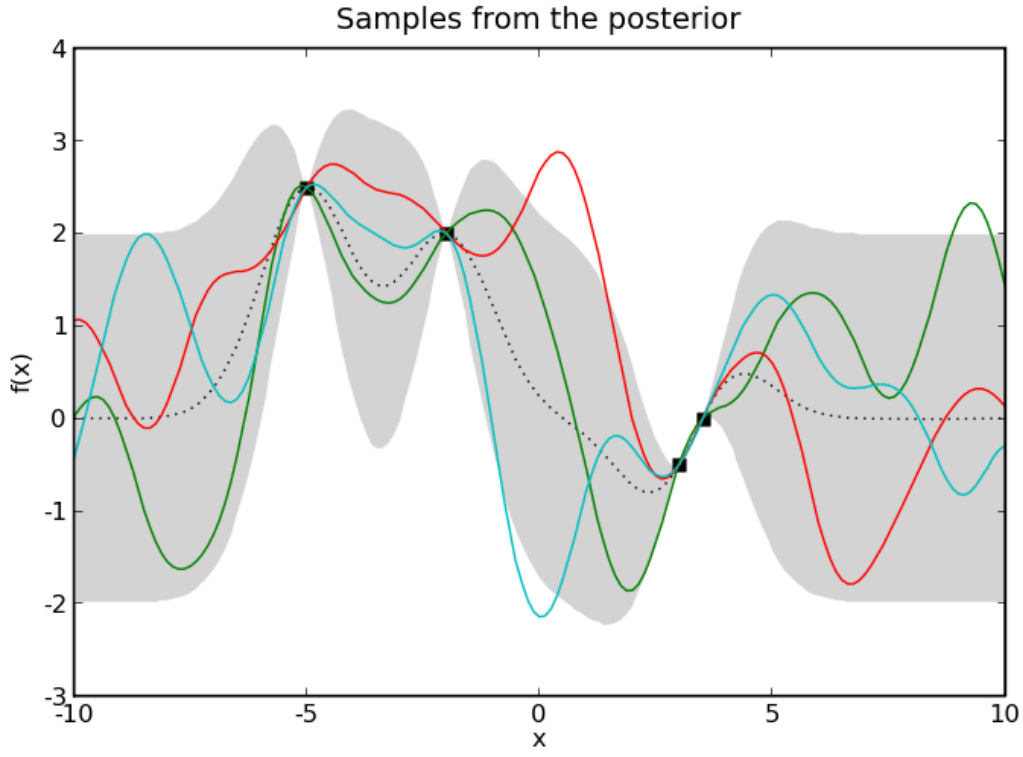


Gaussian processes and bayesian optimization

Stanisław Jastrzębski

kudkudak.github.io kudkudak

Plan

Goal: talk about modern hyperparameter optimization algorithms

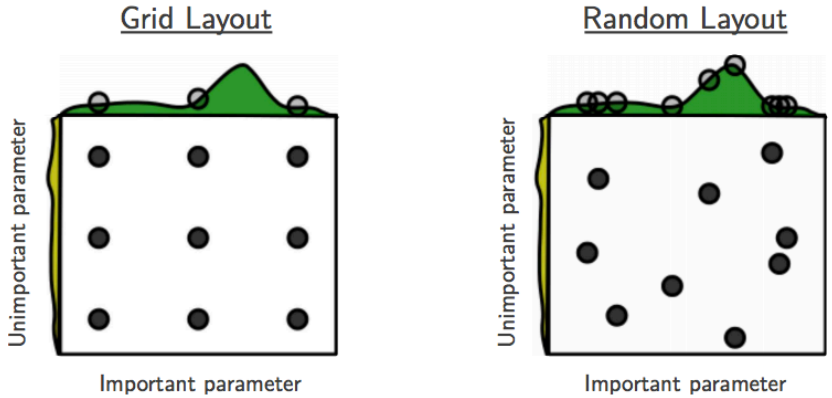
- Bayes reminder: equivalent linear regression formulations
- Gaussian Processes
- Hyperparameter optimization via Gaussian Processes (+code and simple experiment)

