf{y}^\star$ at test points $\{\mathbf{x}_m^\star\}_{m=1}^M$ and $\mathbf{y}$:

$$p(\mathbf{y}^\star, \mathbf{y}) = \mathcal{N}\left(\left[\begin{array}{c} \mathbf{0} \\ \mathbf{0} \end{array}\right], \left[\begin{array}{cc} \mathbf{k}_\theta & \mathbf{K}_\theta \\ \boldsymbol{\kappa}_\theta & \mathbf{k}_\theta^\mathsf{T} \end{array}\right]\right)$$

Gaussian Processes

- The **joint distribution** for $\mathbf{y}^\star$ at test points $\{\mathbf{x}^\star_m\}_{m=1}^M$ and $\mathbf{y}$:

$$p(\mathbf{y}^\star, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{k}_\theta & \mathbf{K}_\theta \\ \boldsymbol{\kappa}_\theta & \mathbf{k}_\theta^\mathsf{T} \end{bmatrix}\right)$$

- These **matrices** are computed from the covariance $C(\cdot, \cdot; \theta)$:

$$[\mathbf{K}_\theta]_{n,n'} = C(\mathbf{x}_n, \mathbf{x}_{n'}; \theta)$$
$$[\mathbf{k}_\theta]_{n,m} = C(\mathbf{x}_n, \mathbf{x}^\star_m; \theta)\,, \qquad [\boldsymbol{\kappa}_\theta]_{m,m'} = C(\mathbf{x}^\star_m, \mathbf{x}^\star_{m'}; \theta)\,,$$

Gaussian Processes

- The **joint distribution** for $\mathbf{y}^\star$ at test points $\{\mathbf{x}^\star_m\}_{m=1}^M$ and $\mathbf{y}$:

$$p(\mathbf{y}^\star, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{k}_\theta & \mathbf{K}_\theta \\ \boldsymbol{\kappa}_\theta & \mathbf{k}_\theta^\mathsf{T} \end{bmatrix} \right)$$

- These **matrices** are computed from the covariance $C(\cdot, \cdot; \theta)$:

$$[\mathbf{K}_\theta]_{n,n'} = C(\mathbf{x}_n, \mathbf{x}_{n'}; \theta)$$
$$[\mathbf{k}_\theta]_{n,m} = C(\mathbf{x}_n, \mathbf{x}^\star_m; \theta)\,, \qquad [\boldsymbol{\kappa}_\theta]_{m,m'} = C(\mathbf{x}^\star_m, \mathbf{x}^\star_{m'}; \theta)\,,$$

- The **predictive distribution** for $\mathbf{y}^\star$ given $\mathbf{y}$, $p(\mathbf{y}^\star | \mathbf{y})$, is:

$$\mathbf{y}^\star \sim \mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma})$$
$$\mathbf{m} = \mathbf{k}_\theta^\mathsf{T} \mathbf{K}_\theta^{-1} \mathbf{y}\,, \qquad \boldsymbol{\Sigma} = \boldsymbol{\kappa}_\theta - \mathbf{k}_\theta^\mathsf{T} \mathbf{K}_\theta^{-1} \mathbf{k}_\theta\,,$$

Gaussian Processes

- The **joint distribution** for $\mathbf{y}^\star$ at test points $\{\mathbf{x}^\star_m\}_{m=1}^M$ and $\mathbf{y}$:

$$p(\mathbf{y}^\star, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{k}_\theta & \mathbf{K}_\theta \\ \boldsymbol{\kappa}_\theta & \mathbf{k}_\theta^\top \end{bmatrix} \right)$$

- These **matrices** are computed from the covariance $C(\cdot, \cdot; \theta)$:

$$[\mathbf{K}_\theta]_{n,n'} = C(\mathbf{x}_n, \mathbf{x}_{n'}; \theta)$$
$$[\mathbf{k}_\theta]_{n,m} = C(\mathbf{x}_n, \mathbf{x}^\star_m; \theta)\,, \qquad [\boldsymbol{\kappa}_\theta]_{m,m'} = C(\mathbf{x}^\star_m, \mathbf{x}^\star_{m'}; \theta)\,,$$

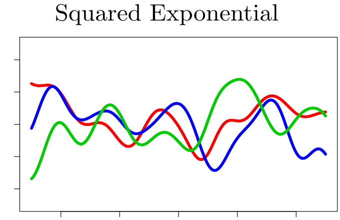
- The **predictive distribution** for $\mathbf{y}^\star$ given $\mathbf{y}$, $p(\mathbf{y}^\star|\mathbf{y})$, is:

$$\mathbf{y}^\star \sim \mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma})$$
$$\mathbf{m} = \mathbf{k}_\theta^\top \mathbf{K}_\theta^{-1} \mathbf{y}\,, \qquad \boldsymbol{\Sigma} = \boldsymbol{\kappa}_\theta - \mathbf{k}_\theta^\top \mathbf{K}_\theta^{-1} \mathbf{k}_\theta\,,$$

- The log of the **marginal likelihood**, $p(\mathbf{y}|\theta)$, is:

$$\log p(\mathbf{y}) = -\frac{N}{2} \log 2\pi - \frac{1}{2} \log |\mathbf{K}_\theta| - \frac{1}{2} \mathbf{y}^\top \mathbf{K}_\theta^{-1} \mathbf{y}$$

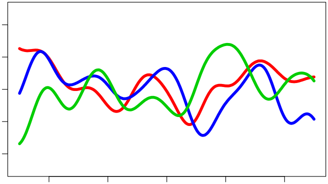
Some Covariance Functions



$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ \frac{1}{2} \sum_j \left(\frac{x_j - x_j'}{l_j} \right)^2 \right\}$$

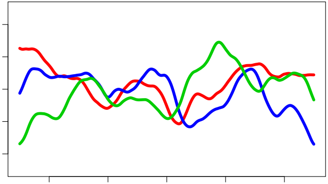
Some Covariance Functions

Squared Exponential



$$C(\mathbf{x},\mathbf{x}') = \sigma^2 \exp\left\{\frac{1}{2}\sum_j\left(\frac{x_j - x'_j}{l_j}\right)^2\right\}$$

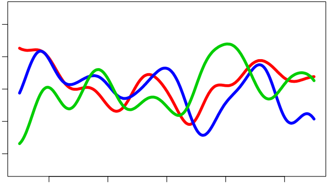
Matérn



$$C(\mathbf{x},\mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{l}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}r}{l}\right)$$

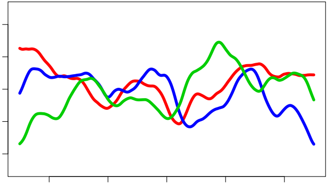
Some Covariance Functions

Squared Exponential



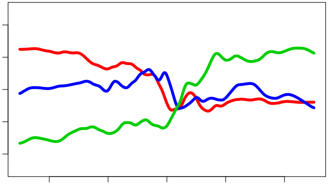
$$C(\mathbf{x},\mathbf{x}') = \sigma^2 \exp\left\{\frac{1}{2}\sum_j \left(\frac{x_j - x_j'}{l_j}\right)^2\right\}$$

Matérn



$$C(\mathbf{x},\mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}r}{l}\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu}r}{l}\right)$$

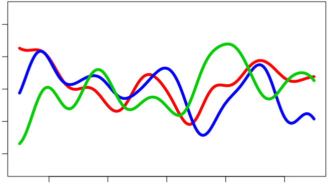
Neural Network



$$C(\mathbf{x},\mathbf{x}') = \frac{2}{\pi}\sin^{-1}\left(\frac{2\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}'}{\sqrt{(1+2\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x})(1+2\mathbf{x}'^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}')}}\right)$$

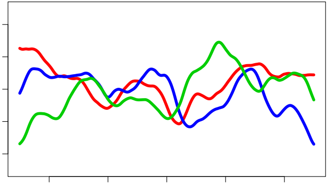
Some Covariance Functions

Squared Exponential



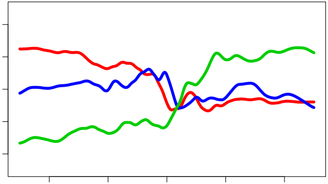
$$C(\mathbf{x},\mathbf{x}') = \sigma^2 \exp\left\{ \frac{1}{2} \sum_j \left(\frac{x_j - x'_j}{l_j} \right)^2 \right\}$$

Matérn



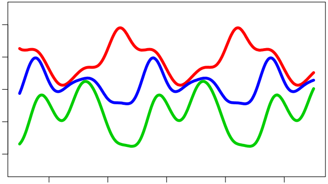
$$C(\mathbf{x},\mathbf{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} r}{l} \right)^{\nu} K_{\nu} \left(\frac{\sqrt{2\nu} r}{l} \right)$$

Neural Network



$$C(\mathbf{x},\mathbf{x}') = \frac{2}{\pi} \sin^{-1} \left(\frac{2\mathbf{x}^{\mathrm{T}} \boldsymbol{\Sigma} \mathbf{x}'}{\sqrt{(1+2\mathbf{x}^{\mathrm{T}} \boldsymbol{\Sigma} \mathbf{x})(1+2\mathbf{x}'^{\mathrm{T}} \boldsymbol{\Sigma} \mathbf{x}')}} \right)$$

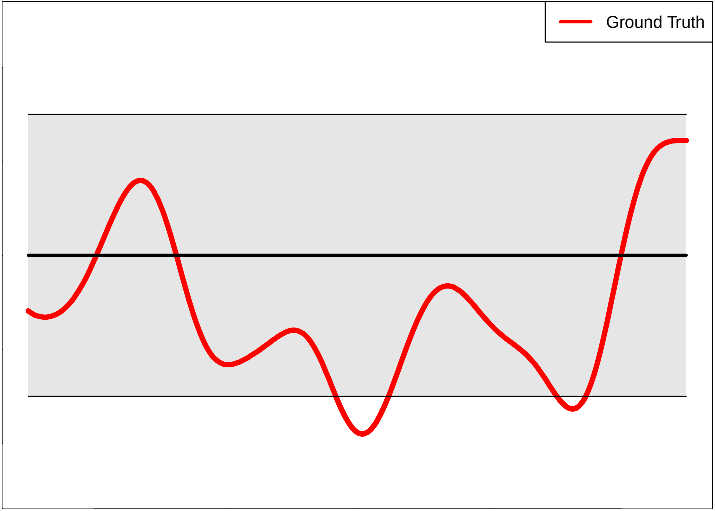
Periodic



$$C(\mathbf{x},\mathbf{x}') = \exp\left\{ - \frac{2 \sin^2 \left(\frac{|\mathbf{x}-\mathbf{x}'|}{2} \right)}{l^2} \right\}$$

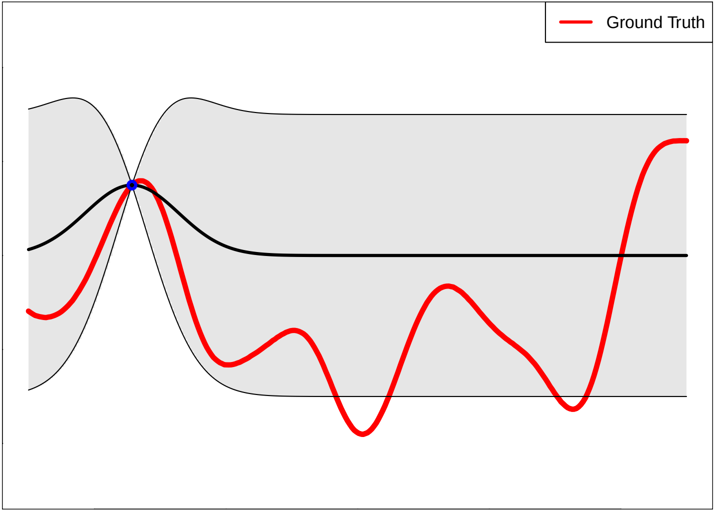
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



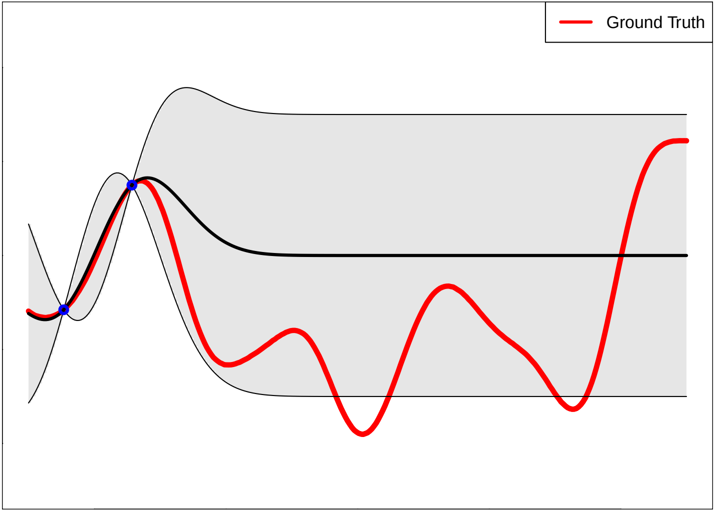
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



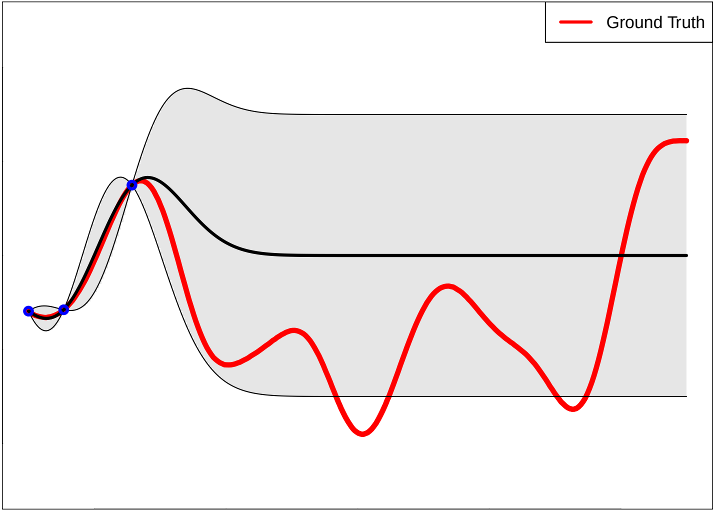
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



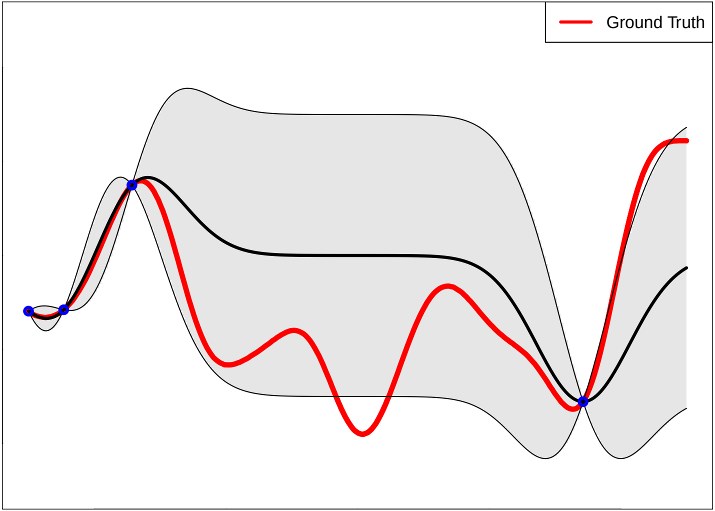
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



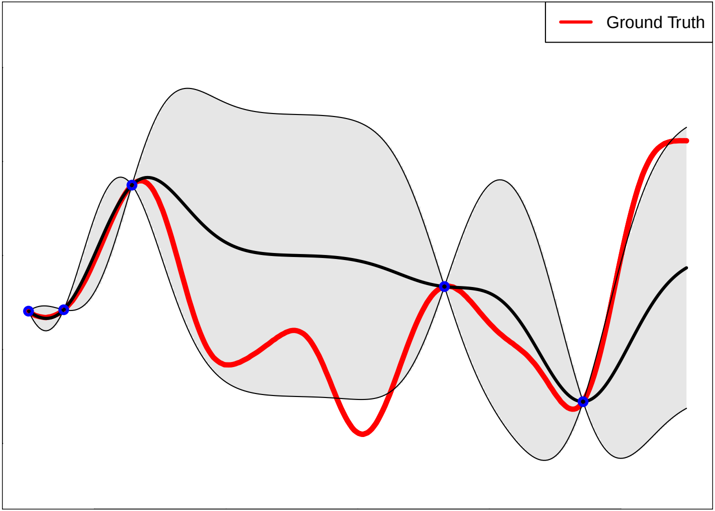
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



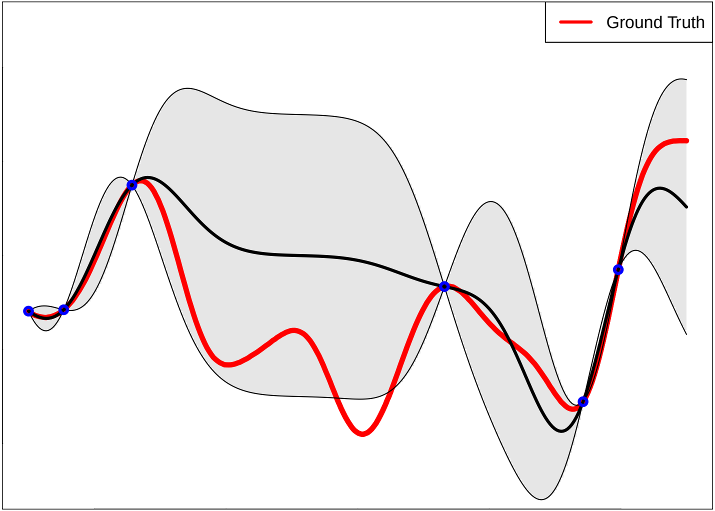
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



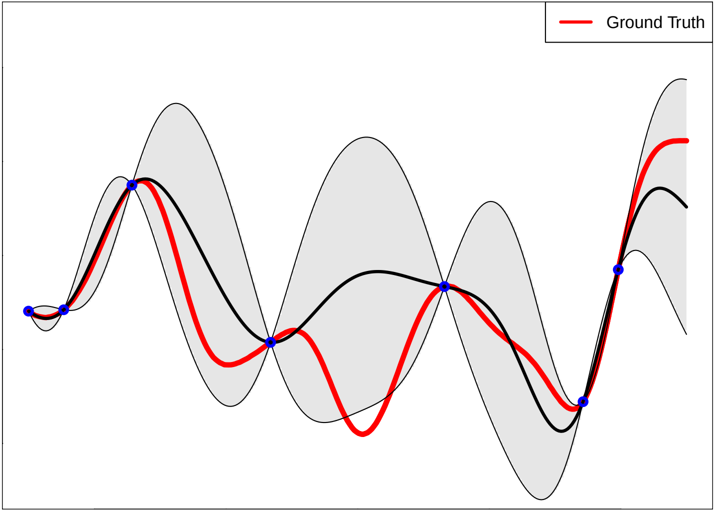
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



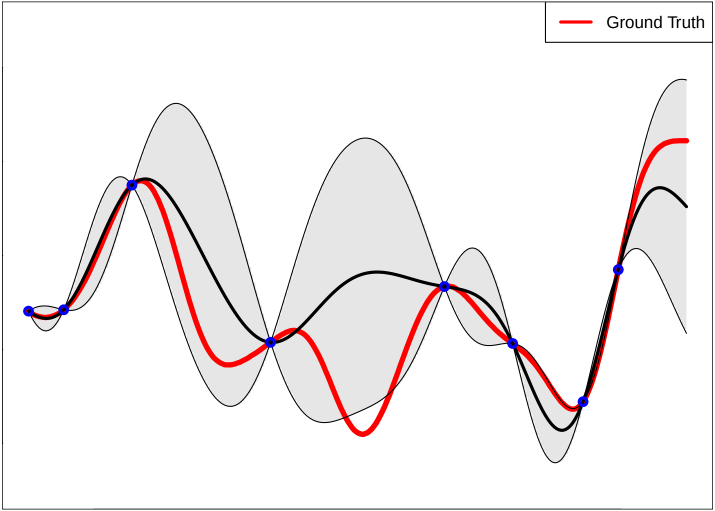
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



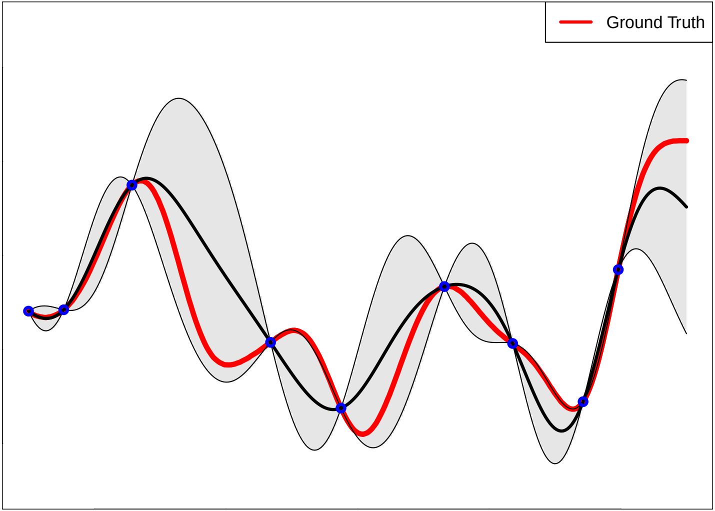
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