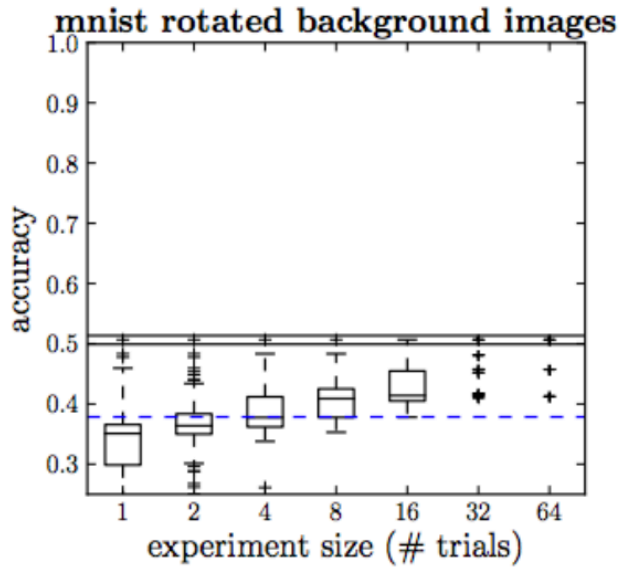
What do we want to achieve

- Select best hyperparameters for our model
- We will model directly $f(x)$, where x are our hyperparameters
- We will ask question like "how probably it is that accuracy is $> 60\%$ for C=100"
- We will need accuracy and variance estimate for each point

Why care? We have grid search

[Bergsta & Bengio, 2012] show that random search is usually better!





Linear *Regression* :)

Linear regression [Bishop, ch.3]

- Assume model $y(x) = \boldsymbol{\theta}^T x$
- Optimal θ under MSE error $J(\theta) = \sum_i (y_n - \theta^T x_n)^2$ is

Definition (Least Squares estimate)

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

Ridge regression [Bishop, ch.3]

Definition (Ridge regression cost)

$J(\boldsymbol{\theta}) = \sum_i (y_n - \boldsymbol{\theta}^T x_n)^2 + \lambda \|w\|^2$

Definition (Ridge regression estimate)

$\hat{\boldsymbol{\theta}} = (\lambda \mathbf{I} + \mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$

Kernels [Bishop, ch.3]

Just replace $\mathbf{x}$ with $\phi_i(\mathbf{x})$, each ϕ_i is a new feature. Examples:

- $\phi_i(\mathbf{x}) = x_i$
- $\phi_i(\mathbf{x}) = \exp\{\frac{-(x-\mu_j)^2}{2s^2}\}$
- $\phi_i(\mathbf{x}) = x_i^2$

Kernels [Bishop, ch.3]

Just replace $\mathbf{x}$ with $\phi_i(\mathbf{x})$, each ϕ_i is a new feature. Examples:

- $\phi_i(\mathbf{x}) = x_i$
- $\phi_i(\mathbf{x}) = \exp\{\frac{-(x-\mu_j)^2}{2s^2}\}$
- $\phi_i(\mathbf{x}) = x_i^2$

$$y(x) = \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_3 \phi_3(x)$$

Kernels [Bishop, ch.3]

Just replace $\mathbf{x}$ with $\phi_i(\mathbf{x})$, each ϕ_i is a new feature. Examples:

- $\phi_i(\mathbf{x}) = x_i$
- $\phi_i(\mathbf{x}) = \exp\{\frac{-(x-\mu_j)^2}{2s^2}\}$
- $\phi_i(\mathbf{x}) = x_i^2$

$$y(x) = \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_3 \phi_3(x)$$
$$y(x) = 1.2 \exp\{-(x-1)^2\} + 2.3 \exp\{-(x-2)^2\} + 0.3 \exp\{-(x-3)^2\}$$

Kernels [Bishop, ch.3]

$\mathbf{\Phi}_{ij} = \phi_i(x_j)$

Definition (Kernelized ridge regression estimate)

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

Probabilistic formulation [Bishop, ch.3]

- Standard probabilistic model $y = \theta^T x + \nu$, where $\nu \sim \mathcal{N}(0, \sigma^2)$
- Wow! It follows $y - \boldsymbol{\theta}^T x \sim \mathcal{N}(0, \sigma^2)$, or equivalently
  $P(y|x, \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y-\boldsymbol{\theta}^T x)^2)}{2\sigma^2})$

Probabilistic formulation [Bishop, ch.3]

Everything will be a Gauss from now on :)

Definition (Least-squares likelihood function)

$p(y|\mathbf{x}, \boldsymbol{\theta}, \sigma) = \mathcal{N}(\boldsymbol{\theta}^T\mathbf{x}, \sigma^2)$

[Derive on whiteboard MLE estimate]

Probabilistic formulation [Bishop, ch.3]

Everything will be a Gauss from now on :)

Definition (Least-squares likelihood function)

$p(y|\mathbf{x}, \boldsymbol{\theta}, \sigma) = \mathcal{N}(\boldsymbol{\theta}^T\mathbf{x}, \sigma^2)$

[Derive on whiteboard MLE estimate]

Definition (MLE estimate)

$\hat{\boldsymbol{\theta}} = argmax_\theta\, p(\mathbf{t}|\mathbf{X}, \boldsymbol{\theta}, \sigma) = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\mathbf{y}$

Bayesian approach

Bayesian approach is all about priors. Every analysis makes prior assumption, but here we state them in an explicit manner.