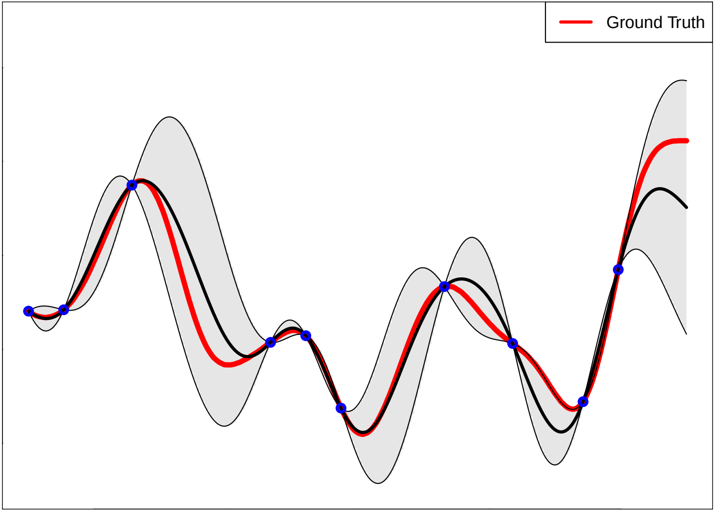
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



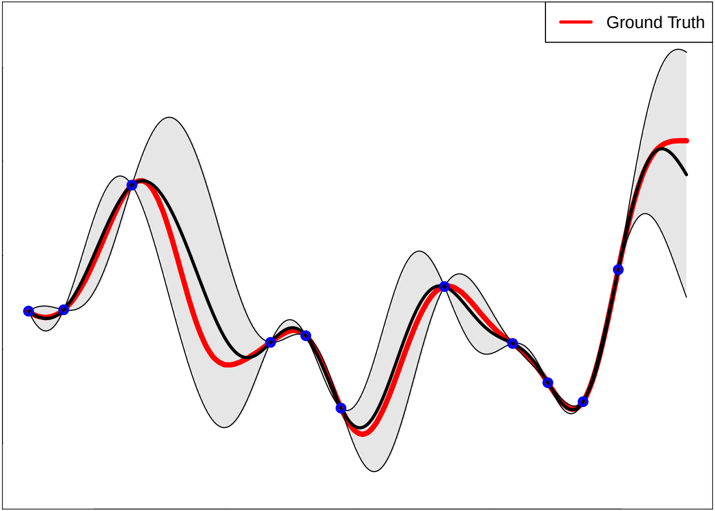
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



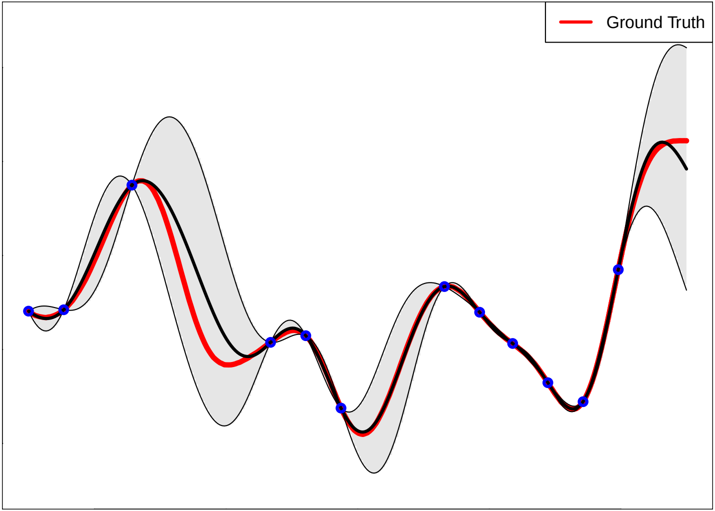
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



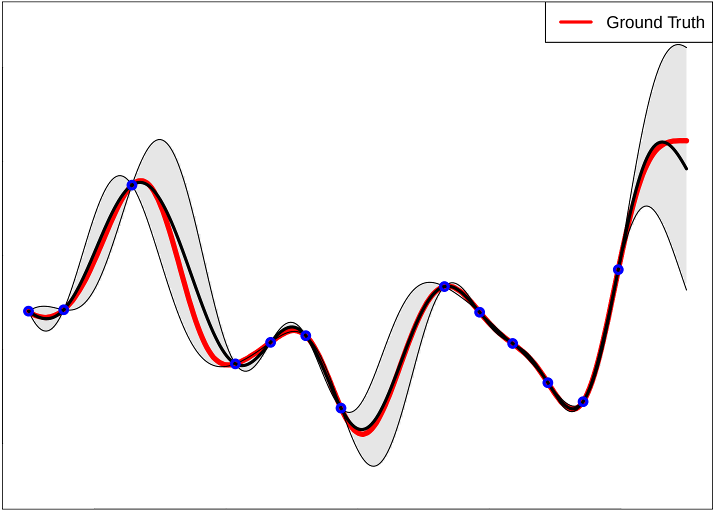
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



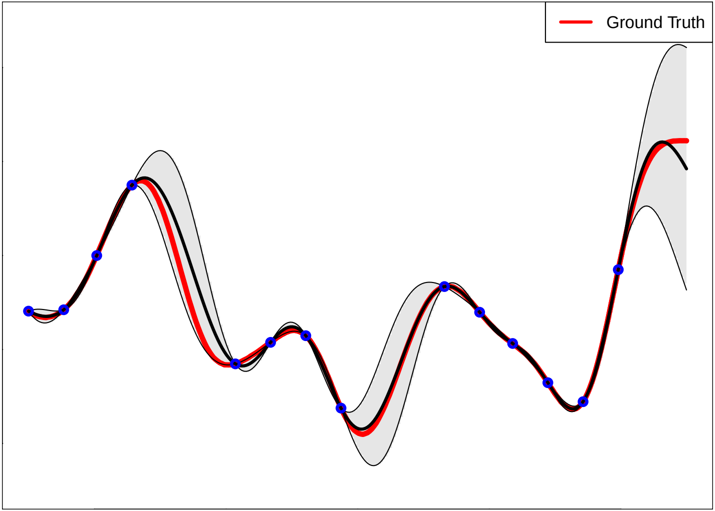
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



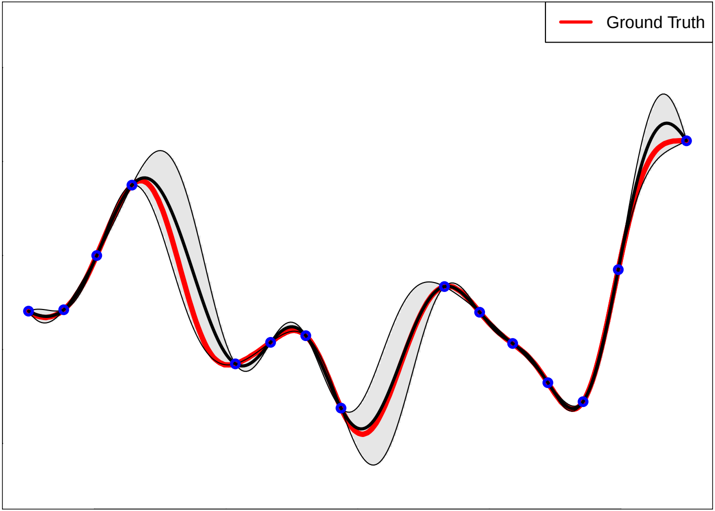
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



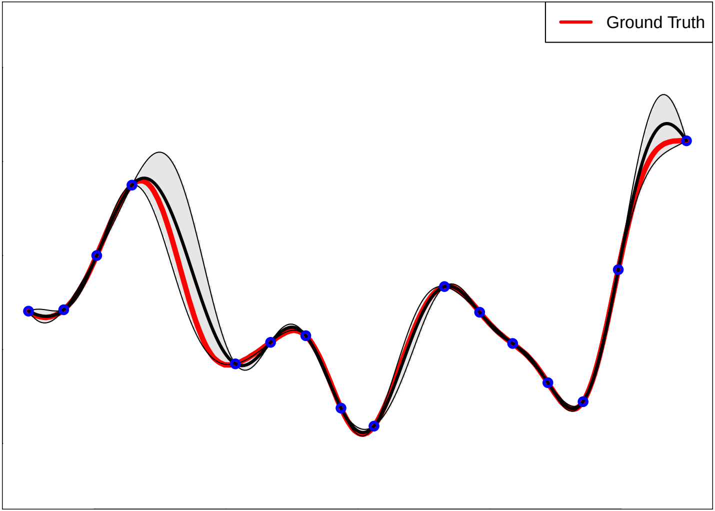
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



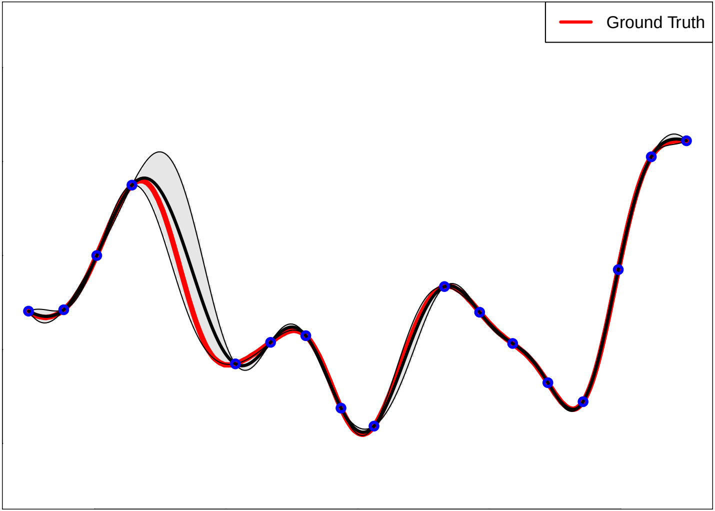
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



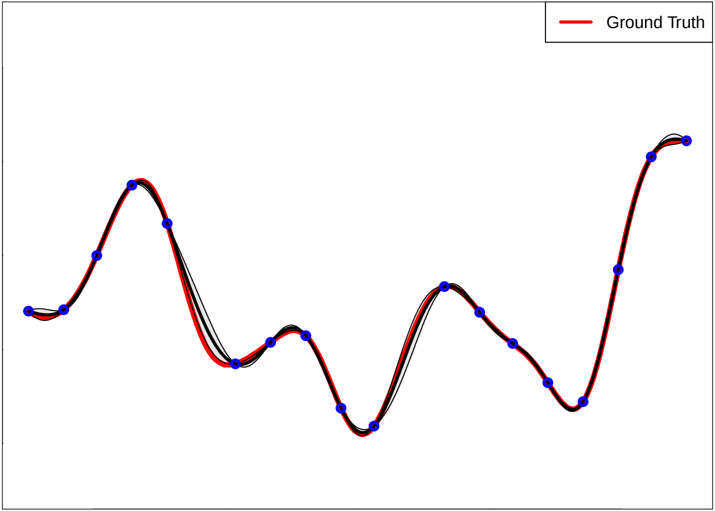
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



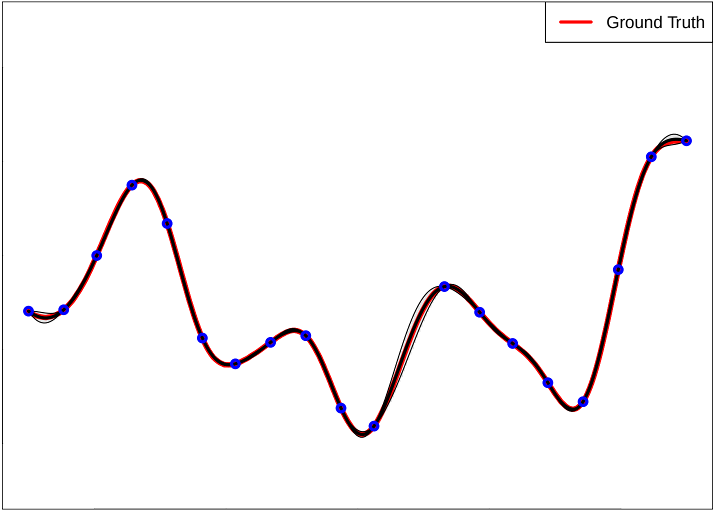
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



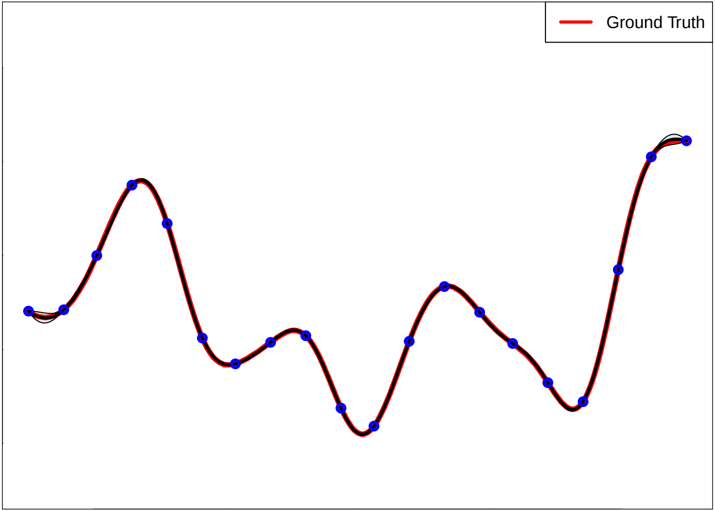
From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



From the Prior to the Posterior

GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.

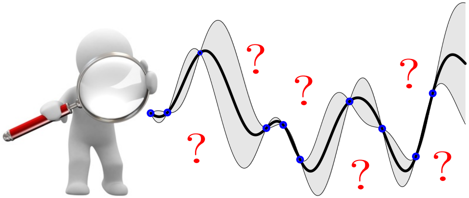


Using the GP Uncertainty in Optimization

Where to evaluate **next**?

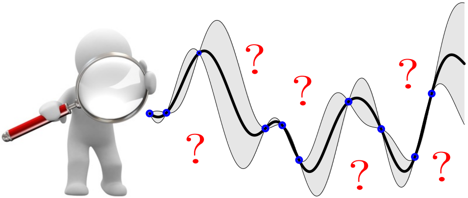
Using the GP Uncertainty in Optimization

Where to evaluate **next**?



Using the GP Uncertainty in Optimization

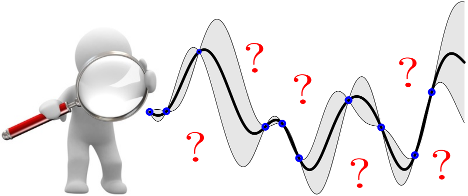
Where to evaluate **next**?



- **Exploration:** seek places with high variance.

Using the GP Uncertainty in Optimization

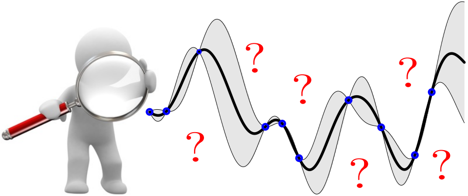
Where to evaluate **next**?



- **Exploration:** seek places with high variance.
- **Exploitation:** seek places with low mean.

Using the GP Uncertainty in Optimization

Where to evaluate **next**?

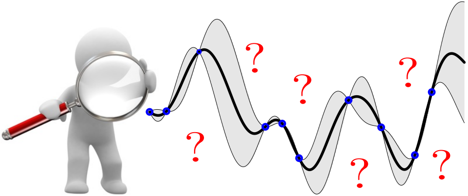


- **Exploration:** seek places with high variance.
- **Exploitation:** seek places with low mean.

The acquisition function balances these two, to choose in an intelligent way the next evaluation point!

Using the GP Uncertainty in Optimization

Where to evaluate **next**?



- **Exploration:** seek places with high variance.
- **Exploitation:** seek places with low mean.

The acquisition function balances these two, to choose in an intelligent way the next evaluation point!

$$\alpha(\mathbf{x}) = \mathbb{E}_{p(y^\star|\mathcal{D}_N,\mathbf{x})}\left[U(y^\star|\mathbf{x},\mathcal{D}_N)\right]$$

Some Acquisition Functions

Let $\nu = \min\{y_1, \ldots, y_N\}$ and $\gamma(\mathbf{x}) = \frac{\nu - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$.

Some Acquisition Functions

Let $\nu = \min\{y_1, \ldots, y_N\}$ and $\gamma(\mathbf{x}) = \frac{\nu - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$.

- **Probability of Improvement**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \mathbb{I}(y_\star < \nu)\,, \qquad \alpha(\mathbf{x}) = \Phi\left(\gamma(\mathbf{x})\right)$$

Some Acquisition Functions

Let $\nu = \min\{y_1, \ldots, y_N\}$ and $\gamma(\mathbf{x}) = \frac{\nu - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$.

- **Probability of Improvement**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \mathbb{I}(y_\star < \nu)\,, \qquad \alpha(\mathbf{x}) = \Phi\left(\gamma(\mathbf{x})\right)$$

- **Expected Improvement**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \max(0, \nu - y^\star)\,, \alpha(\mathbf{x}) = \sigma(\mathbf{x})\left(\gamma(\mathbf{x})\Phi\left(\gamma(\mathbf{x})\right) + \phi(\gamma(\mathbf{x}))\right)$$

Some Acquisition Functions

Let $\nu = \min\{y_1, \ldots, y_N\}$ and $\gamma(\mathbf{x}) = \frac{\nu - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$.

- **Probability of Improvement**:

$$U(y^\star|\mathcal{D}_N, \mathbf{x}) = \mathbb{I}(y_\star < \nu)\,, \qquad \alpha(\mathbf{x}) = \Phi\left(\gamma(\mathbf{x})\right)$$

- **Expected Improvement**:

$$U(y^\star|\mathcal{D}_N, \mathbf{x}) = \max(0, \nu - y^\star)\,, \alpha(\mathbf{x}) = \sigma(\mathbf{x})\left(\gamma(\mathbf{x})\Phi\left(\gamma(\mathbf{x})\right) + \phi(\gamma(\mathbf{x}))\right)$$

- **Lower Confidence Bound**:

$$\alpha(\mathbf{x}) = -\left(\mu(\mathbf{x}) - \kappa\sigma(\mathbf{x})\right)$$

Some Acquisition Functions

Let $\nu = \min\{y_1, \ldots, y_N\}$ and $\gamma(\mathbf{x}) = \frac{\nu - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$.

- **Probability of Improvement**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \mathbb{I}(y_\star < \nu)\,, \qquad \alpha(\mathbf{x}) = \Phi\left(\gamma(\mathbf{x})\right)$$

- **Expected Improvement**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \max(0, \nu - y^\star)\,, \alpha(\mathbf{x}) = \sigma(\mathbf{x})\left(\gamma(\mathbf{x})\Phi\left(\gamma(\mathbf{x})\right) + \phi(\gamma(\mathbf{x}))\right)$$

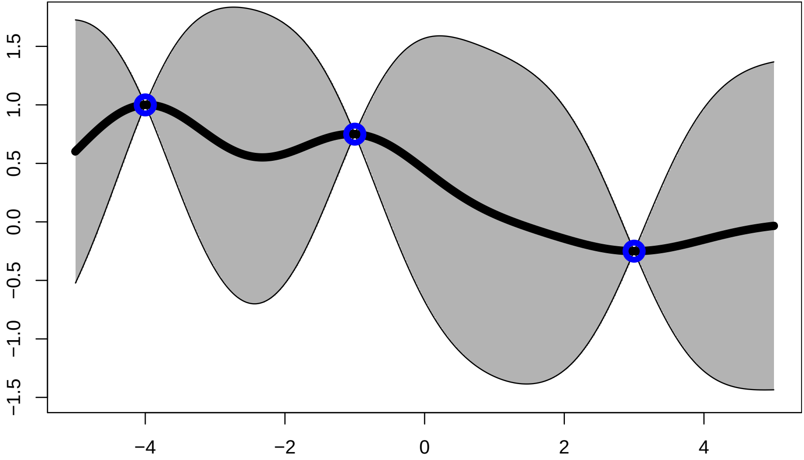
- **Lower Confidence Bound**:

$$\alpha(\mathbf{x}) = -\left(\mu(\mathbf{x}) - \kappa\sigma(\mathbf{x})\right)$$

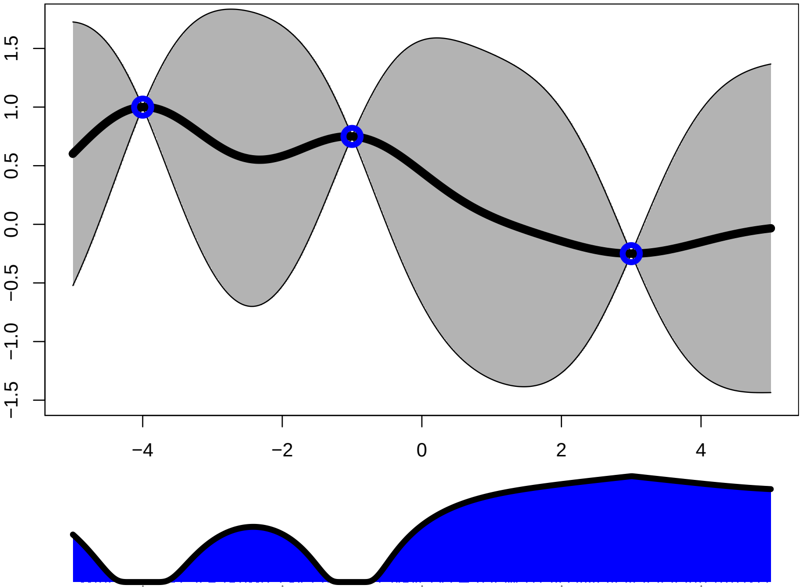
- **Entropy Search**:

$$U(y^\star | \mathcal{D}_N, \mathbf{x}) = \mathrm{H}[p(\mathbf{x}_{\min} | \mathcal{D}_N)] - \mathrm{H}[p(\mathbf{x}_{\min} | \mathcal{D}_N \cup \{\mathbf{x}, y^\star\})]$$

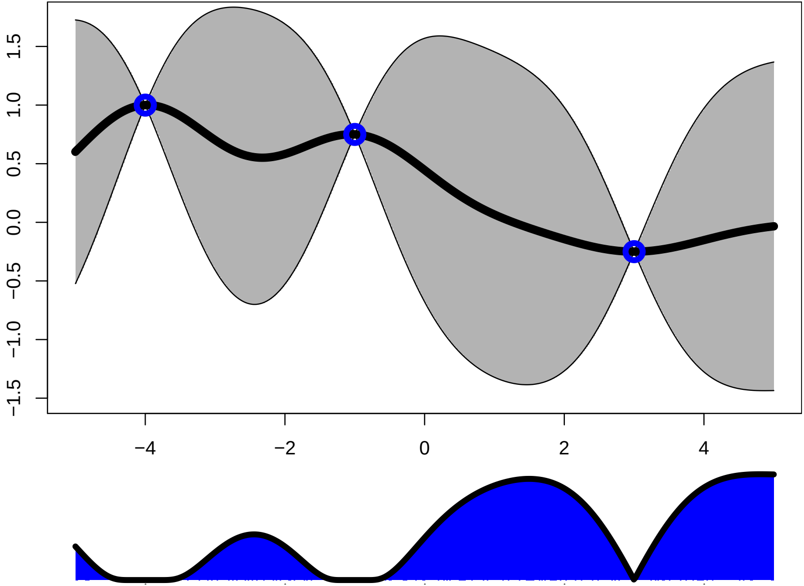
Some Acquisition Functions:



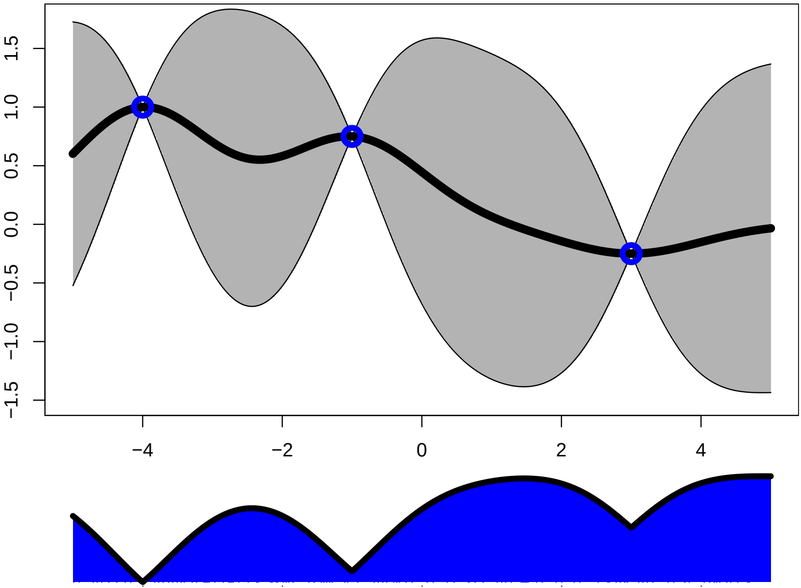
Some Acquisition Functions: Prob. Improvement



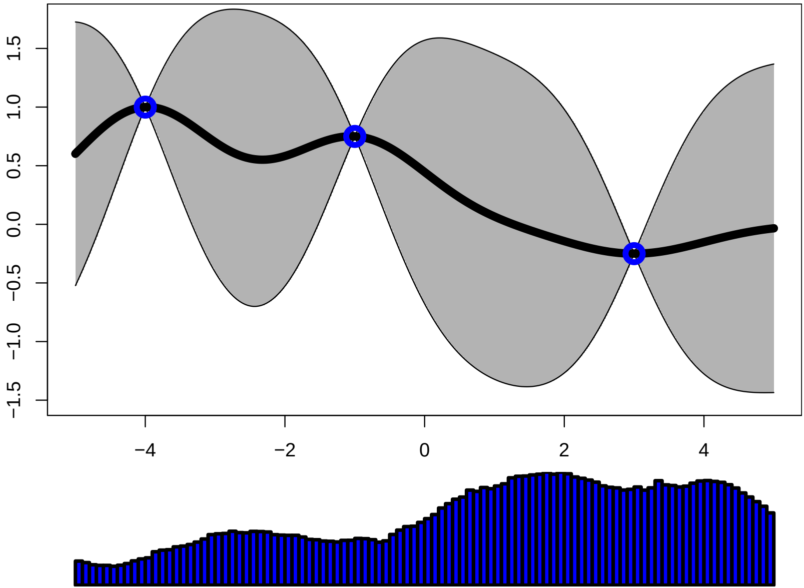
Some Acquisition Functions: Exp. Improvement



Some Acquisition Functions: Lower Conf. Bound



Some Acquisition Functions: Entropy Search

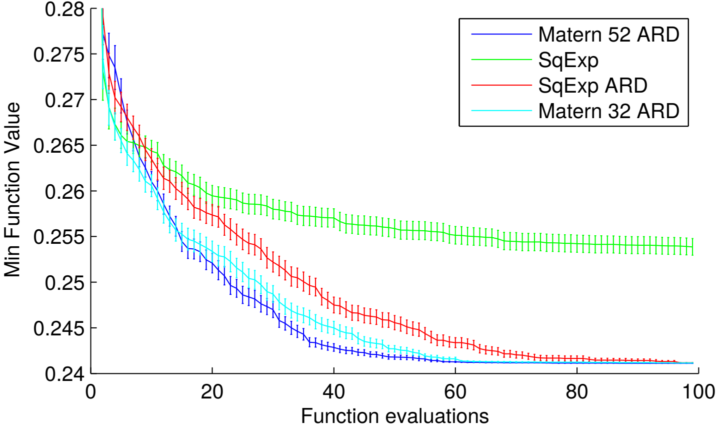


Bayesian Optimization and Model Selection

- **Covariance function selection:** **critical** to achieve good performance. The default choice for regression (squared exponential) is too smooth. Matérn $\nu = 5/2$ kernel works better.

Bayesian Optimization and Model Selection

- **Covariance function selection:** **critical** to achieve good performance. The default choice for regression (squared exponential) is too smooth. Matérn $\nu = 5/2$ kernel works better.



Structured SVM for protein motif finding (Snoek *et al.*, 2012).

Bayesian Optimization and Model Selection

- **Hyper-parameter selection:** with a small number of observations maximizing $p(\mathbf{y}|\theta)$ can give **too confident** uncertainty estimates.

Bayesian Optimization and Model Selection

- **Hyper-parameter selection:** with a small number of observations maximizing $p(\mathbf{y}|\theta)$ can give **too confident** uncertainty estimates.
- **Sampling the hyper-parameters:** computing $p(\theta|\mathbf{y})$ is **intractable!** Alternative: generate a few samples form $p(\theta|\mathbf{y})$ using MCMC.

Bayesian Optimization and Model Selection

- **Hyper-parameter selection:** with a small number of observations maximizing $p(\mathbf{y}|\theta)$ can give **too confident** uncertainty estimates.
- **Sampling the hyper-parameters:** computing $p(\theta|\mathbf{y})$ is **intractable!** Alternative: generate a few samples form $p(\theta|\mathbf{y})$ using MCMC.

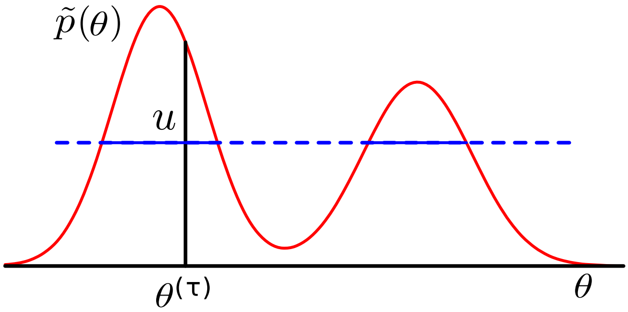
Slice sampling means no additional hyper-parameters!

(Neal, 2003)

Bayesian Optimization and Model Selection

- **Hyper-parameter selection:** with a small number of observations maximizing $p(\mathbf{y}|\theta)$ can give **too confident** uncertainty estimates.
- **Sampling the hyper-parameters:** computing $p(\theta|\mathbf{y})$ is **intractable!** Alternative: generate a few samples form $p(\theta|\mathbf{y})$ using MCMC.

Slice sampling means no additional hyper-parameters!

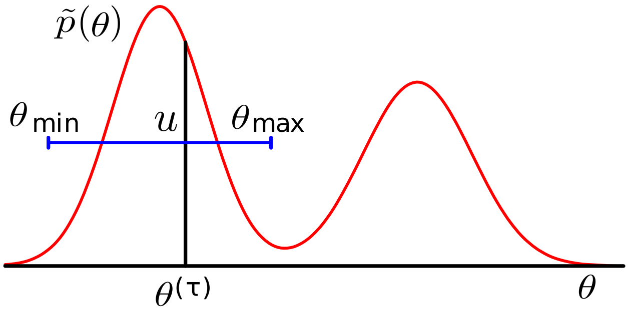


(Neal, 2003)

Bayesian Optimization and Model Selection

- **Hyper-parameter selection:** with a small number of observations maximizing $p(\mathbf{y}|\theta)$ can give **too confident** uncertainty estimates.
- **Sampling the hyper-parameters:** computing $p(\theta|\mathbf{y})$ is **intractable!** Alternative: generate a few samples form $p(\theta|\mathbf{y})$ using MCMC.

Slice sampling means no additional hyper-parameters!



(Neal, 2003)

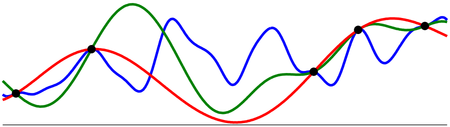
Integrated Acquisition Function

$$\hat{\alpha}(\mathbf{x}) = \int \alpha(\mathbf{x};\theta)p(\theta|\mathbf{y})d\theta \approx \frac{1}{K}\sum_{k=1}^{K}\alpha(\mathbf{x};\theta^{(k)}) \quad \theta^{(k)} \sim p(\theta|\mathbf{y}),$$

Integrated Acquisition Function

$$\hat{\alpha}(\mathbf{x}) = \int \alpha(\mathbf{x};\theta) p(\theta|\mathbf{y}) d\theta \approx \frac{1}{K} \sum_{k=1}^{K} \alpha(\mathbf{x};\theta^{(k)}) \quad \theta^{(k)} \sim p(\theta|\mathbf{y}),$$

Posterior samples with three different length-scales

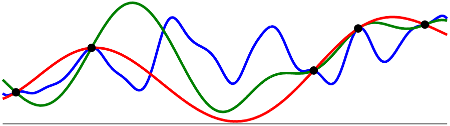


(Snoek *et al.*, 2012)

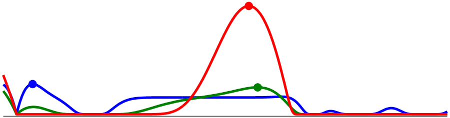
Integrated Acquisition Function

$$\hat{\alpha}(\mathbf{x}) = \int \alpha(\mathbf{x};\theta)p(\theta|\mathbf{y})d\theta \approx \frac{1}{K}\sum_{k=1}^{K}\alpha(\mathbf{x};\theta^{(k)}) \quad \theta^{(k)} \sim p(\theta|\mathbf{y})\,,$$

Posterior samples with three different length-scales



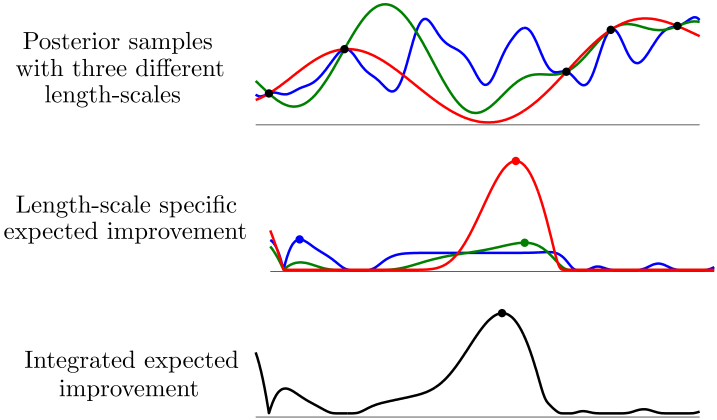
Length-scale specific expected improvement



(Snoek *et al.*, 2012)

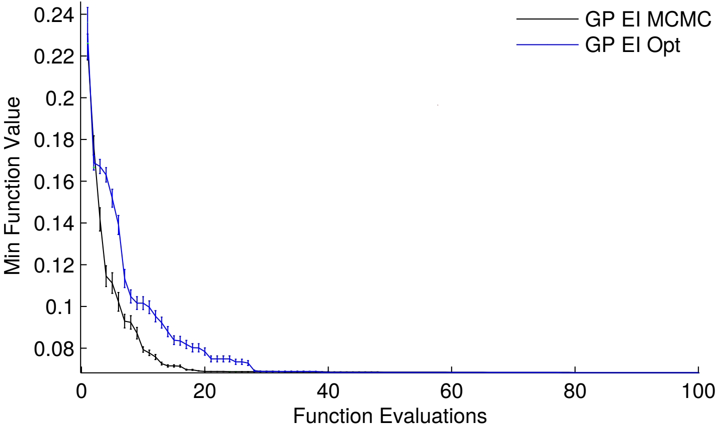
Integrated Acquisition Function

$$\hat{\alpha}(\mathbf{x}) = \int \alpha(\mathbf{x};\theta)p(\theta|\mathbf{y})d\theta \approx \frac{1}{K}\sum_{k=1}^{K}\alpha(\mathbf{x};\theta^{(k)}) \quad \theta^{(k)} \sim p(\theta|\mathbf{y}),$$



(Snoek *et al.*, 2012)

MCMC estimation vs. Maximization



Logistic regression on the MNIST (Snoek *et al.*, 2012).

Cost-sensitive Bayesian Optimization

- Different inputs may have **different computational costs**, *e.g.*, training a neural network of increasing hidden layers and units.

Cost-sensitive Bayesian Optimization

- Different inputs may have **different computational costs**, *e.g.*, training a neural network of increasing hidden layers and units.
- Better to do **cheap evaluations** before expensive ones!

Cost-sensitive Bayesian Optimization

- Different inputs may have **different computational costs**, *e.g.*, training a neural network of increasing hidden layers and units.
- Better to do **cheap evaluations** before expensive ones!
- The evaluation costs are **unknown** but they can be **recorded** and then **modeled** with an additional **Gaussian process**.

Cost-sensitive Bayesian Optimization

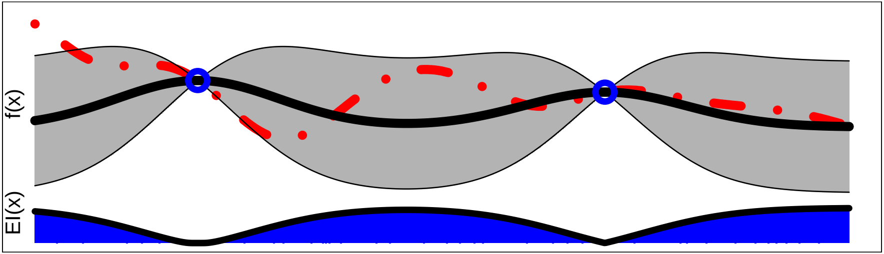
- Different inputs may have **different computational costs**, *e.g.*, training a neural network of increasing hidden layers and units.
- Better to do **cheap evaluations** before expensive ones!
- The evaluation costs are **unknown** but they can be **recorded** and then **modeled** with an additional **Gaussian process**.

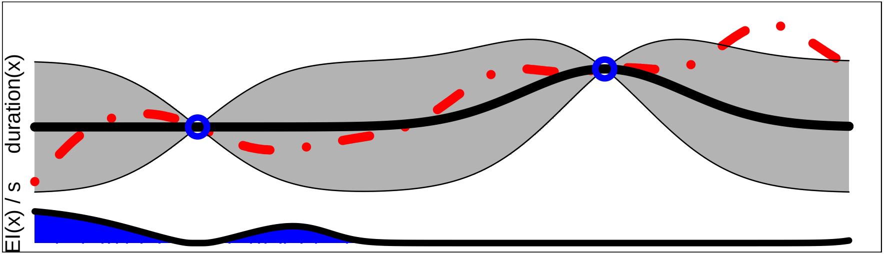
Expected Improvement per-second:

$$\alpha(\mathbf{x}) = \frac{\sigma(\mathbf{x})\left(\gamma(\mathbf{x})\Phi\left(\gamma(\mathbf{x})\right) + \phi(\gamma(\mathbf{x}))\right)}{\exp\left\{\mu_{\text{log-time}}(\mathbf{x})\right\}}$$

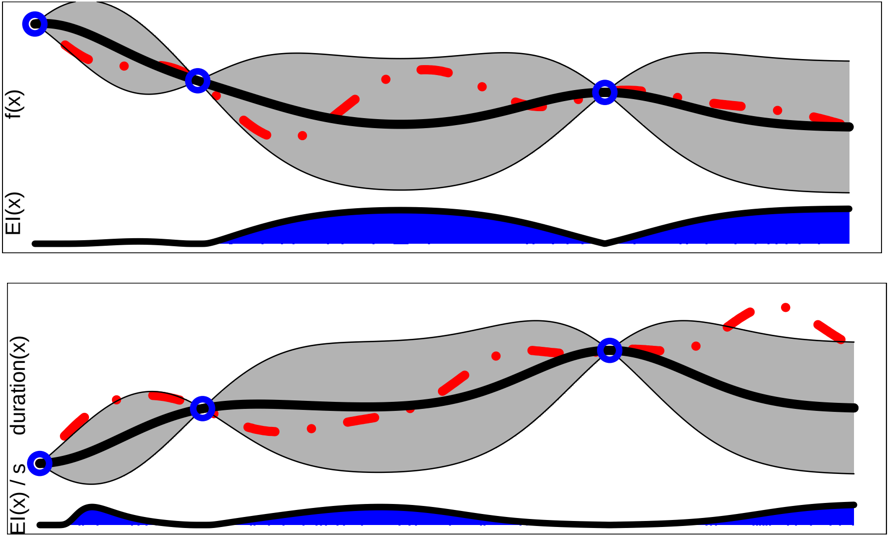
(Snoek *et al.*, 2012)