Bayesian approach

Bayesian approach is all about priors. Every analysis makes prior assumption, but here we state them in an explicit manner. In general we have 3 things in a bayesian model (everything can be formulated as a bayes model, even SVM):

- likelihood $p(\mathbf{y}|x, \boldsymbol{\theta})$
- prior $p(\boldsymbol{\theta})$
- postertior $p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y})$

Bayesian linear regression

Goal: formulate linear regression with explicit prior on weights

- likelihood $p(y|x, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^T \phi(x), \beta^{-1})$

Bayesian linear regression

Goal: formulate linear regression with explicit prior on weights

- likelihood $p(y|x, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^T \phi(x), \beta^{-1})$
- prior $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\theta_0, \boldsymbol{\Sigma}_0)$

Bayesian linear regression

Goal: formulate linear regression with explicit prior on weights

- likelihood $p(y|x, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^T \phi(x), \beta^{-1})$
- prior $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\theta_0, \boldsymbol{\Sigma}_0)$
- posterior $p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})} = ?$

Gaussian ju–jitsu

Gaussian are “closed“ with respect to many operations

Theorem (Marginal and Conditional Gaussian)

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$$
$$p(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{x}, \boldsymbol{L}^{-1})$$

Then

$$p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{\mu}, \boldsymbol{L}^{-1} + \boldsymbol{A}\boldsymbol{\Lambda}^{-1}\boldsymbol{A}^T)$$
$$p(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\Sigma}\{\boldsymbol{A}^T\boldsymbol{L}\boldsymbol{y} + \boldsymbol{\Lambda}\boldsymbol{\mu}\}, \boldsymbol{\Sigma})$$

, where $\boldsymbol{\Sigma} = (\Lambda + \boldsymbol{A}^T\boldsymbol{A})^{-1}$

Bayesian linear regression

Goal: formulate linear regression with explicit prior on weights

- likelihood $p(y|x, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^T \phi(x), \beta^{-1})$
- prior $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\theta_0, \boldsymbol{\Sigma}_0)$
- posterior $p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})} = \mathcal{N}(\boldsymbol{\theta}|\theta_N, \boldsymbol{\Sigma}_N)$

, where

$$\boldsymbol{\theta}_N = \boldsymbol{\Sigma}_N(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\theta}_0 + \beta\boldsymbol{\Phi}^T\mathbf{y})$$
$$\boldsymbol{\Sigma}_N^{-1} = \boldsymbol{\Sigma}_0^{-1} + \beta\boldsymbol{\Phi}^T\boldsymbol{\Phi}$$

Let's assume $\boldsymbol{\Sigma}_0 = \alpha^{-1}I$. What happens when $\alpha \to 0$? . Infinitely broad prior gives 0 regularization.

Bayesian linear regression

If $\alpha \neq 0$:

$$\ln p(\boldsymbol{\theta}|\mathbf{y}) = -\frac{\beta}{2}\sum_{n=1}^{N}\{y_n - \boldsymbol{\theta}^T\phi(x_n)\} - \frac{\alpha}{2}\boldsymbol{\theta}^T\boldsymbol{\theta} + const \qquad (1)$$

Bayesian linear regression

If $\alpha \neq 0$:

$$\ln p(\boldsymbol{\theta}|\mathbf{y}) = -\frac{\beta}{2}\sum_{n=1}^{N}\{y_n - \boldsymbol{\theta}^T\phi(x_n)\} - \frac{\alpha}{2}\boldsymbol{\theta}^T\boldsymbol{\theta} + const \tag{1}$$