Cost-sensitive Bayesian Optimization

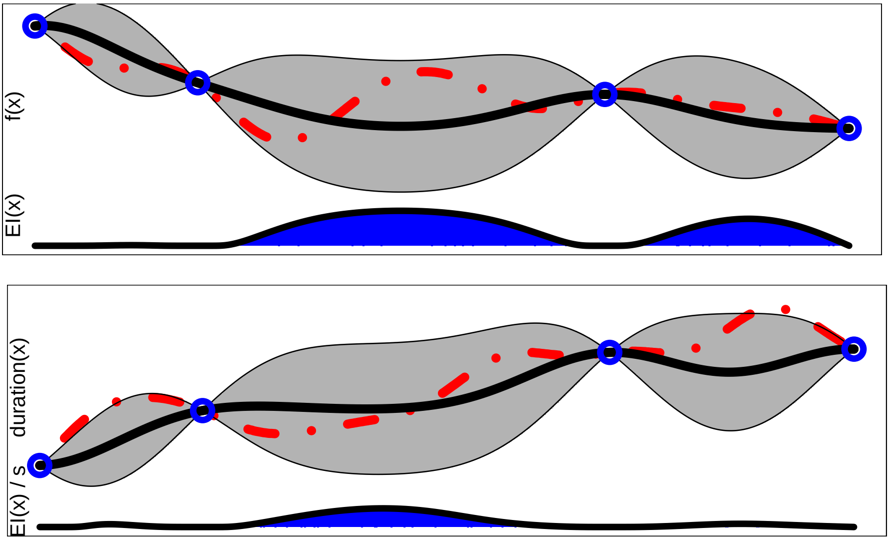




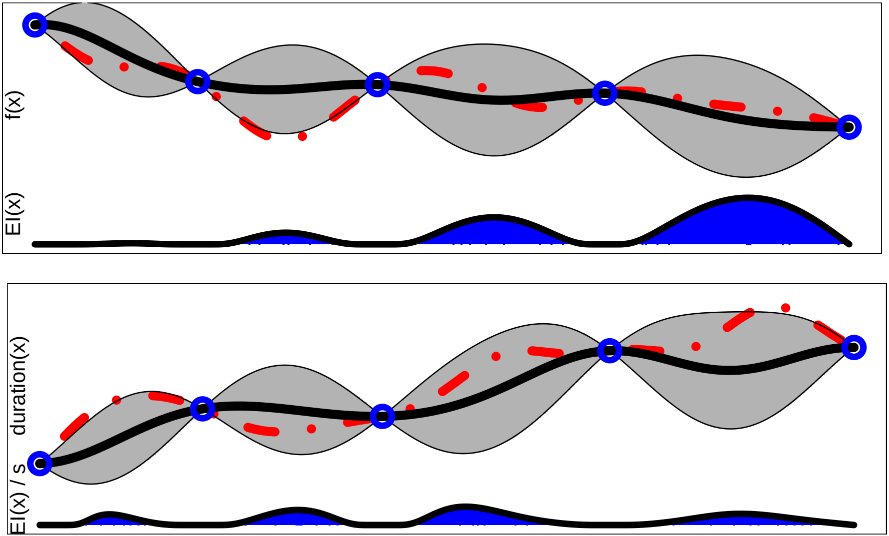
Cost-sensitive Bayesian Optimization



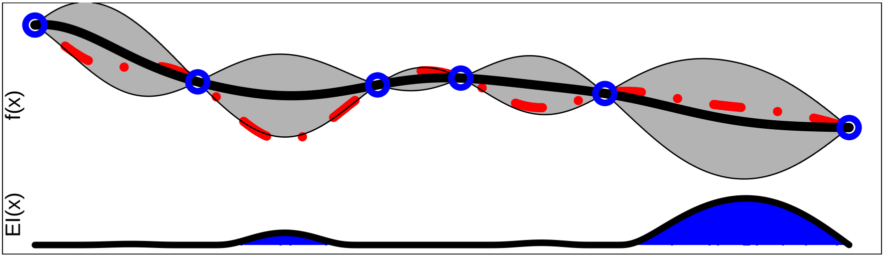
Cost-sensitive Bayesian Optimization

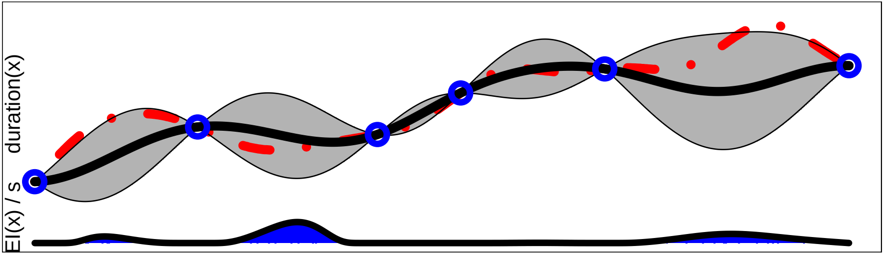


Cost-sensitive Bayesian Optimization

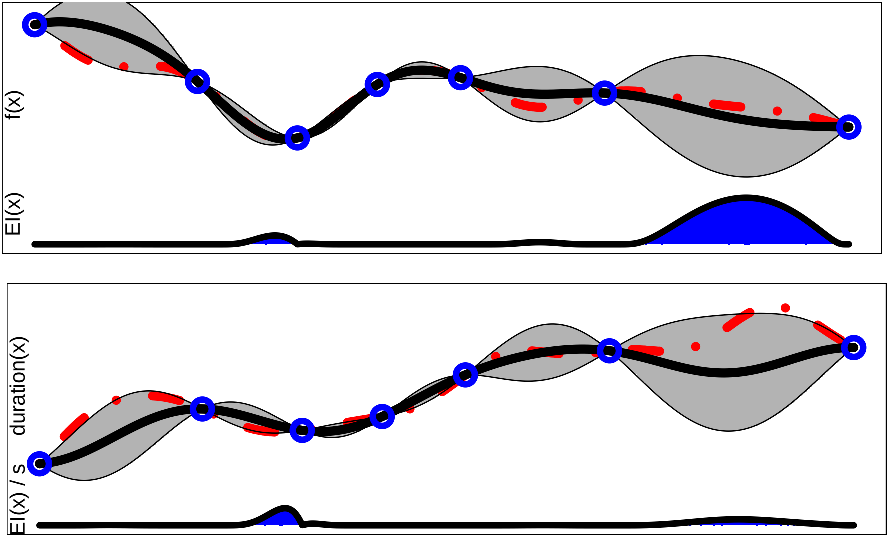


Cost-sensitive Bayesian Optimization

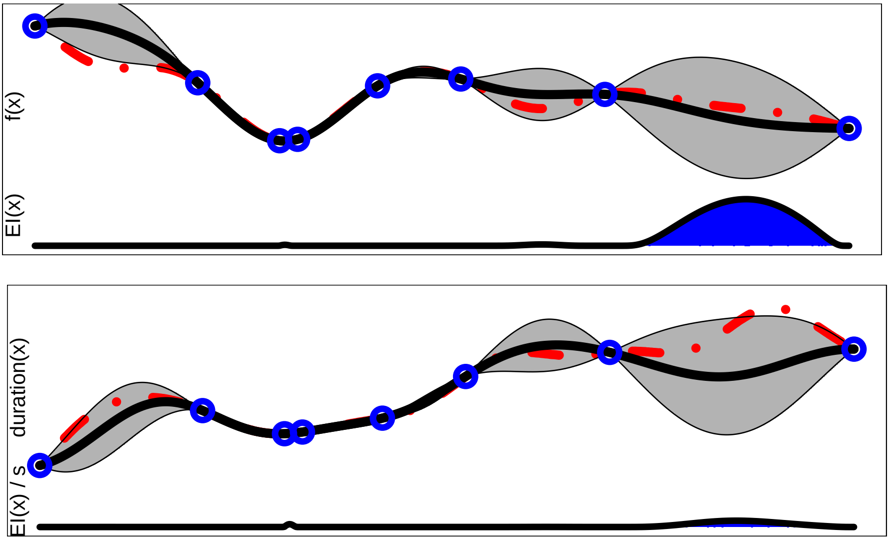




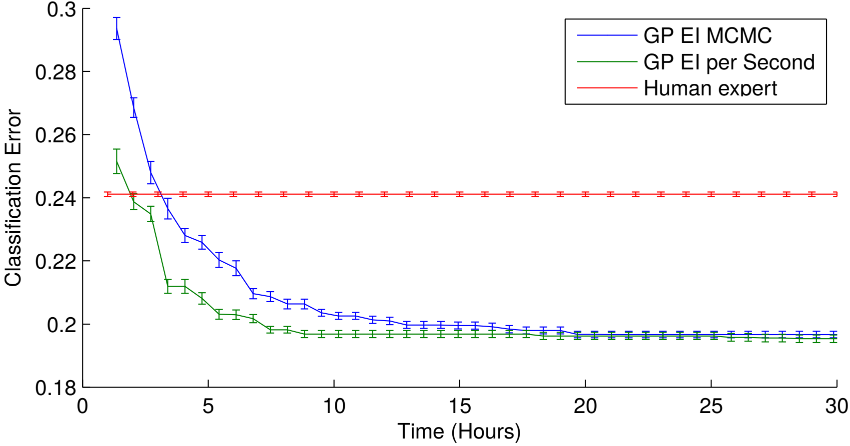
Cost-sensitive Bayesian Optimization



Cost-sensitive Bayesian Optimization



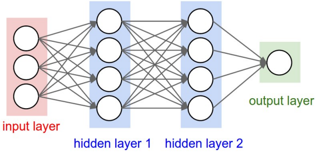
Cost-sensitive Bayesian Optimization

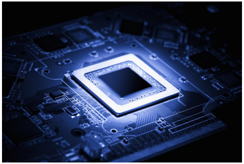


Deep neural network on the CIFAR dataset (Snoek *et al.*, 2012)

Several Objectives and Constraints

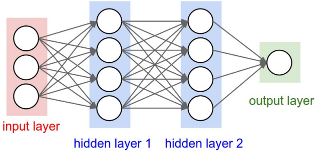
Optimal design of **hardware accelerator** for neural network predictions.

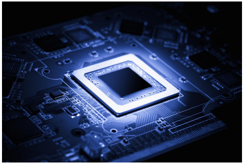




Several Objectives and Constraints

Optimal design of **hardware accelerator** for neural network predictions.



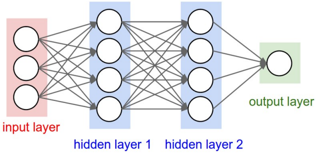


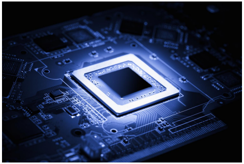
Goals:

- Minimize **prediction error**.
- Minimize **prediction time**.

Several Objectives and Constraints

Optimal design of **hardware accelerator** for neural network predictions.





Goals:

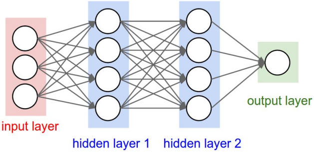
- Minimize **prediction error**.
- Minimize **prediction time**.

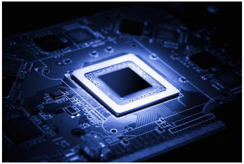
Constrained to:

- **Chip area** below a value.
- **Power consumption** below a level.

Several Objectives and Constraints

Optimal design of **hardware accelerator** for neural network predictions.



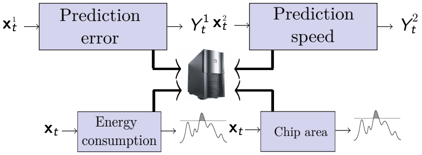


Goals:

- Minimize **prediction error**.
- Minimize **prediction time**.

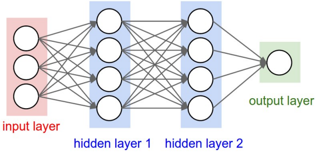
Constrained to:

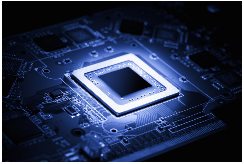
- **Chip area** below a value.
- **Power consumption** below a level.



Several Objectives and Constraints

Optimal design of **hardware accelerator** for neural network predictions.



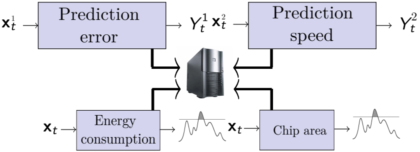


Goals:

- Minimize **prediction error**.
- Minimize **prediction time**.

Constrained to:

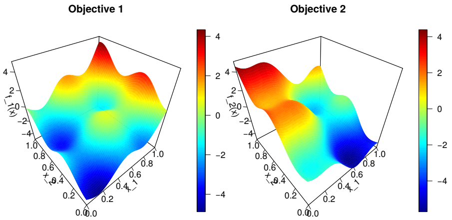
- **Chip area** below a value.
- **Power consumption** below a level.



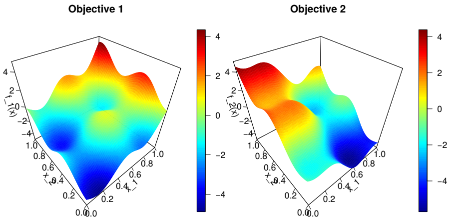
Challenges:

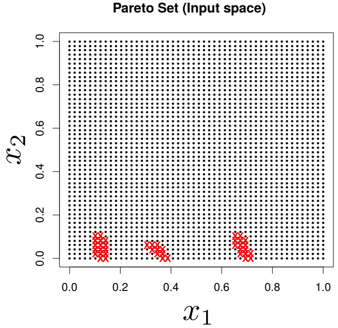
- **Complicated** constraints.
- **Conflictive** objectives.

Constrained Multi-Objective Optimization

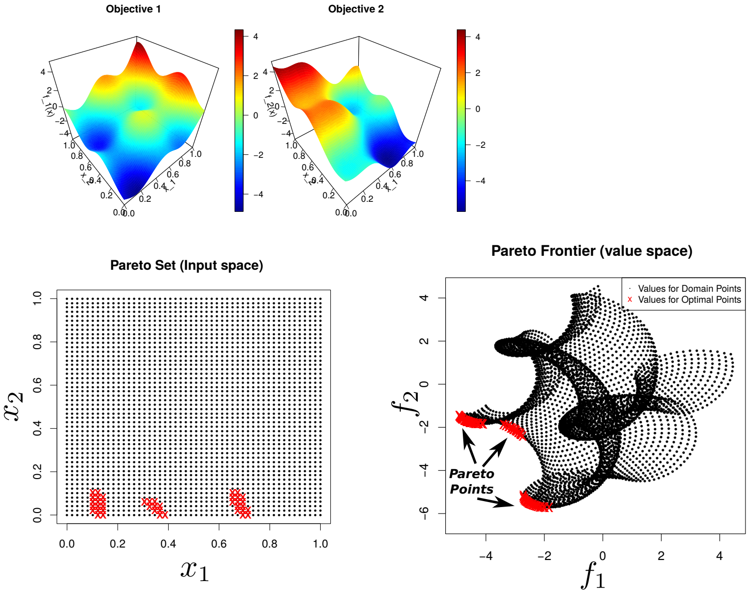


Constrained Multi-Objective Optimization

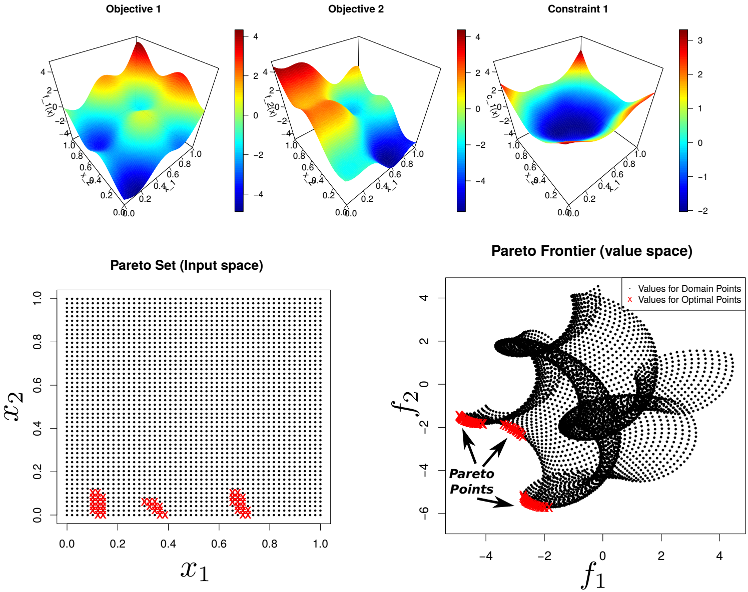




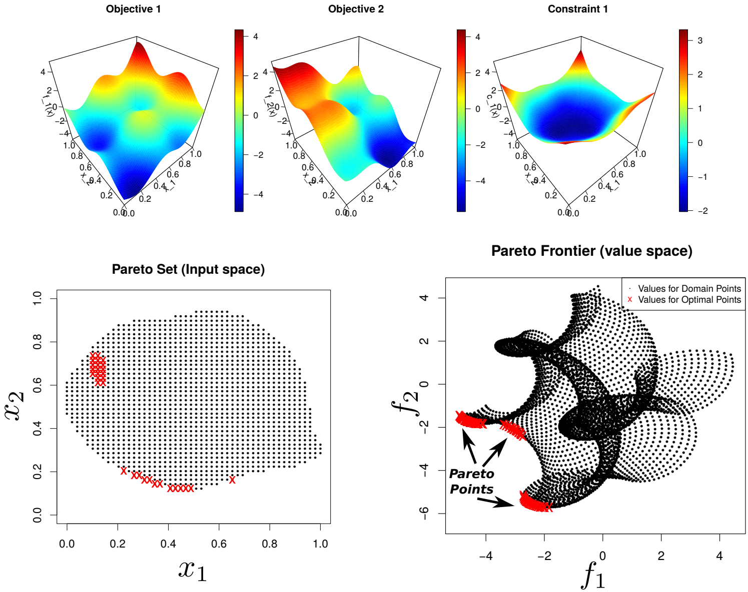
Constrained Multi-Objective Optimization



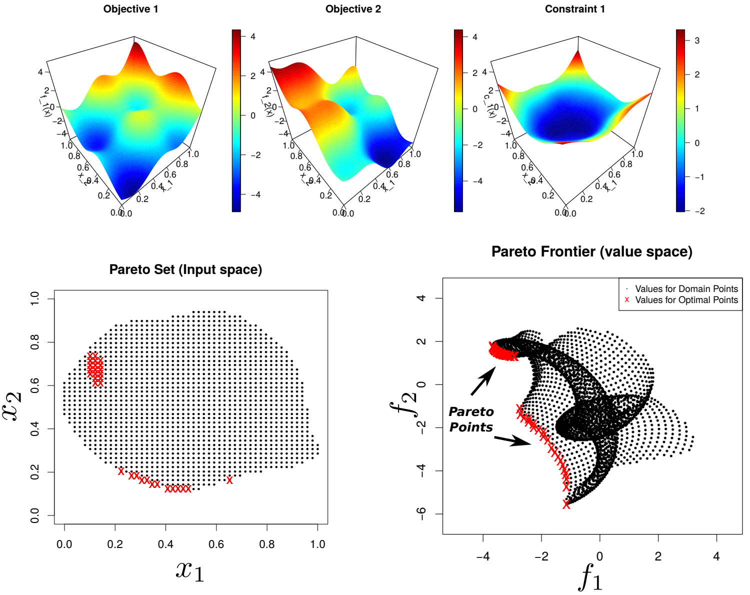
Constrained Multi-Objective Optimization



Constrained Multi-Objective Optimization



Constrained Multi-Objective Optimization



Bayesian Optimization Methods

Additional challenges when dealing with several black-boxes.

Bayesian Optimization Methods

Additional challenges when dealing with several black-boxes.

- Simple approach: evaluate **all** the objectives and constraints at the **same input location**. Expected to be sub-optimal.

Bayesian Optimization Methods

Additional challenges when dealing with several black-boxes.

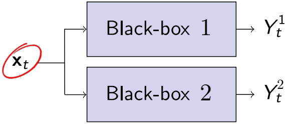
- Simple approach: evaluate **all** the objectives and constraints at the **same input location**. Expected to be sub-optimal.
- Advanced approach: make **intelligent** decisions about what **black-box** to **evaluate next** and on **which location**.

Bayesian Optimization Methods

Additional challenges when dealing with several black-boxes.

- Simple approach: evaluate **all** the objectives and constraints at the **same input location**. Expected to be sub-optimal.
- Advanced approach: make **intelligent** decisions about what **black-box** to **evaluate next** and on **which location**.

Coupled evaluations

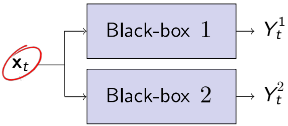


Bayesian Optimization Methods

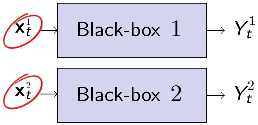
Additional challenges when dealing with several black-boxes.

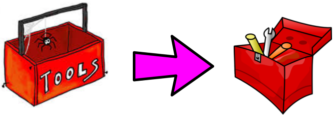
- Simple approach: evaluate **all** the objectives and constraints at the **same input location**. Expected to be sub-optimal.
- Advanced approach: make **intelligent** decisions about what **black-box** to **evaluate next** and on **which location**.

Coupled evaluations



Decoupled evaluations





Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information-based Approach

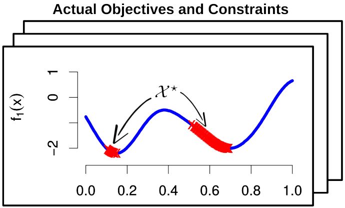
The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

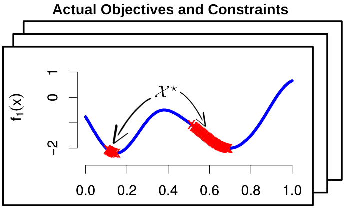
Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

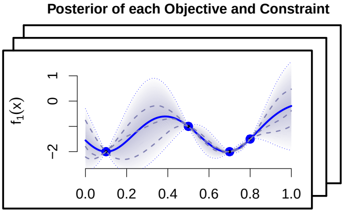


Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.



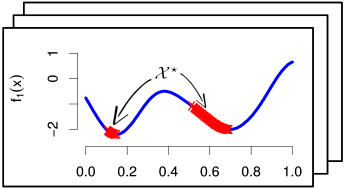


Information-based Approach

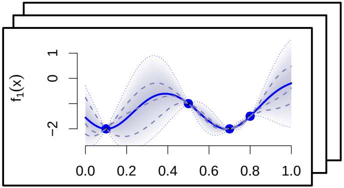
The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

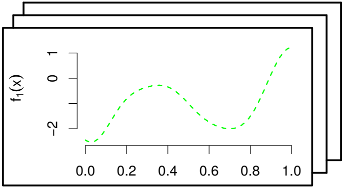
Actual Objectives and Constraints



Posterior of each Objective and Constraint



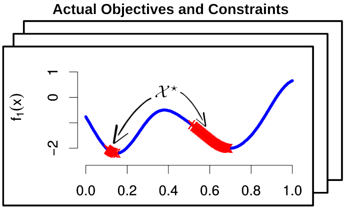
Optimized Samples Drawn from the Posterior

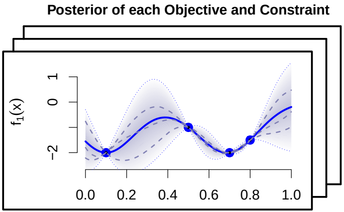


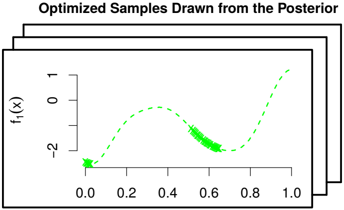
Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.