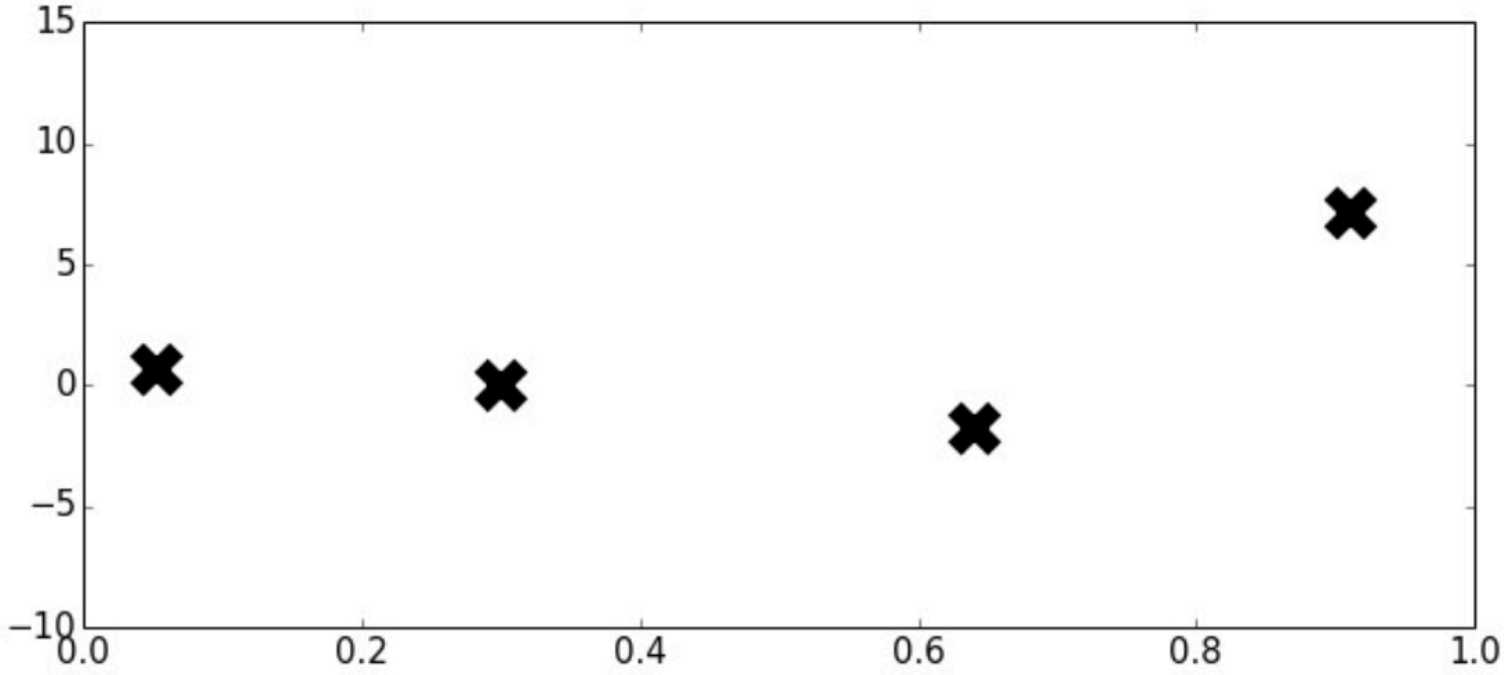
$lnp(\boldsymbol{\theta}|\mathbf{y})$ (MAP - maximum a posteriori estimate) *is same as ridge regression* objective. Magic?

Bayes a'int magic

Putting up your Bayes Hat is just about making your assumptions clear. In the end it is just different road to the same goal (objective).