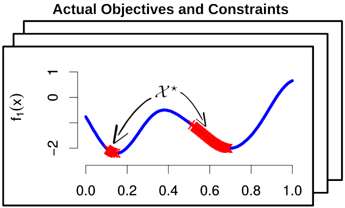


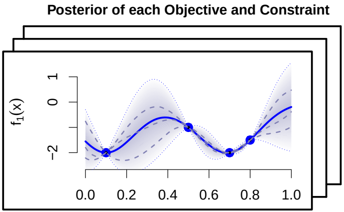


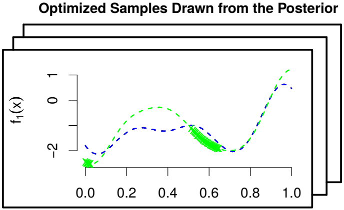
Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star | \mathcal{D}_N)$.





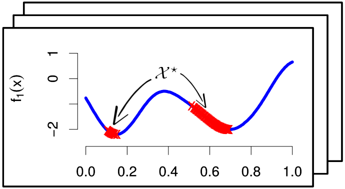


Information-based Approach

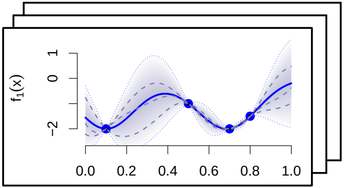
The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

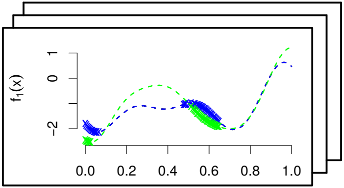
Actual Objectives and Constraints



Posterior of each Objective and Constraint



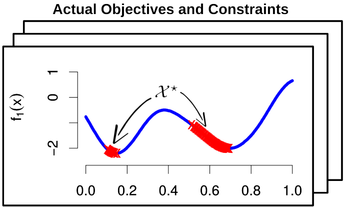
Optimized Samples Drawn from the Posterior

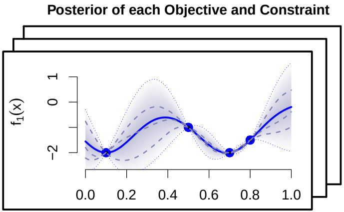


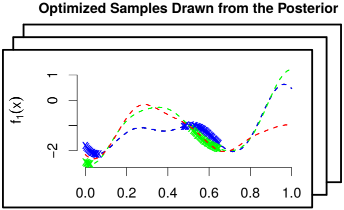
Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star | \mathcal{D}_N)$.





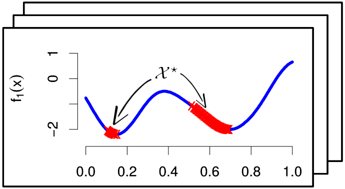


Information-based Approach

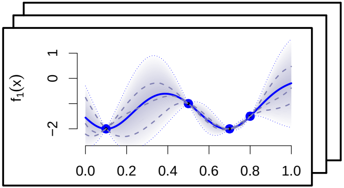
The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

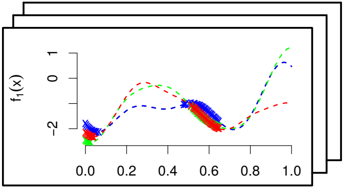
Actual Objectives and Constraints



Posterior of each Objective and Constraint



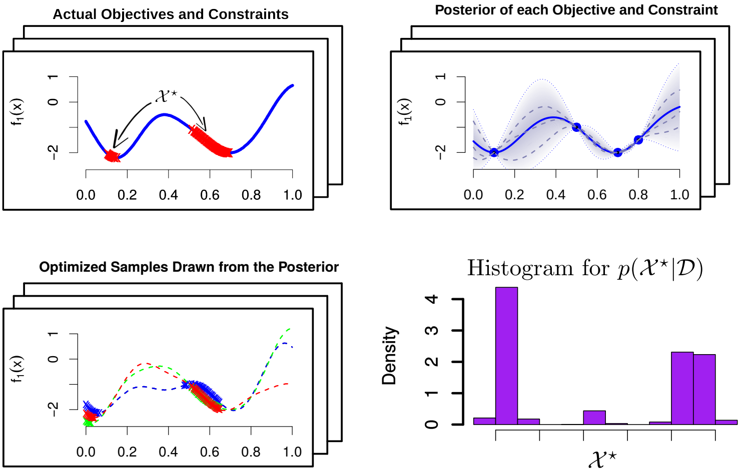
Optimized Samples Drawn from the Posterior



Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.



Information-based Approach

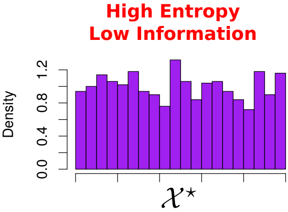
The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

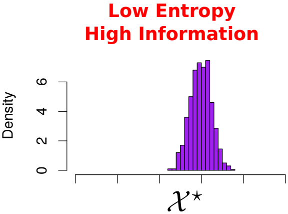
Information is measured by the **entropy** of $p(\mathcal{X}^\star | \mathcal{D}_N)$.

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.

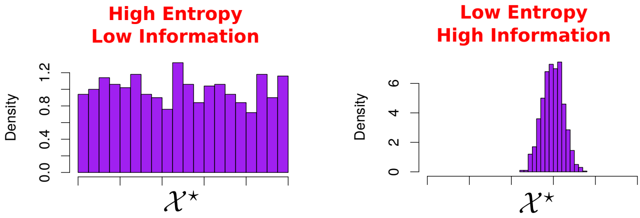




Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.



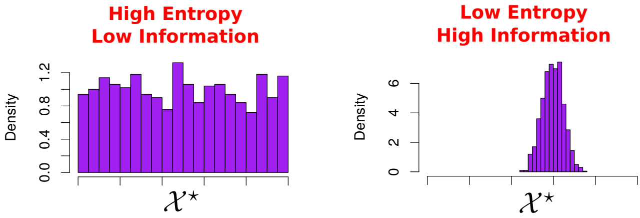
The acquisition function is

$$\alpha(\mathbf{x}) = \mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \quad (1)$$

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.



The acquisition function is

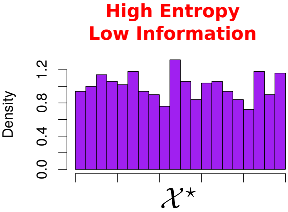
$$\alpha(\mathbf{x}) = \mathrm{H}\left[\mathcal{X}^\star\middle|\mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathcal{X}^\star\middle|\mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right]\middle|\mathcal{D}_t, \mathbf{x}\right] \quad (1)$$

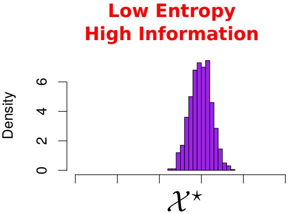
How much we know about $\mathcal{X}^\star$ now.

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.





The acquisition function is

$$\alpha(\mathbf{x}) = \mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \quad (1)$$

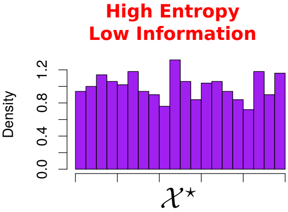
How much we know about $\mathcal{X}^\star$now.

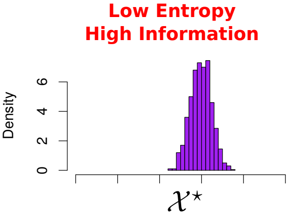
How much we will know about $\mathcal{X}^\star$after collecting y at $\mathbf{x}$.

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.





The acquisition function is

$$\alpha(\mathbf{x}) = \mathrm{H}\left[\mathcal{X}^\star\middle|\mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathcal{X}^\star\middle|\mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right]\middle|\mathcal{D}_t, \mathbf{x}\right] \quad (1)$$

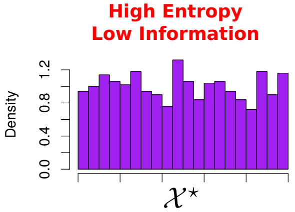
How much we know about $\mathcal{X}^\star$now.

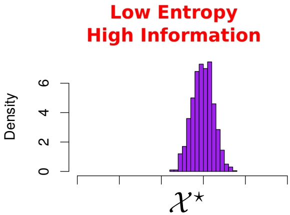
How much we will know about $\mathcal{X}^\star$after collecting y at $\mathbf{x}$.

Information-based Approach

The Pareto set $\mathcal{X}^\star$ in the feasible space is a **random variable**!

Information is measured by the **entropy** of $p(\mathcal{X}^\star|\mathcal{D}_N)$.





The acquisition function is

$$\alpha(\mathbf{x}) = \mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \quad (1)$$

How much we know about $\mathcal{X}^\star$ now.

How much we will know about $\mathcal{X}^\star$ after collecting y at $\mathbf{x}$.

Computing (1) is **very difficult in practice**!

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\big[\mathcal{X}^\star \big| \mathcal{D}_t\big] - \mathbb{E}_{\mathbf{y}}\Big[\mathsf{H}\big[\mathcal{X}^\star \big| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\big] \Big| \mathcal{D}_t, \mathbf{x}\Big] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\big[\mathbf{y} \big| \mathcal{D}_t, \mathbf{x}\big] - \mathbb{E}_{\mathcal{X}^\star}\Big[\mathsf{H}\big[\mathbf{y} \big| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\big] \Big| \mathcal{D}_t, \mathbf{x}\Big] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star | \mathcal{D}_t$

Predictive Entropy Search (PES)

We **swap $\mathbf{y}$ and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star | \mathcal{D}_t$

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star | \mathcal{D}_t)$

Factorized Gaussian approximation with **expectation propagation**. $\mathcal{X}^\star$ dominates any other point in $\mathcal{X}$.

(Minka, 2001)

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star | \mathcal{D}_t)$

Factorized Gaussian approximation with **expectation propagation**. $\mathcal{X}^\star$ dominates any other point in $\mathcal{X}$.

$$\alpha(\mathbf{x}) \approx \sum_{c=1}^{C} \log v_c^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{c=1}^{C} \log v_c^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right) + \sum_{k=1}^{K} \log v_k^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{k=1}^{K} \log v_k^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right)$$

(Minka, 2001)

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star|\mathcal{D}_t)$

Factorized Gaussian approximation with **expectation propagation**. $\mathcal{X}^\star$ dominates any other point in $\mathcal{X}$.

$$\alpha(\mathbf{x}) \approx \sum_{c=1}^{C} \log v_c^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{c=1}^{C} \log v_c^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right) + \sum_{k=1}^{K} \log v_k^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{k=1}^{K} \log v_k^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right) = \sum_{i=1}^{C+K} \alpha_i(\mathbf{x})$$

(Minka, 2001)

Predictive Entropy Search (PES)

We **swap y and $\mathcal{X}^\star$** to obtain a reformulation of the acquisition function.

$$\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathcal{X}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}, \mathbf{y}\}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathbf{y}, \mathcal{X}^\star) \quad \text{(ESMOC)}$$

$$\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathcal{X}^\star}\left[\mathsf{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathcal{X}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathsf{MI}(\mathcal{X}^\star, \mathbf{y}) \quad \text{(PESMOC)}$$

Gaussian distribution

Approximated by sampling from $p(\mathcal{X}^\star|\mathcal{D}_t)$

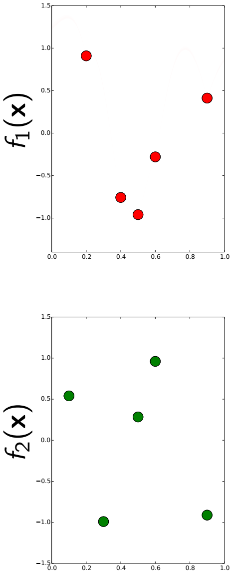
Factorized Gaussian approximation with **expectation** ... $\mathcal{X}^\star$ dominates any ... .

One acquisition per black-box

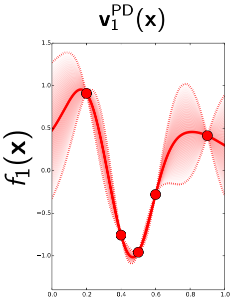
$$\alpha(\mathbf{x}) \approx \sum_{c=1}^{C} \log v_c^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{c=1}^{C} \log v_c^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right) + \sum_{k=1}^{K} \log v_k^{PD}(\mathbf{x}) - \frac{1}{M}\sum_{m=1}^{M}\left(\sum_{k=1}^{K} \log v_k^{CPD}(\mathbf{x}|\mathcal{X}^\star_{(m)})\right) = \sum_{i=1}^{C+K} \alpha_i(\mathbf{x})$$

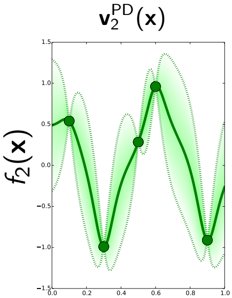
(Minka, 2001)

Example of PES' acquisition

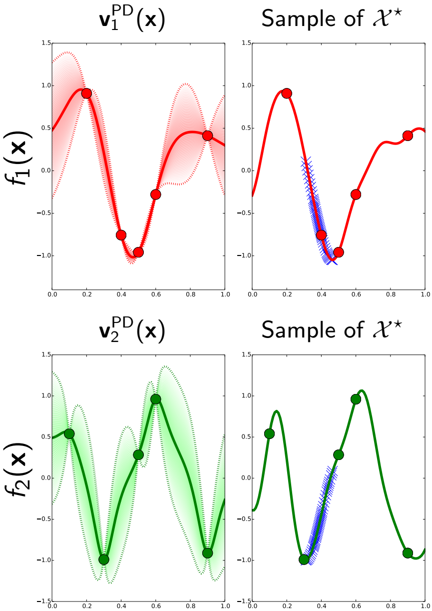


Example of PES' acquisition

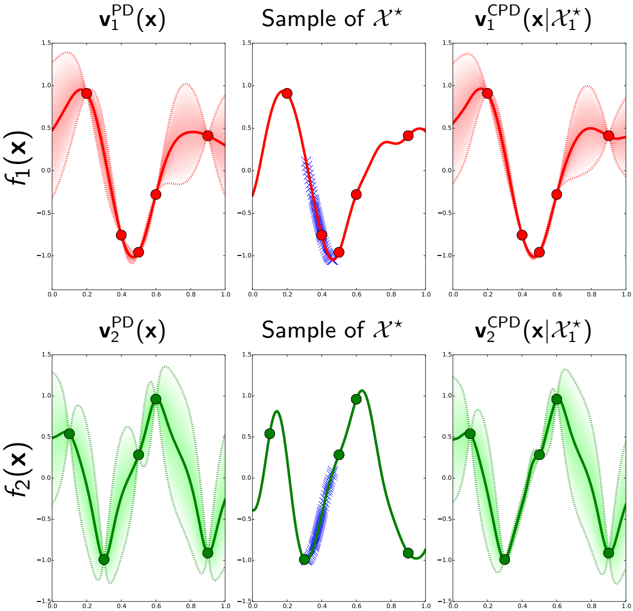




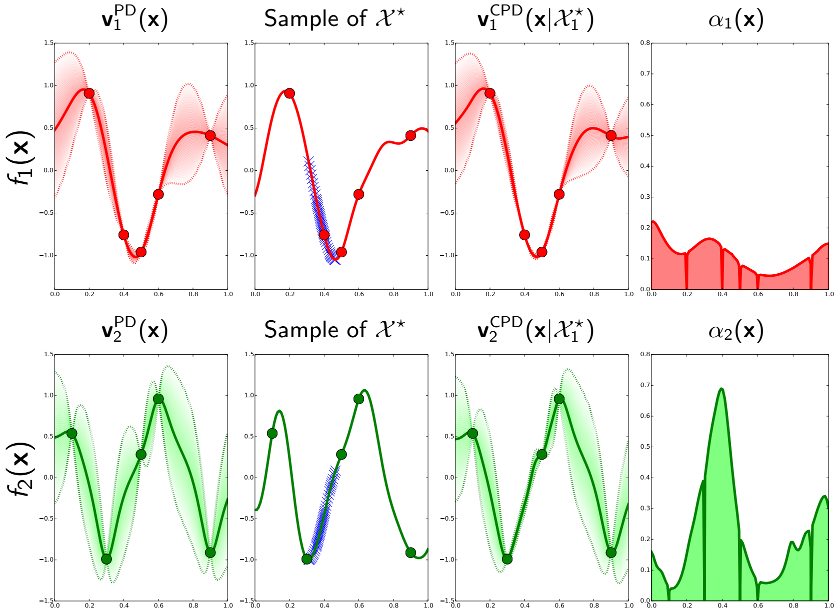
Example of PES' acquisition



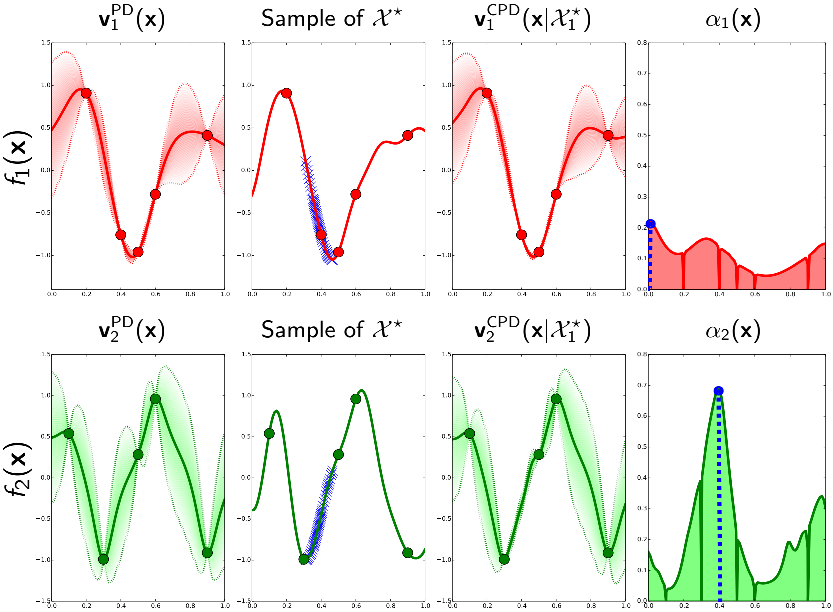
Example of PES' acquisition



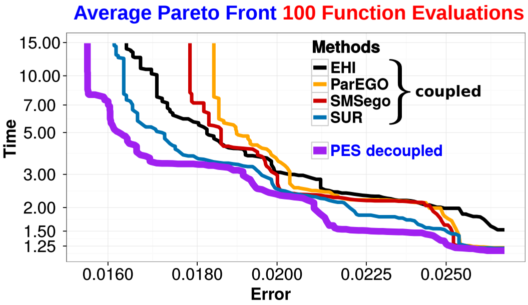
Example of PES' acquisition



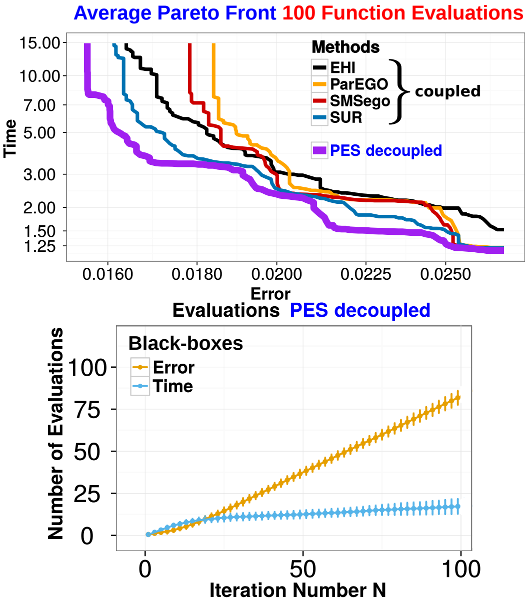
Example of PES' acquisition



Finding a Fast and Accurate Neural Network

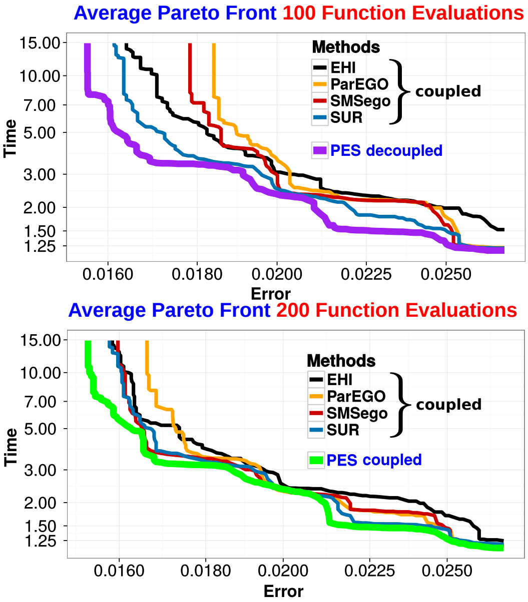


Finding a Fast and Accurate Neural Network



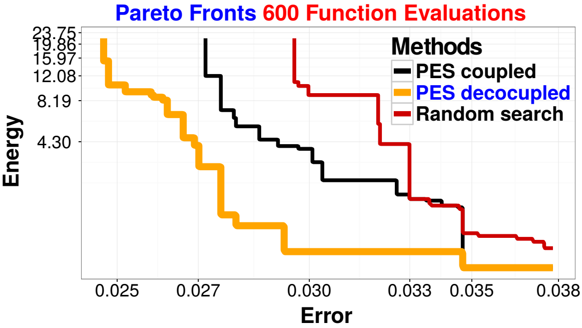
(Hernández-Lobato *et al.*, 2016)

Finding a Fast and Accurate Neural Network

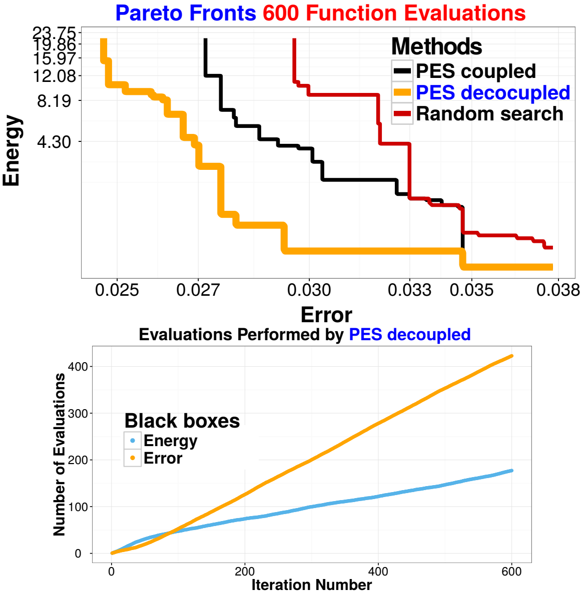


(Hernández-Lobato *et al.*, 2016)

Low energy hardware accelerator



Low energy hardware accelerator



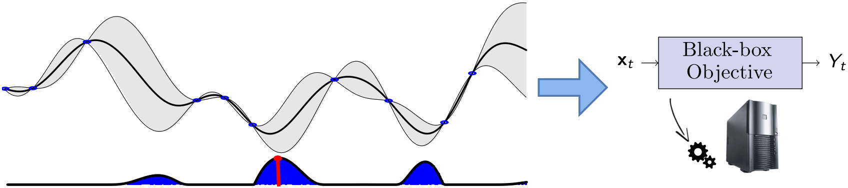
(Hernández-Lobato *et al.*, 2016)

Parallel Bayesian Optimization

Traditional Bayesian optimization is **sequential**!