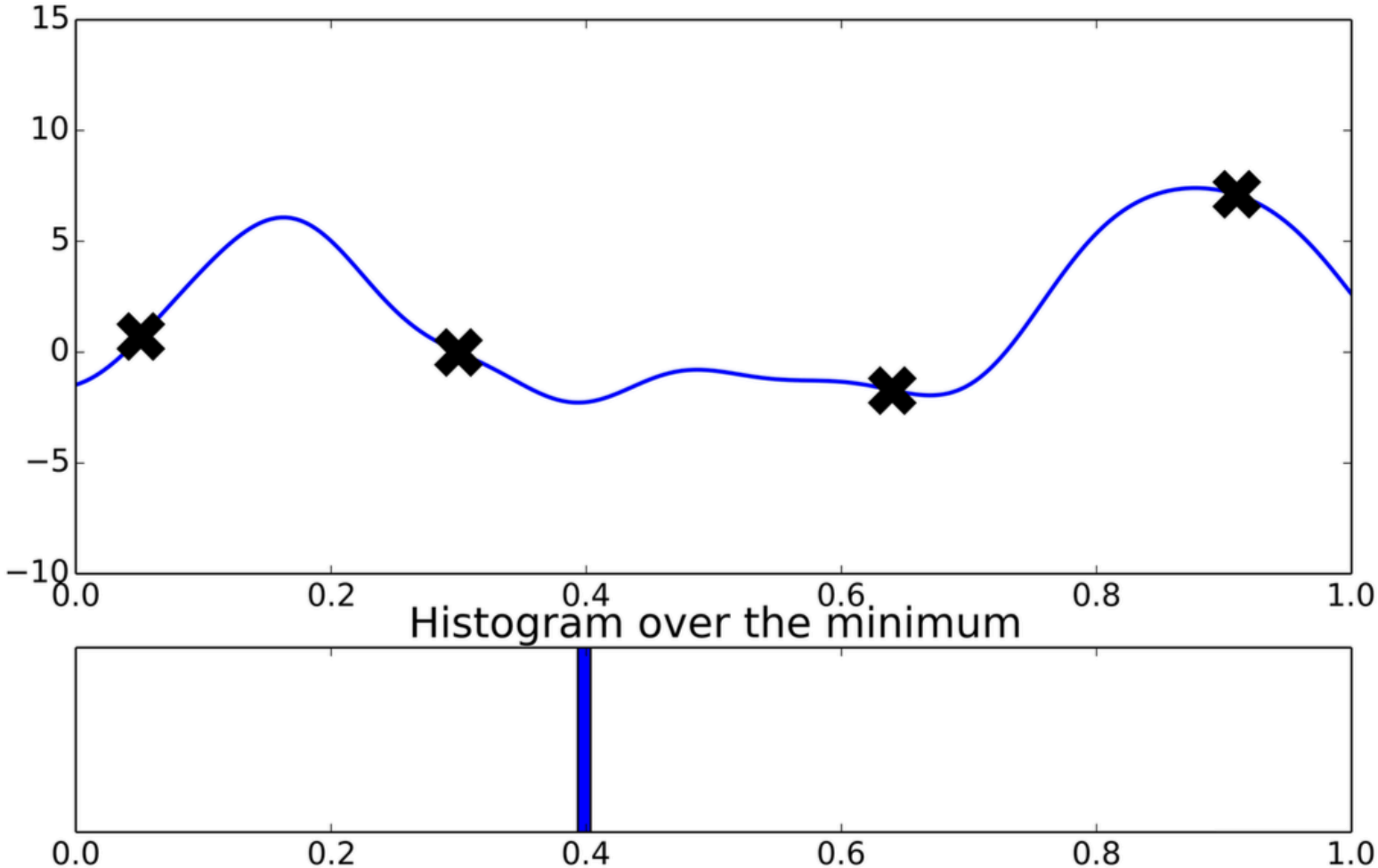
Gaussian *Processes*

Intuition - we want to build this somehow