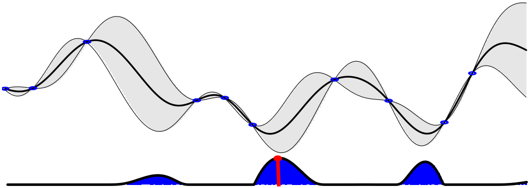
Parallel Bayesian Optimization

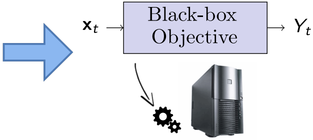
Traditional Bayesian optimization is **sequential**!



Parallel Bayesian Optimization

Traditional Bayesian optimization is **sequential**!

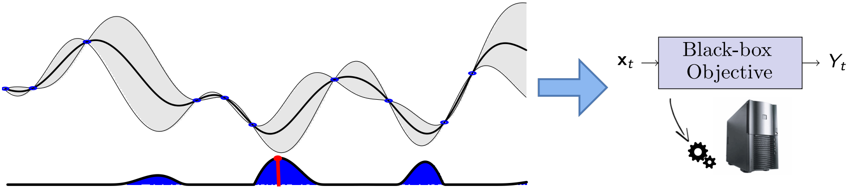




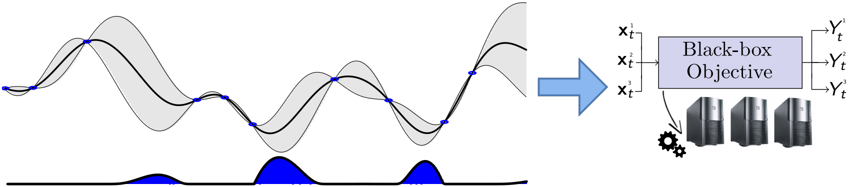
Computing clusters let us do **many things** at once!

Parallel Bayesian Optimization

Traditional Bayesian optimization is **sequential**!

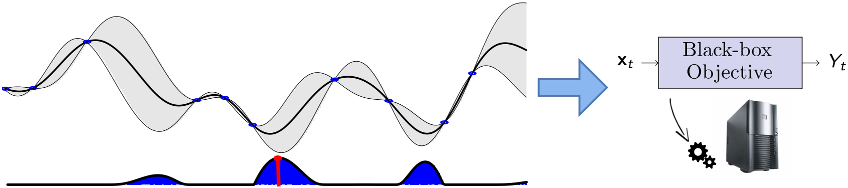


Computing clusters let us do **many things** at once!

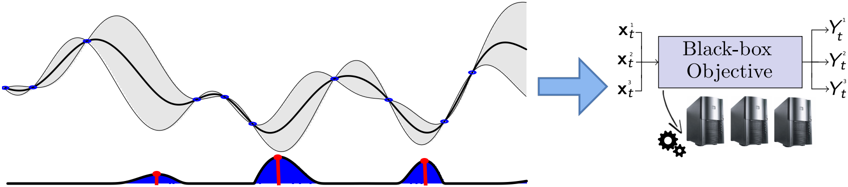


Parallel Bayesian Optimization

Traditional Bayesian optimization is **sequential**!

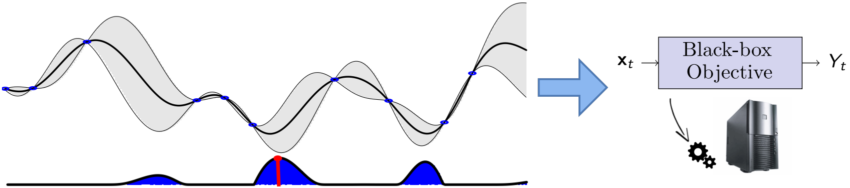


Computing clusters let us do **many things** at once!

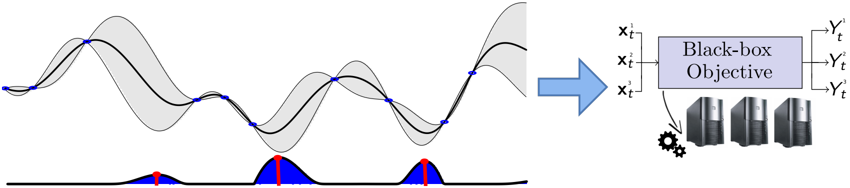


Parallel Bayesian Optimization

Traditional Bayesian optimization is **sequential**!



Computing clusters let us do **many things** at once!



Parallel experiments should be highly informative but different!

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathsf{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathsf{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \color{red}{\mathrm{MI}(\mathbf{y}, \mathbf{x}^\star)} \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \color{red}{\mathrm{MI}(\mathbf{x}^\star, \mathbf{y})} \quad \text{(Parallel PES)}$$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

Approximated by sampling from $p(\mathbf{x}^\star | \mathcal{D}_t)$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

Approximated by sampling from $p(\mathbf{x}^\star | \mathcal{D}_t)$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

Approximated by sampling from $p(\mathbf{x}^\star | \mathcal{D}_t)$

Multivariate Gaussian approximation with **expectation propagation**
$\mathbf{x}^\star$ is better than any other point in $\mathcal{X}$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

Approximated by sampling from $p(\mathbf{x}^\star|\mathcal{D}_t)$

Multivariate Gaussian approximation with **expectation propagation**
$\mathbf{x}^\star$ is better than any other point in $\mathcal{X}$

$$\alpha(\mathcal{S}_t) = \log|\mathbf{V}^{\mathrm{PD}}(\mathcal{S}_t)| - \frac{1}{M}\sum_{m=1}^{M} \log|\mathbf{V}^{\mathrm{CPD}}(\mathcal{S}_t|\mathbf{x}^\star_{(m)})|$$

(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search

Choose a set Q points $\mathcal{S}_t = \{\mathbf{x}_q\}_{q=1}^{Q}$ to minimize the entropy of $\mathbf{x}^\star$.

$$\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t\right] - \mathbb{E}_{\mathbf{y}}\left[\mathrm{H}\left[\mathbf{x}^\star \middle| \mathcal{D}_t \cup \{\mathbf{x}_q, y_q\}_{q=1}^{Q}\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{y}, \mathbf{x}^\star) \quad \text{(Parallel ES)}$$

$$\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}\right] - \mathbb{E}_{\mathbf{x}^\star}\left[\mathrm{H}\left[\mathbf{y} \middle| \mathcal{D}_t, \mathbf{x}, \mathbf{x}^\star\right] \middle| \mathcal{D}_t, \mathbf{x}\right] \equiv \mathrm{MI}(\mathbf{x}^\star, \mathbf{y}) \quad \text{(Parallel PES)}$$

Multi-variate Gaussian distribution

Approximated by sampling from $p(\mathbf{x}^\star | \mathcal{D}_t)$

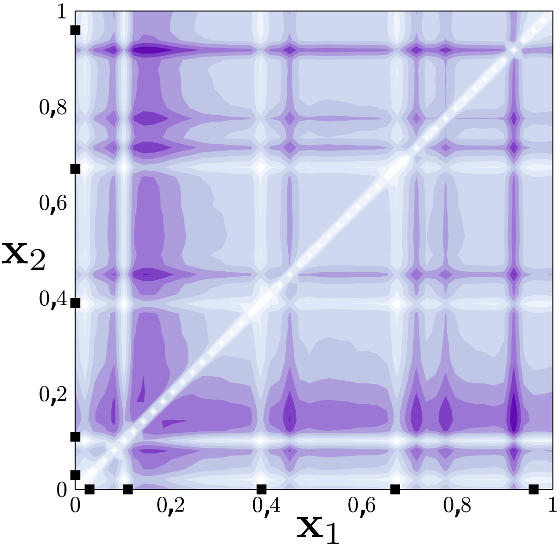
Multivariate Gaussian approximation with **expectation propagation**
$\mathbf{x}^\star$ is better than any other point in $\mathcal{X}$

$$\boxed{\alpha(\mathcal{S}_t) = \log |\mathbf{V}^{\mathrm{PD}}(\mathcal{S}_t)| - \frac{1}{M} \sum_{m=1}^{M} \log |\mathbf{V}^{\mathrm{CPD}}(\mathcal{S}_t | \mathbf{x}^\star_{(m)})|}$$

It is possible to compute the gradient of $\alpha(\cdot)$ w.r.t. each $\mathbf{x}_q \in \mathcal{S}_t$!

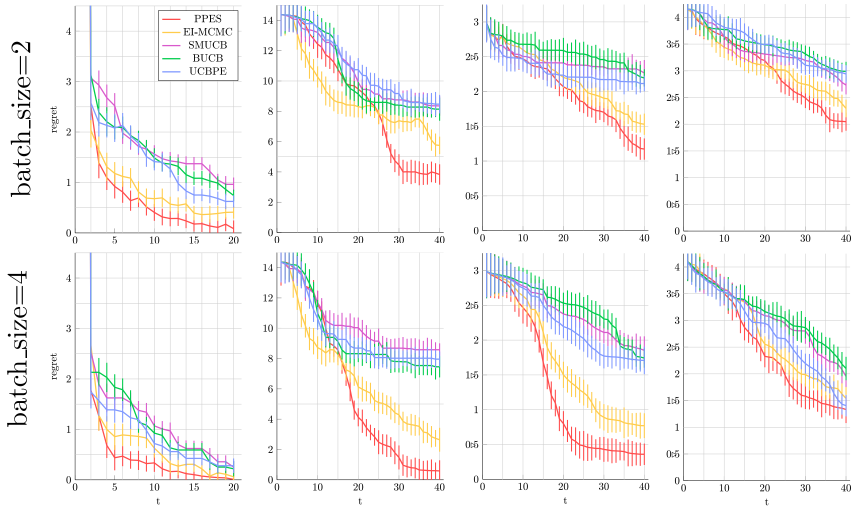
(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search: Level Curves



(Shah and Ghahramani, 2015)

Parallel Predictive Entropy Search: Results



(a) `boston` (b) `hydrogen` (c) `rocket` (d) `robot`

(Shah and Ghahramani, 2015)

BO with Integer-valued and Categorical Variables

Standard GP assume continuous input variables which makes BO with integer-valued or categorical challenging.

BO with Integer-valued and Categorical Variables

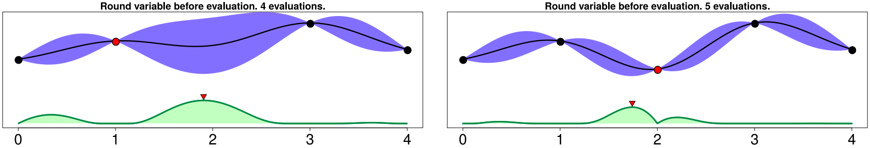
Standard GP assume continuous input variables which makes BO with integer-valued or categorical challenging.

A naive approach is to round the suggested value to the closest integer or to the closest one-hot encoding.

BO with Integer-valued and Categorical Variables

Standard GP assume continuous input variables which makes BO with integer-valued or categorical challenging.

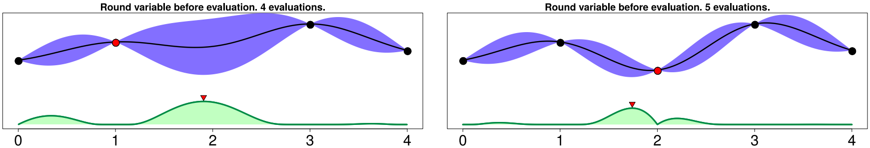
A naive approach is to round the suggested value to the closest integer or to the closest one-hot encoding.



BO with Integer-valued and Categorical Variables

Standard GP assume continuous input variables which makes BO with integer-valued or categorical challenging.

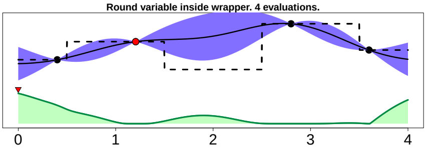
A naive approach is to round the suggested value to the closest integer or to the closest one-hot encoding.

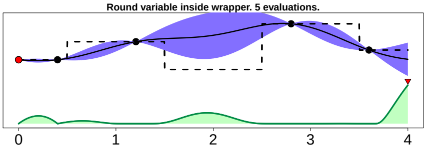


The BO algorithm may get stuck and may always perform the next evaluation at the same input location!

BO with Integer-valued and Categorical Variables

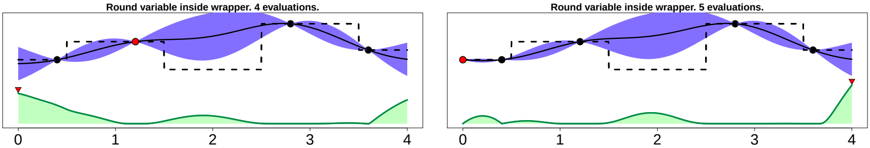
Rounding inside of the wrapper works but makes the objective flat!





BO with Integer-valued and Categorical Variables

Rounding inside of the wrapper works but makes the objective flat!



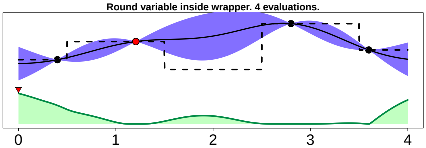
A modified GP covariance function accounts for this:

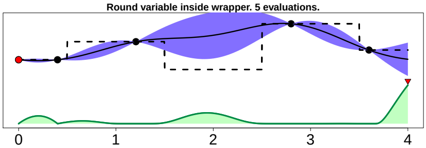
$$C_{\text{new}}(\mathbf{x}_n, \mathbf{x}_{n'}) = C(T(\mathbf{x}_n), T(\mathbf{x}_{n'}); \boldsymbol{\theta})$$

where $T(\cdot)$ does the rounding to the closest integer or one-hot encoding.

BO with Integer-valued and Categorical Variables

Rounding inside of the wrapper works but makes the objective flat!

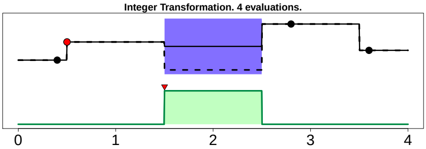


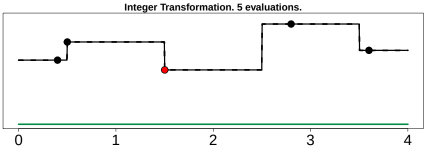


A modified GP covariance function accounts for this:

$$C_{\text{new}}(\mathbf{x}_n, \mathbf{x}_{n'}) = C(T(\mathbf{x}_n), T(\mathbf{x}_{n'}); \boldsymbol{\theta})$$

where $T(\cdot)$ does the rounding to the closest integer or one-hot encoding.



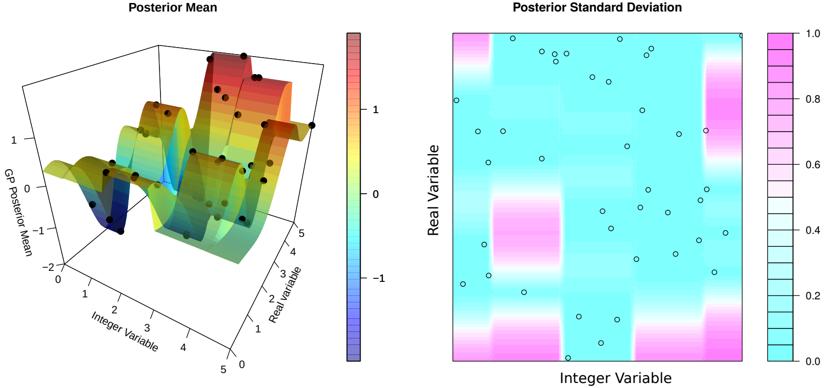


BO with Integer-valued and Categorical Variables

The GP predictive distribution is constant across all variables that lead to the same integer or one-hot-encoding.

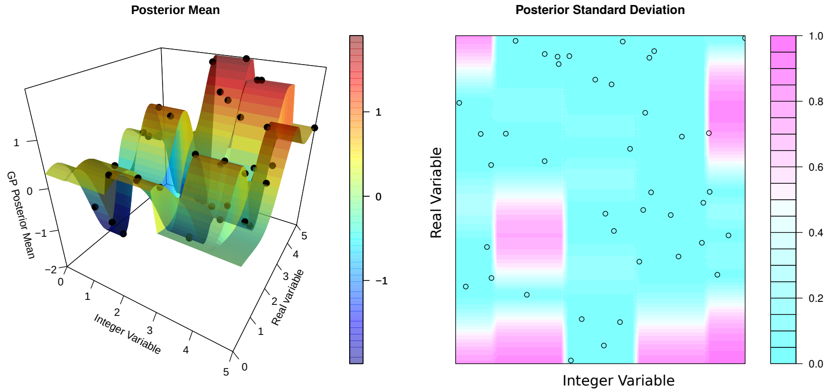
BO with Integer-valued and Categorical Variables

The GP predictive distribution is constant across all variables that lead to the same integer or one-hot-encoding.



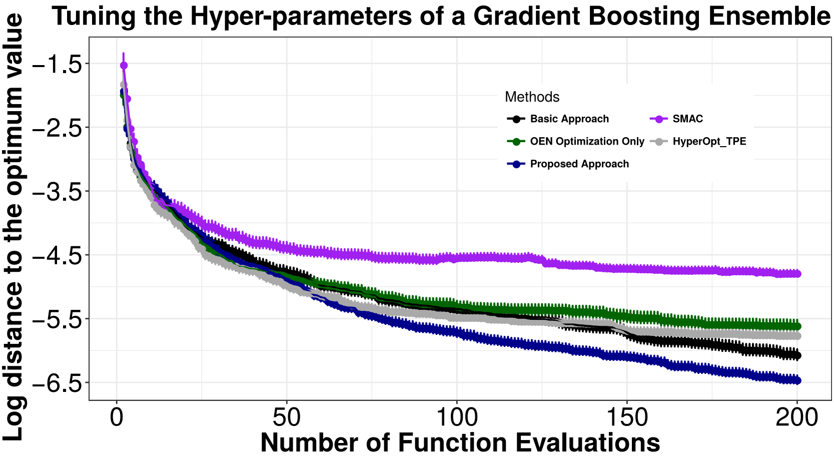
BO with Integer-valued and Categorical Variables

The GP predictive distribution is constant across all variables that lead to the same integer or one-hot-encoding.



Similar results for categorical variables!

BO with Integer-valued and Categorical Variables



One continuous variable and two integer-valued variables.

Freeze-Thaw Bayesian Optimization

Common aspects of many machine learning algorithms:

Freeze-Thaw Bayesian Optimization

Common aspects of many machine learning algorithms:

1. A minimization step must be performed with, *e.g.*, gradient descent.

Freeze-Thaw Bayesian Optimization

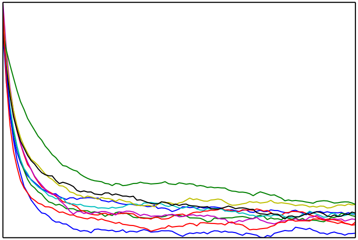
Common aspects of many machine learning algorithms:

1. A minimization step must be performed with, *e.g.*, gradient descent.
2. There are hyper-parameters that impact the final performance.

Freeze-Thaw Bayesian Optimization

Common aspects of many machine learning algorithms:

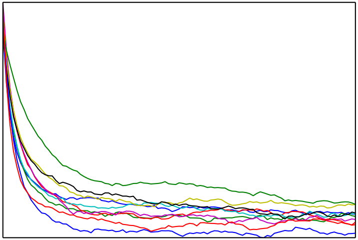
1. A minimization step must be performed with, *e.g.*, gradient descent.
2. There are hyper-parameters that impact the final performance.



Freeze-Thaw Bayesian Optimization

Common aspects of many machine learning algorithms:

1. A minimization step must be performed with, *e.g.*, gradient descent.
2. There are hyper-parameters that impact the final performance.

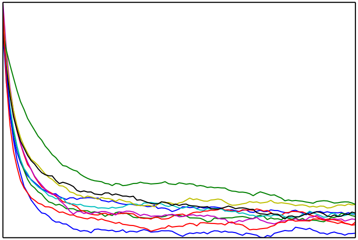


Can we use partial training information and a model to determine which hyper-parameter configuration is going to be optimal?

Freeze-Thaw Bayesian Optimization

Common aspects of many machine learning algorithms:

1. A minimization step must be performed with, *e.g.*, gradient descent.
2. There are hyper-parameters that impact the final performance.



Can we use partial training information and a model to determine which hyper-parameter configuration is going to be optimal?

Yes, that is precisely what Freeze-Thaw BO does!

(Swersky et al., 2014)

A GP Kernel for Training Curves

We want to specify a kernel that supports exponentially decaying functions of the form $\exp\{-\lambda t\}$ for $t, \lambda \geq 0$.

A GP Kernel for Training Curves

We want to specify a kernel that supports exponentially decaying functions of the form $\exp\{-\lambda t\}$ for $t, \lambda \geq 0$.

The covariance between inputs t and t' is:

$$C(t, t') = \int_0^\infty e^{-\lambda t} e^{-\lambda t'} \psi(\lambda; \alpha, \beta) d\lambda = \frac{\beta^\alpha}{(t + t' + \beta)^\alpha}$$

where $\psi(\lambda; \alpha, \beta)$ is a gamma distribution with parameters α and β.

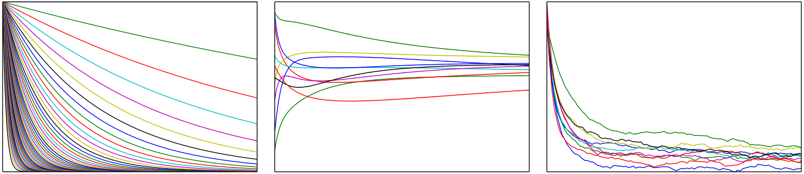
A GP Kernel for Training Curves

We want to specify a kernel that supports exponentially decaying functions of the form $\exp\{-\lambda t\}$ for $t, \lambda \geq 0$.

The covariance between inputs t and t' is:

$$C(t, t') = \int_0^\infty e^{-\lambda t} e^{-\lambda t'} \psi(\lambda; \alpha, \beta) d\lambda = \frac{\beta^\alpha}{(t + t' + \beta)^\alpha}$$

where $\psi(\lambda; \alpha, \beta)$ is a gamma distribution with parameters α and β.



(a) Exponential Decay Basis (b) Samples (c) Training Curve Samples

Inference on Asymptotic Values

A standard GP is used as the prior for the asymptotic values of each training curve.

Inference on Asymptotic Values

A standard GP is used as the prior for the asymptotic values of each training curve.

Hierarchical generative model:

$$p(\{\mathbf{y}_n\}_{n=1}^N | \{\mathbf{x}_n\}_{n=1}^N) = \int \left[\prod_{n=1}^N \mathcal{N}(\mathbf{y}_n | f_n \mathbf{1}, \mathbf{K}_{t_n}) \right] \mathcal{N}(\mathbf{f} | \mathbf{m}, \mathbf{K}_{\mathbf{x}}) d\mathbf{f}$$

where

$$\begin{aligned} \mathbf{x}_n &\equiv n \text{ configuration}\,, & \mathbf{y}_n &\equiv n \text{ observed curve}\,, \\ f_n &\equiv n \text{ asymptotic value}\,, & \mathbf{m} &\equiv \text{prior asymptotic mean values}\,, \\ \mathbf{K}_{t_n} &\equiv \text{covariances for curve values}\,, & \mathbf{K}_{\mathbf{x}} &\equiv \text{cov. for asymptotic values} \end{aligned}$$

Inference on Asymptotic Values

A standard GP is used as the prior for the asymptotic values of each training curve.

Hierarchical generative model:

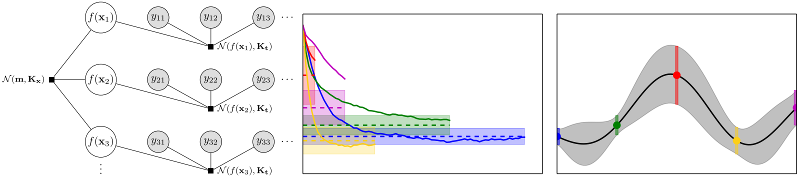
$$p(\{\mathbf{y}_n\}_{n=1}^N|\{\mathbf{x}_n\}_{n=1}^N) = \int \left[\prod_{n=1}^N \mathcal{N}(\mathbf{y}_n|f_n\mathbf{1},\mathbf{K}_{t_n})\right]\mathcal{N}(\mathbf{f}|\mathbf{m},\mathbf{K}_\mathbf{x})d\mathbf{f}$$

where

$$\begin{aligned}
\mathbf{x}_n &\equiv n \text{ configuration}\,, & \mathbf{y}_n &\equiv n \text{ observed curve}\,,\\
f_n &\equiv n \text{ asymptotic value}\,, & \mathbf{m} &\equiv \text{prior asymptotic mean values}\,,\\
\mathbf{K}_{t_n} &\equiv \text{covariances for curve values}\,, & \mathbf{K}_\mathbf{x} &\equiv \text{cov. for asymptotic values}
\end{aligned}$$

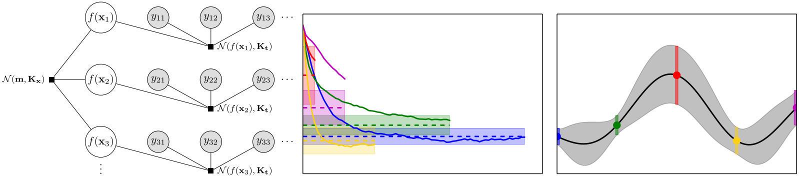
The joint distribution of $\{\mathbf{y}\}_{n=1}^N$ and $\mathbf{f}$ is Gaussian and hence so it the predictive distribution $p(\mathbf{f}|\{\mathbf{y}\}_{n=1}^N)$!

Inference on Asymptotic Values and BO



(a) Graphical Model (b) Training curve predictions (c) Asymptotic GP

Inference on Asymptotic Values and BO

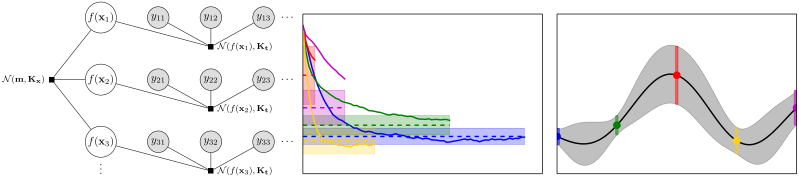


(a) Graphical Model (b) Training curve predictions (c) Asymptotic GP

Bayesian Optimization:

- $p(\mathbf{f}|\{\mathbf{y}_n\}_{n=1}^N, \{\mathbf{x}_n\}_{n=1}^N)$ determines asymptotic values.

Inference on Asymptotic Values and BO

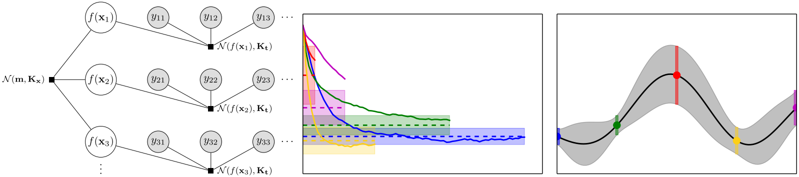


(a) Graphical Model (b) Training curve predictions (c) Asymptotic GP

Bayesian Optimization:

- $p(\mathbf{f}|\{\mathbf{y}_n\}_{n=1}^N, \{\mathbf{x}_n\}_{n=1}^N)$ determines asymptotic values.
- This distribution can be used to make intelligent decisions!

Inference on Asymptotic Values and BO

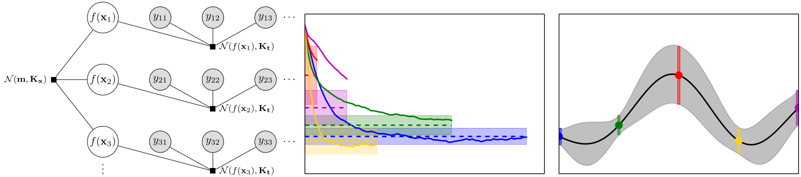


(a) Graphical Model (b) Training curve predictions (c) Asymptotic GP

Bayesian Optimization:

- $p(\mathbf{f} | \{\mathbf{y}_n\}_{n=1}^N, \{\mathbf{x}_n\}_{n=1}^N)$ determines asymptotic values.
- This distribution can be used to make intelligent decisions!
- Shall we train more one configuration or shall we start a new one?

Inference on Asymptotic Values and BO



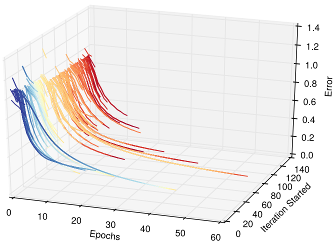
(a) Graphical Model (b) Training curve predictions (c) Asymptotic GP

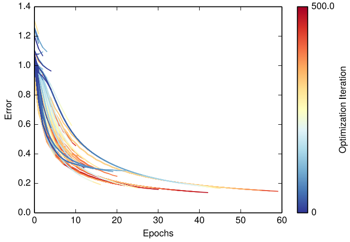
Bayesian Optimization:

- $p(\mathbf{f} | \{\mathbf{y}_n\}_{n=1}^N, \{\mathbf{x}_n\}_{n=1}^N)$ determines asymptotic values.
- This distribution can be used to make intelligent decisions!
- Shall we train more one configuration or shall we start a new one?
- A combination of EI and ES is used as the acquisition function.

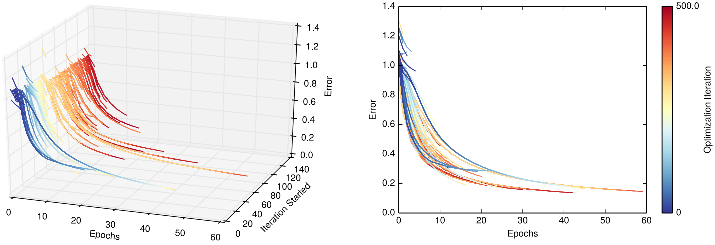
(Swersky et al., 2014)

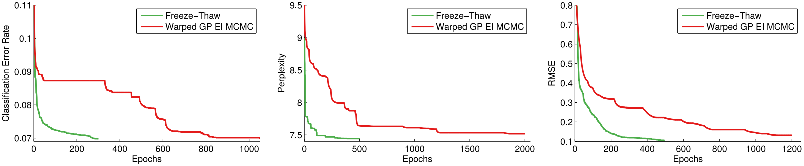
Freeze-Thaw BO in practice





Freeze-Thaw BO in practice





(a) Logistic Regression (b) Online LDA (c) PMF

(Swersky et al., 2014)

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Most successful approaches are based on inducing points:

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Most successful approaches are based on inducing points:

$$\overline{\mathbf{X}} \equiv \text{Matrix of } m \ll n \text{ inducing or pseudo-inputs.}$$

$$\mathbf{u} = f(\overline{\mathbf{X}}) \equiv \text{Inducing values / values of the process at } \overline{\mathbf{X}}.$$

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Most successful approaches are based on inducing points:

$$\overline{\mathbf{X}} \equiv \text{Matrix of } m \ll n \text{ inducing or pseudo-inputs.}$$

$$\mathbf{u} = f(\overline{\mathbf{X}}) \equiv \text{Inducing values / values of the process at } \overline{\mathbf{X}}.$$

The predictive distribution for $f^\star$ at a new point $\mathbf{x}^\star$ is:

$$p(f^\star|\mathcal{D}_n) \approx \int p(f^\star|\mathbf{u})q(\mathbf{u})d\mathbf{u} = \mathcal{N}(f^\star|\mu,\nu^2)$$

$$\mu = \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}\mathbf{m}\,, \quad \nu^2 = \kappa_{\mathbf{x}^\star,\mathbf{x}^\star} - \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}(\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}} - \mathbf{S})\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}\mathbf{k}_{\overline{\mathbf{X}},\mathbf{x}^\star}$$

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Most successful approaches are based on inducing points:

$$\overline{\mathbf{X}} \equiv \text{Matrix of } m \ll n \text{ inducing or pseudo-inputs.}$$

$$\mathbf{u} = f(\overline{\mathbf{X}}) \equiv \text{Inducing values / values of the process at } \overline{\mathbf{X}}.$$

The predictive distribution for $f^\star$ at a new point $\mathbf{x}^\star$ is:

$$p(f^\star|\mathcal{D}_n) \approx \int p(f^\star|\mathbf{u})q(\mathbf{u})d\mathbf{u} = \mathcal{N}(f^\star|\mu,\nu^2)$$

$$\mu = \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}^{-1}\mathbf{m}\,, \quad \nu^2 = \kappa_{\mathbf{x}^\star,\mathbf{x}^\star} - \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}^{-1}(\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}} - \mathbf{S})\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}^{-1}\mathbf{k}_{\overline{\mathbf{X}},\mathbf{x}^\star}$$

$$q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m},\mathbf{S}) \equiv \text{Gaussian approximation to } p(\mathbf{u}|\mathcal{D}_n).$$

Computational Cost of GPs and Other Models

- Exact inference with GP has cost in $\mathcal{O}(n^3)$.
- For large evaluation budgets one has to use approximations.

Most successful approaches are based on inducing points:

$$\overline{\mathbf{X}} \equiv \text{Matrix of } m \ll n \text{ inducing or pseudo-inputs.}$$

$$\mathbf{u} = f(\overline{\mathbf{X}}) \equiv \text{Inducing values / values of the process at } \overline{\mathbf{X}}.$$

The predictive distribution for $f^\star$ at a new point $\mathbf{x}^\star$ is:

$$p(f^\star|\mathcal{D}_n) \approx \int p(f^\star|\mathbf{u})q(\mathbf{u})d\mathbf{u} = \mathcal{N}(f^\star|\mu,\nu^2)$$

$$\mu = \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}\mathbf{m}\,, \quad \nu^2 = \kappa_{\mathbf{x}^\star,\mathbf{x}^\star} - \mathbf{k}_{\mathbf{x}^\star,\overline{\mathbf{X}}}\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}(\mathbf{K}_{\overline{\mathbf{X}},\overline{\mathbf{X}}} - \mathbf{S})\mathbf{K}^{-1}_{\overline{\mathbf{X}},\overline{\mathbf{X}}}\mathbf{k}_{\overline{\mathbf{X}},\mathbf{x}^\star}$$

$$q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{m},\mathbf{S}) \equiv \text{Gaussian approximation to } p(\mathbf{u}|\mathcal{D}_n).$$

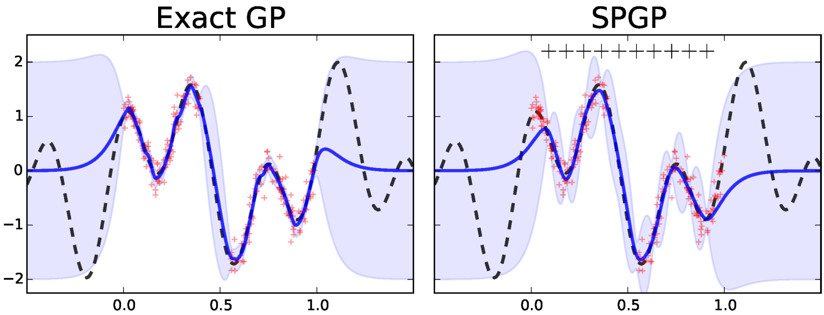
The computational cost is in $\mathcal{O}(nm^2)$!

Sparse GP based on Inducing Points

The approximate predictive distribution can be sub-optimal if the inducing points are not chosen carefully.

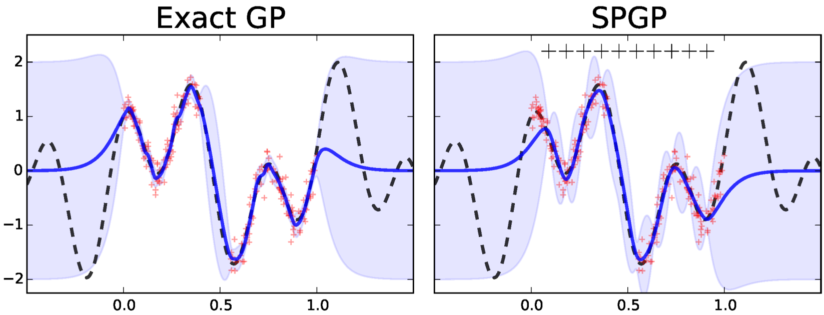
Sparse GP based on Inducing Points

The approximate predictive distribution can be sub-optimal if the inducing points are not chosen carefully.



Sparse GP based on Inducing Points

The approximate predictive distribution can be sub-optimal if the inducing points are not chosen carefully.



- Too small variance at the pseudo-inputs.
- Too big variance in between and away from pseudo-inputs.

(Shahriari et al., 2016)

Optimizing the Inducing Points

Two approaches:

Optimizing the Inducing Points

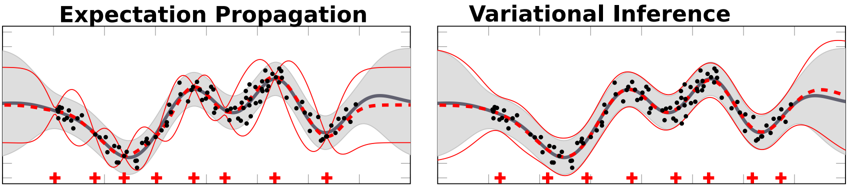
Two approaches:

- EP: optimize the marginal likelihood of an approximate GP model.
- VI: maximize fidelity to the original exact GP.

Optimizing the Inducing Points

Two approaches:

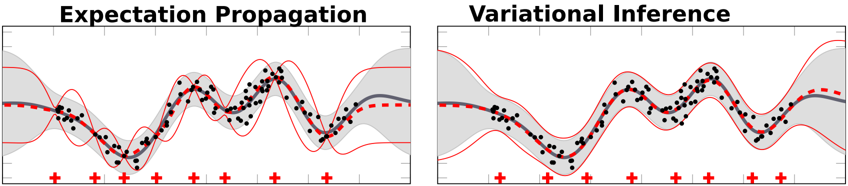
- EP: optimize the marginal likelihood of an approximate GP model.
- VI: maximize fidelity to the original exact GP.



Optimizing the Inducing Points

Two approaches:

- EP: optimize the marginal likelihood of an approximate GP model.
- VI: maximize fidelity to the original exact GP.



- EP: less local optima and easier to optimize, also less accurate.
- VI: more accurate, more local optima, more difficult to optimize.

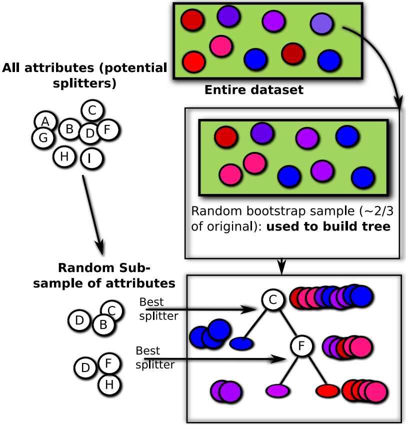
(Bui et al., 2017) (Bauer et al., 2016)

Other Models: Random Forest

Ensemble method where the predictors are random regression trees trained on random subsamples of the data.

Other Models: Random Forest

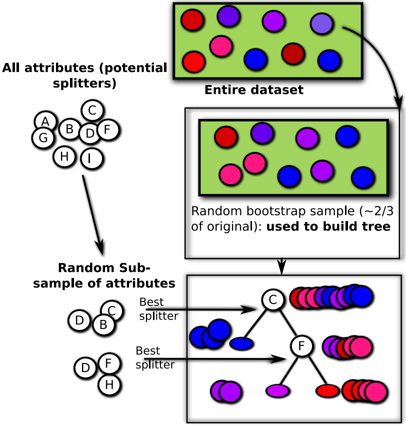
Ensemble method where the predictors are random regression trees trained on random subsamples of the data.



- Trees are grown on different bootstrap samples of the data.