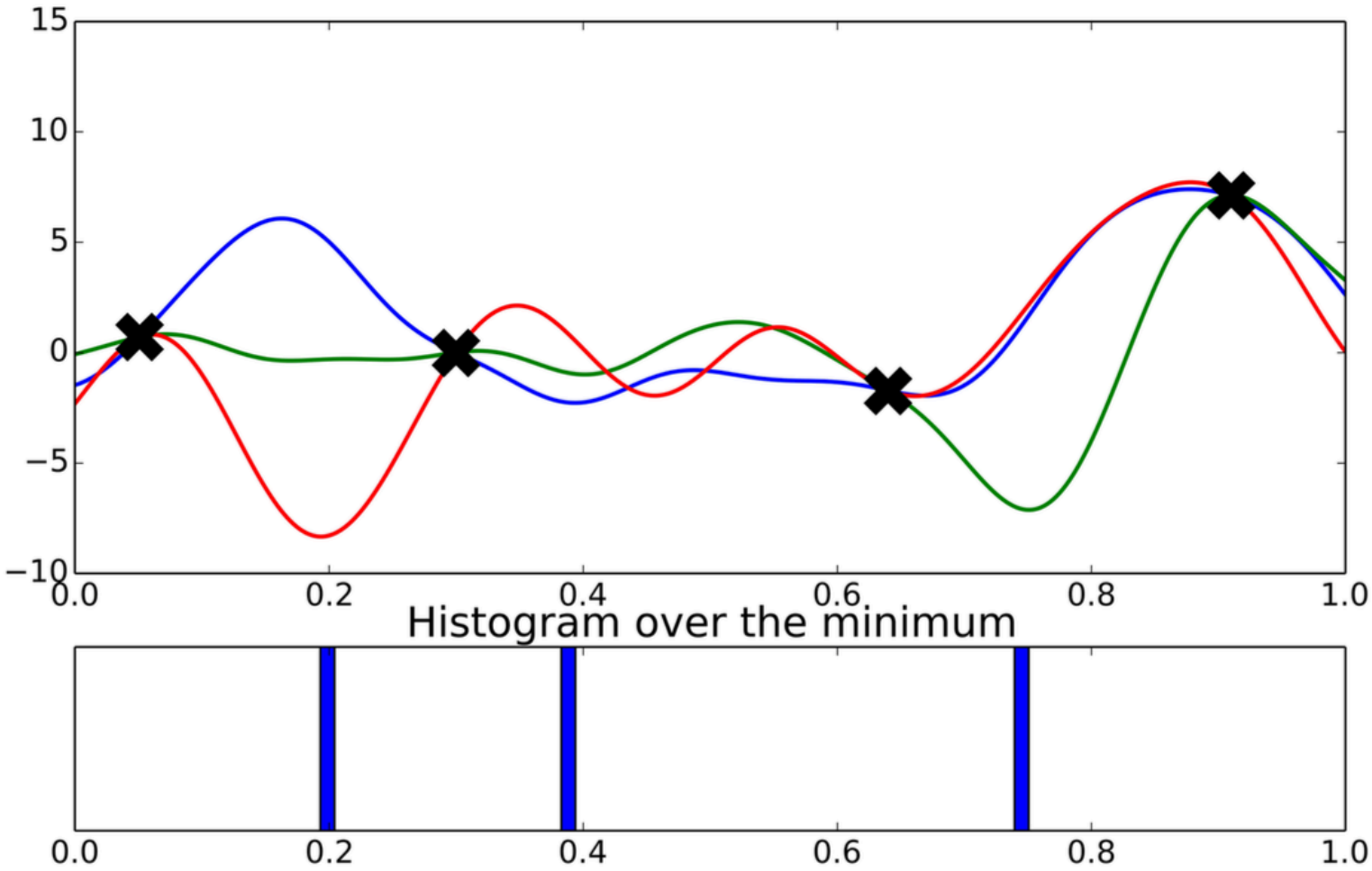
Where is the minimum of f?
Where should the take the next evaluation?