Other Models: Random Forest

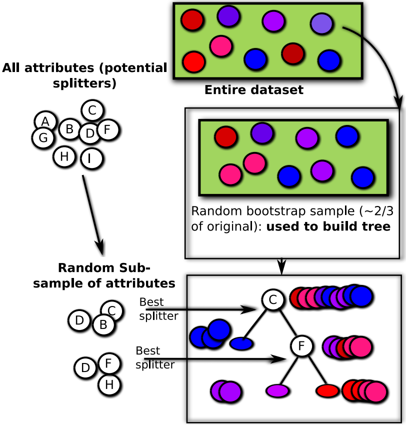
Ensemble method where the predictors are random regression trees trained on random subsamples of the data.



- Trees are grown on different bootstrap samples of the data.
- At each node the best splitter is chosen randomly.

Other Models: Random Forest

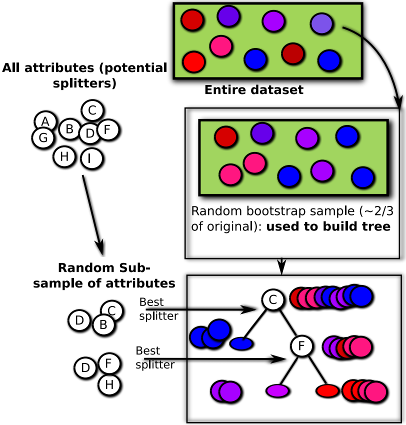
Ensemble method where the predictors are random regression trees trained on random subsamples of the data.



- Trees are grown on different bootstrap samples of the data.
- At each node the best splitter is chosen randomly.
- Leaf nodes predict the average value of the points reaching that node.

Other Models: Random Forest

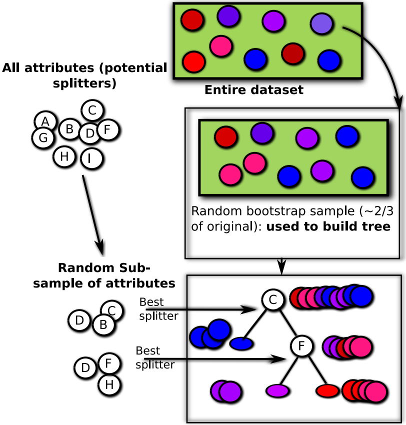
Ensemble method where the predictors are random regression trees trained on random subsamples of the data.



- Trees are grown on different bootstrap samples of the data.
- At each node the best splitter is chosen randomly.
- Leaf nodes predict the average value of the points reaching that node.
- This guarantees that each tree is slightly different.

Other Models: Random Forest

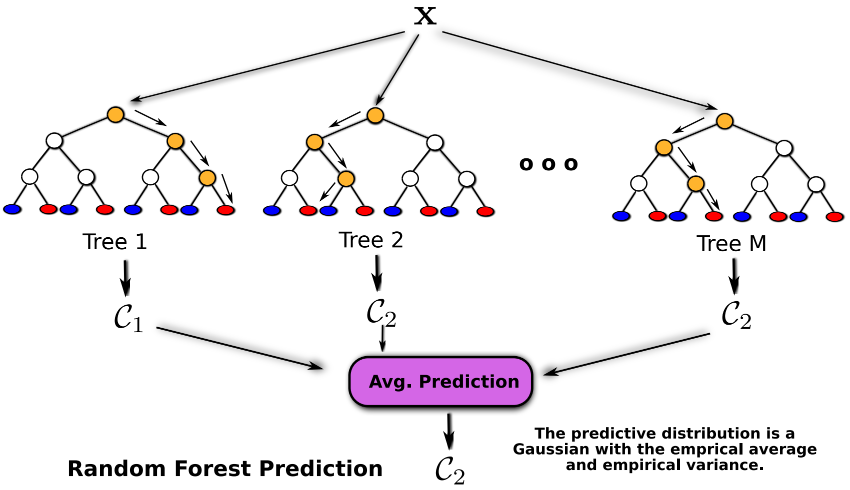
Ensemble method where the predictors are random regression trees trained on random subsamples of the data.



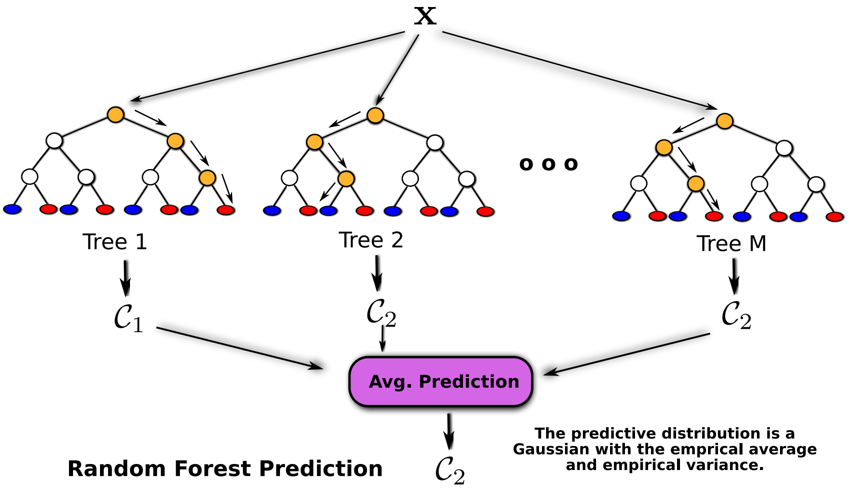
- Trees are grown on different bootstrap samples of the data.
- At each node the best splitter is chosen randomly.
- Leaf nodes predict the average value of the points reaching that node.
- This guarantees that each tree is slightly different.

Very cheap to compute and massively paralelizable!

Random Forest: Predictive Distribution



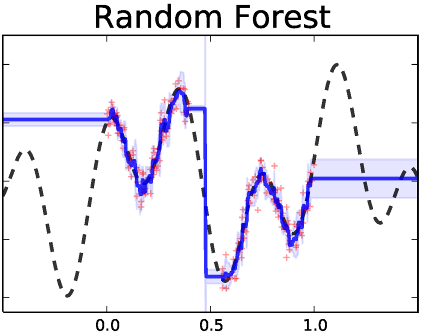
Random Forest: Predictive Distribution



$$p(f^\star|\mathcal{D}_n) = \mathcal{N}(f^\star|\overline{\mu}, \overline{\nu}^2)$$

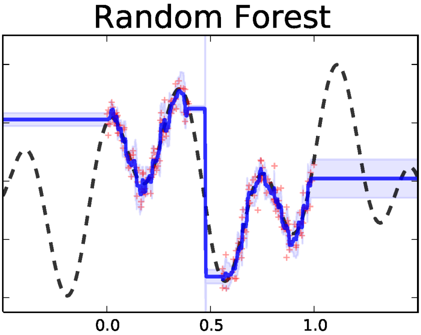
(Hutter et al., 2011)

Random Forest in Practice



(Shahriari et al., 2016)

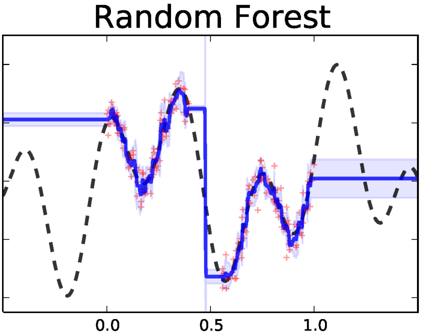
Random Forest in Practice



(Shahriari et al., 2016)

- Allows for a lot of evaluations (good when the objective is cheap).

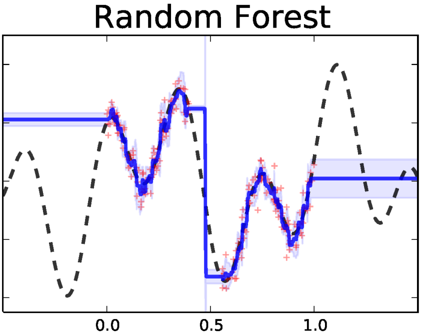
Random Forest in Practice



(Shahriari et al., 2016)

- Allows for a lot of evaluations (good when the objective is cheap).
- Too confident intervals in far away from the data regions.

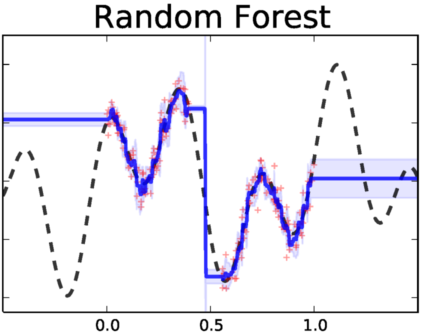
Random Forest in Practice



(Shahriari et al., 2016)

- Allows for a lot of evaluations (good when the objective is cheap).
- Too confident intervals in far away from the data regions.
- Conflictive predictions can cause the variance to be too high.

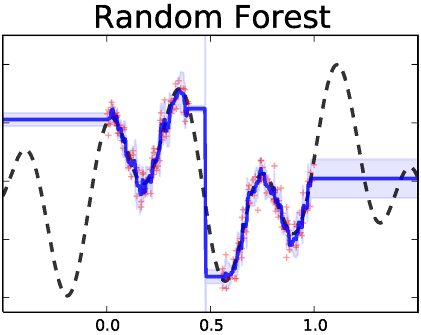
Random Forest in Practice



(Shahriari et al., 2016)

- Allows for a lot of evaluations (good when the objective is cheap).
- Too confident intervals in far away from the data regions.
- Conflictive predictions can cause the variance to be too high.
- Discontinuous: Difficult to optimize the acquisition function.

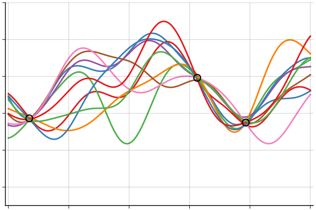
Random Forest in Practice

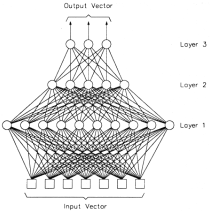


(Shahriari et al., 2016)

- Allows for a lot of evaluations (good when the objective is cheap).
- Too confident intervals in far away from the data regions.
- Conflictive predictions can cause the variance to be too high.
- Discontinuous: Difficult to optimize the acquisition function.
- No parameters to tune.

Other Models: Bayesian Neural Networks



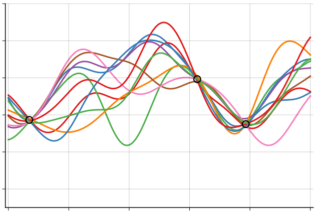


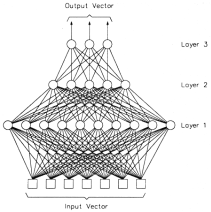
$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

- Neural networks scale well to the training data (linear cost).

Other Models: Bayesian Neural Networks



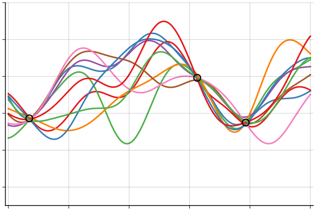


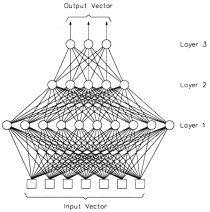
$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

- Neural networks scale well to the training data (linear cost).
- Trained very fast on GPUs.

Other Models: Bayesian Neural Networks



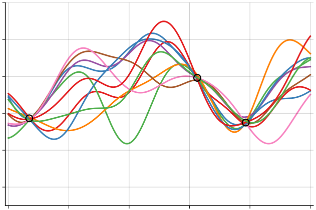


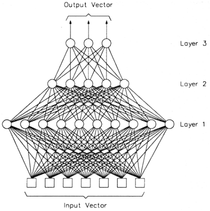
$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

- Neural networks scale well to the training data (linear cost).
- Trained very fast on GPUs.
- State of the art prediction results.

Other Models: Bayesian Neural Networks





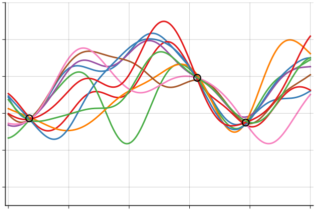
$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

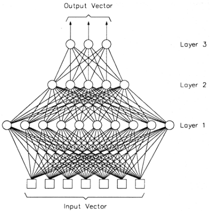
$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

- Neural networks scale well to the training data (linear cost).
- Trained very fast on GPUs.
- State of the art prediction results.

They are an alternative to GPs to allow for a large number observations!

Other Models: Bayesian Neural Networks





$$h_j(\mathbf{x}) = \tanh\left(\sum_{i=1}^{I} x_i w_{ji}\right)$$

$$f(\mathbf{x}) = \sum_{j=1}^{H} v_j h_j(\mathbf{x})$$

- Neural networks scale well to the training data (linear cost).
- Trained very fast on GPUs.
- State of the art prediction results.

They are an alternative to GPs to allow for a large number observations!

The posterior distribution of the networks weights W is intractable!

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.
- Variational Inference.

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.
- Variational Inference.
- Expectation Propagation.

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.
- Variational Inference.
- Expectation Propagation.
- Reinterpretations of dropout.

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.
- Variational Inference.
- Expectation Propagation.
- Reinterpretations of dropout.
- Point estimates and Bayesian linear-models in the last layer.

Bayesian Neural Networks: Predictive Distribution

Several techniques considered to approximate the predictive distribution:

- Markov Chain Monte Carlo methods.
- Variational Inference.
- Expectation Propagation.
- Reinterpretations of dropout.
- Point estimates and Bayesian linear-models in the last layer.

Trade-off between accuracy of the predictive distribution and scalability! Still a lot of research going on!

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
https://github.com/HIPS/Spearmint

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).
3. Allows for non-stationary functions via betawarp.

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).
3. Allows for non-stationary functions via betawarp.
4. Supports ignoring variables in some objectives or constraints.

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).
3. Allows for non-stationary functions via betawarp.
4. Supports ignoring variables in some objectives or constraints.
5. Supports different schedulers including SLURM.

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
`https://github.com/HIPS/Spearmint`

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).
3. Allows for non-stationary functions via betawarp.
4. Supports ignoring variables in some objectives or constraints.
5. Supports different schedulers including SLURM.
6. Stores all results (observations and suggestions) in a database.

Software for Bayesian Optimization

Many of the methods described are implemented into **Spearmint** using Python.
https://github.com/HIPS/Spearmint

Spearmint's super-nice features:

1. Easy problem configuration (JSON file) and wrapper calling.
2. GP hyper-parameter sampling via slice sampling (MCMC).
3. Allows for non-stationary functions via betawarp.
4. Supports ignoring variables in some objectives or constraints.
5. Supports different schedulers including SLURM.
6. Stores all results (observations and suggestions) in a database.

Other tools: SMAC (Java), Hyperopt (Python), Bayesopt (C++), PyBO (Python), MOE (Python / C++).

Further Extensions and Open Issues

1. **High-dimensionality**: BO is restricted to problems of moderate dimension. However, many big problems have low effective dimensionality which can be exploited (Wang *et al.*, 2013).

Further Extensions and Open Issues

1. **High-dimensionality**: BO is restricted to problems of moderate dimension. However, many big problems have low effective dimensionality which can be exploited (Wang *et al.*, 2013).

2. Consider **dependencies** among objectives / tasks (Swersky *et al.*, 2013) (Shah and Ghahramani, 2016). Most of the times the GPs are simply assumed to be independent, which is suboptimal.

Further Extensions and Open Issues

1. **High-dimensionality**: BO is restricted to problems of moderate dimension. However, many big problems have low effective dimensionality which can be exploited (Wang *et al.*, 2013).

2. Consider **dependencies** among objectives / tasks (Swersky *et al.*, 2013) (Shah and Ghahramani, 2016). Most of the times the GPs are simply assumed to be independent, which is suboptimal.

3. Most acquisition functions consider an evaluation **horizon equal to one**. We can do better by considering a particular evaluation budget and taking decisions accordingly (González *et al.*, 2016).

Further Extensions and Open Issues

1. **High-dimensionality**: BO is restricted to problems of moderate dimension. However, many big problems have low effective dimensionality which can be exploited (Wang *et al.*, 2013).

2. Consider **dependencies** among objectives / tasks (Swersky *et al.*, 2013) (Shah and Ghahramani, 2016). Most of the times the GPs are simply assumed to be independent, which is suboptimal.

3. Most acquisition functions consider an evaluation **horizon equal to one**. We can do better by considering a particular evaluation budget and taking decisions accordingly (González *et al.*, 2016).

4. **Safe Bayesian Optimization**: Sometimes we should avoid evaluating the objective at particular input locations (system failure) where it falls below some critical value (Berkenkamp *et al.*, 2016).

Conclusions

Bayesian optimization:

Conclusions

Bayesian optimization:

1. Is a **powerful tool** that can be used to optimize black-box objectives that are very expensive to evaluate and noisy.

Conclusions

Bayesian optimization:

1. Is a **powerful tool** that can be used to optimize black-box objectives that are very expensive to evaluate and noisy.
2. Provides **significantly better results** than a random / uniform exploration of the input space.

Conclusions

Bayesian optimization:

1. Is a **powerful tool** that can be used to optimize black-box objectives that are very expensive to evaluate and noisy.
2. Provides **significantly better results** than a random / uniform exploration of the input space.
3. Can deal with complicated optimization problems with **several objectives and / or multiple constraints**.

Conclusions

Bayesian optimization:

1. Is a **powerful tool** that can be used to optimize black-box objectives that are very expensive to evaluate and noisy.
2. Provides **significantly better results** than a random / uniform exploration of the input space.
3. Can deal with complicated optimization problems with **several objectives and / or multiple constraints**.

Thank you very much!

References

- M. Bauer, M. van der Wilk & C. E. Rasmussen. Understanding probabilistic sparse Gaussian process approximations. In Advances in neural information processing systems (pp. 1533-1541), 2016.
- F. Berkenkamp, A. Krause, A. P. Schoellig. Bayesian Optimization with Safety Constraints: Safe and Automatic Parameter Tuning in Robotics, arXiv:1602.04450, 2016.
- E. Brochu, V. M. Cora and N. de Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. eprint arXiv:1012.2599, 2010.
- E. C. Garrido-Merchán, D. Hernández-Lobato. Predictive Entropy Search for Multi-objective Bayesian Optimization with Constraints. Neurocomputing, 2019. In press.
- J. González, M. Osborne, N. Lawrence. GLASSES: Relieving The Myopia Of Bayesian Optimisation. International Conference on Artificial Intelligence and Statistics, 2016.
- P. Henning, C. J. Schuler. Entropy Search for Information-Efficient Global Optimization. Journal of Machine Learning Research, 13, 1809-1837, 2012.
- D. Hernández-Lobato, J.M. Hernández-Lobato, A. Shah, R. P. Adams. Predictive Entropy Search for Multi-objective Optimization. International Conference on Machine Learning, 2016.
- J. M. Hernández Lobato, M. A. Gelbart, R. P. Adams, M. W. Hoffman, Z. Ghahramani. A General Framework for Constrained Bayesian Optimization using Information-based Search. Journal of Machine Learning Research, 17, 153, 2016.
- J. M. Hernández-Lobato, M. A. Gelbart, M. W. Hoffman, R. P. Adams, Z. Ghahramani. Predictive Entropy Search for Bayesian Optimization with Unknown Constraints, International Conference on Machine Learning, 2015.
- J. M. Hernández-Lobato, M. W. Hoffman, Z. Ghahramani. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. Neural Information Processing Systems, 2014.
- D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. Journal of Global Optimization, 13, 455492, 1998.
- J. H. Kushner. A new method of locating the maximum of an arbitrary multipeak curve in the presence of noise. Journal of Basic Engineering, 86, 97106, 1968.
- R. Neal. Slice Sampling. Annals of Statistics, 31, 705-767, 2003.

References

- A. Shah, Z. Ghahramani. Pareto Frontier Learning with Expensive Correlated Objectives. International Conference on Machine Learning, 2016.
- A. Shah, Z. Ghahramani. Parallel Predictive Entropy Search for Batch Global Optimization of Expensive Functions. Neural Information Processing Systems, 2015.
- Minka, T. Expectation Propagation for approximate Bayesian inference. Uncertainty in Artificial Intelligence, 2001.
- J. Močkus. On Bayesian methods for seeking the extremum. IFIP Technical Conference on Optimization Techniques, 1978.
- C. E. Rasmussen, C. K. I. Williams. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press, 2006.
- J. Snoek, H. Larochelle, R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. Advances in Neural Information Processing Systems, 2012.
- J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, ..., R. Adams, Scalable bayesian optimization using deep neural networks. In International conference on machine learning, 2171-2180, 2015.
- K. Swersky, J. Snoek, R. P. Adams. Multi-task Bayesian optimization. Advances in neural information Processing Systems, 2013.
- K. Swersky, J. Snoek, & R. P. Adams. Freeze-thaw Bayesian optimization. arXiv preprint arXiv:1406.3896, 2014.
- B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. de Freitas Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE, 104, 148–175, 2016.
- J. T. Springenberg, A. Klein, S. Falkner, F. Hutter. Bayesian optimization with robust Bayesian neural networks. Advances in Neural Information Processing Systems, 4134-4142, 2016.
- N. Srinivas, A. Krause, S. Kakade, M. Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design International Conference on Machine Learning, 2010.
- F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In LION, pages 507-523, 2011.
- Z. Wang M. Zoghi F. Hutter D. Matheson, N. de Freitas Bayesian Optimization in High Dimensions via Random Embeddings. International Joint Conferences on Artificial Intelligence, 2013.