Intuition - we want to build this somehow



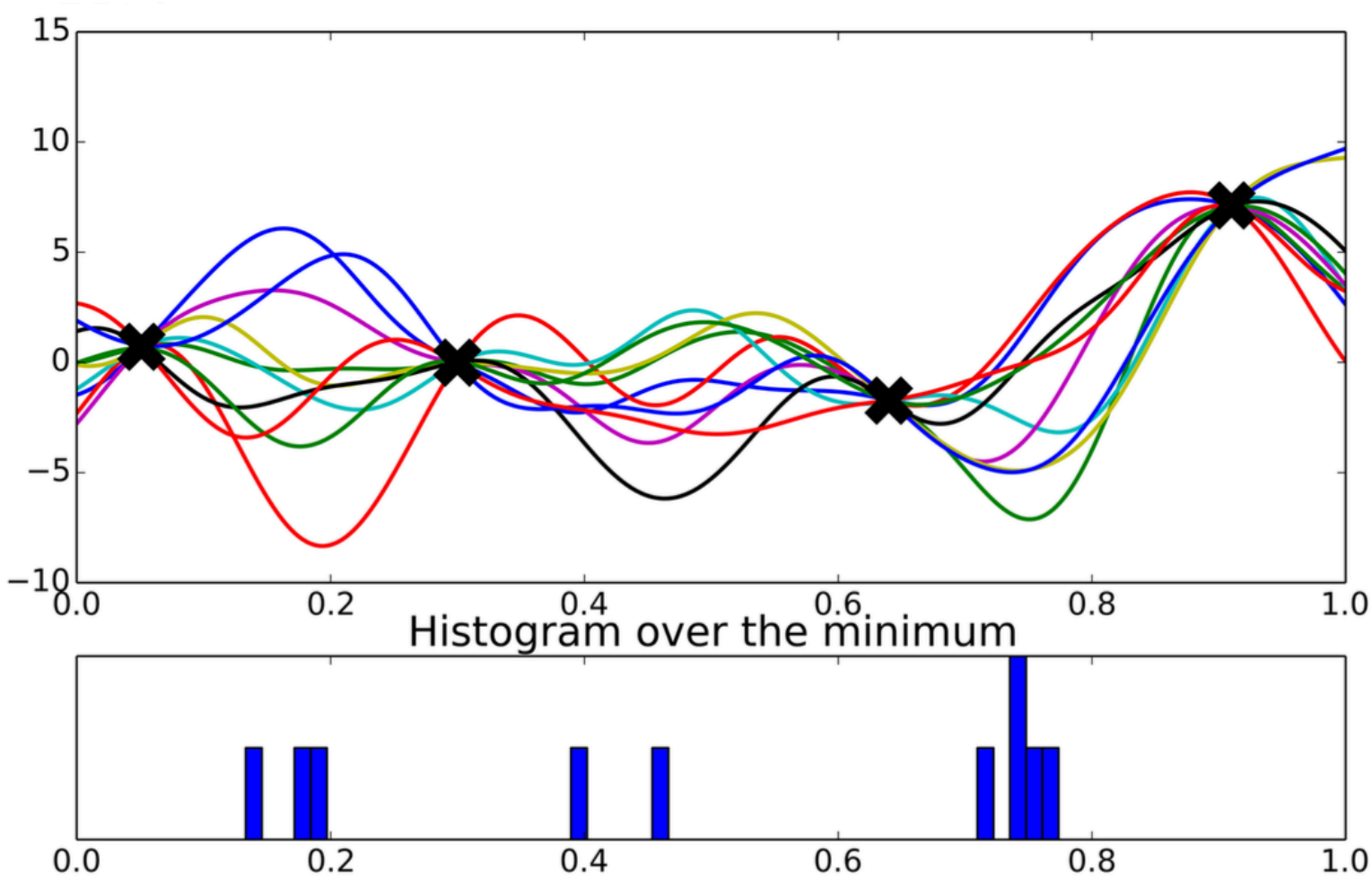
We want to build this

Intuition - we want to build this somehow



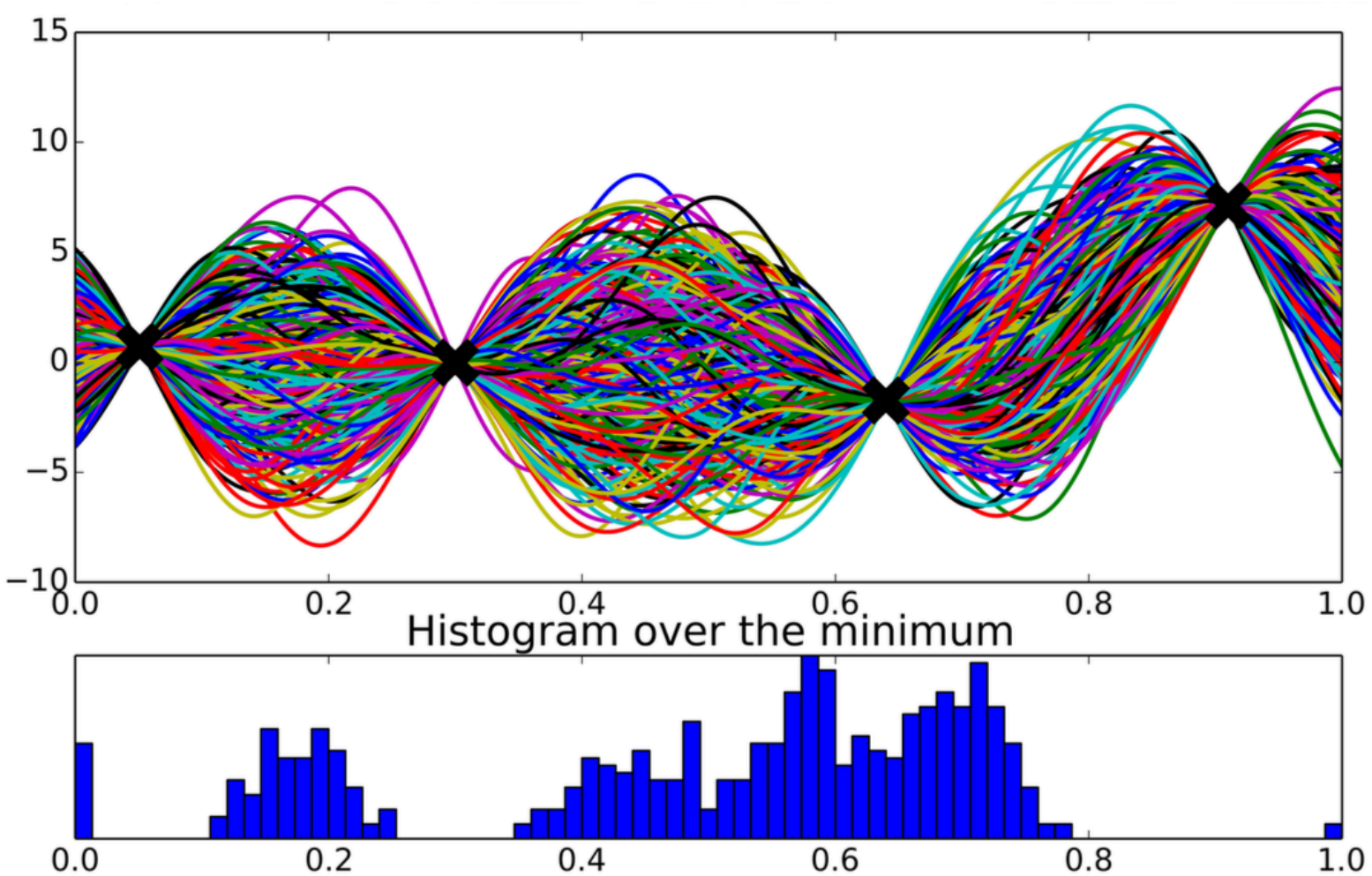
We want to build this

Intuition - we want to build this somehow



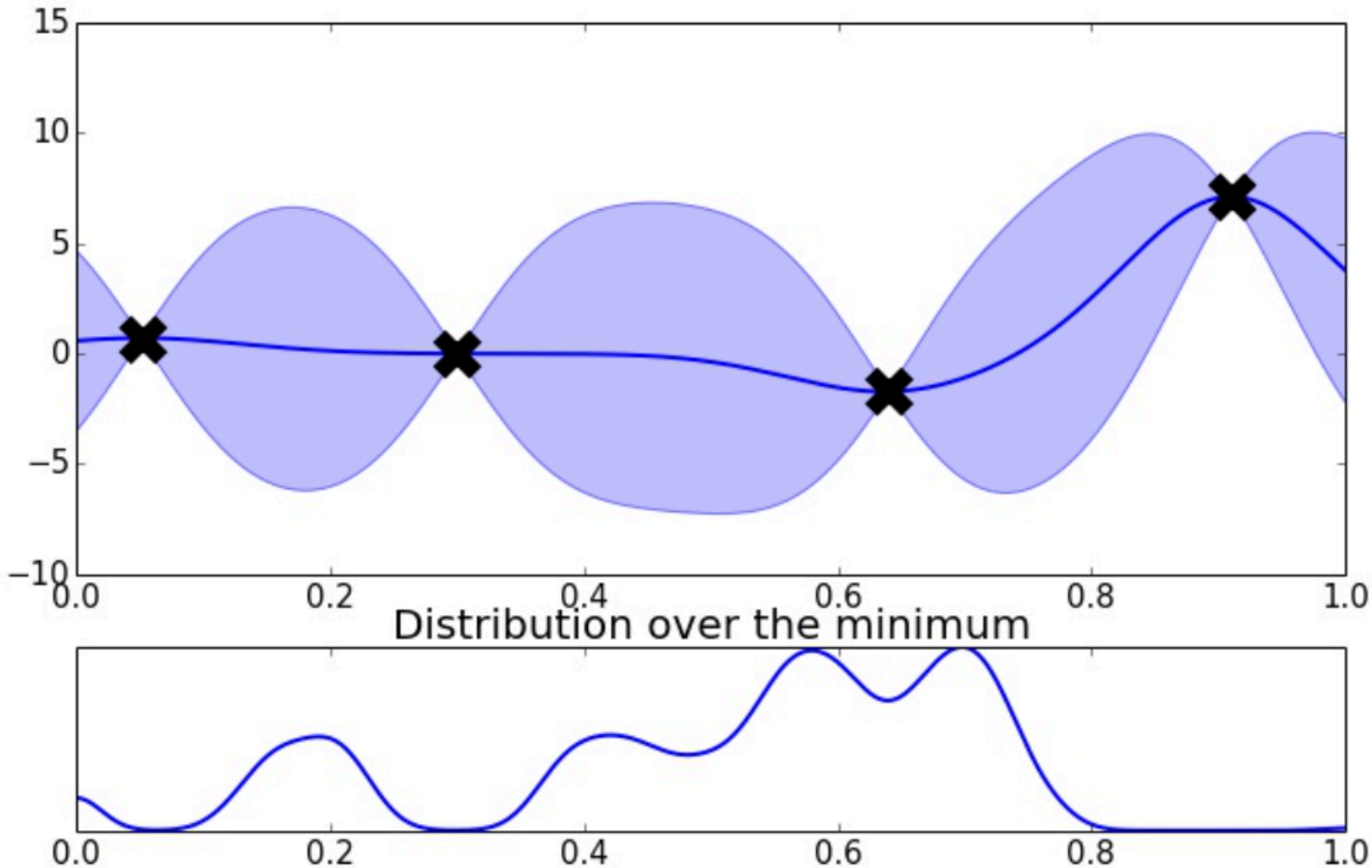
We want to build this

Intuition - we want to build this somehow



We want to build this

Intuition - we want to build this somehow



We want to build this

Gauss family provides inference $\Leftrightarrow$ linear algebra bridge

$$p(\mathbf{x}; \boldsymbol{\mu}\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(x-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(x-\boldsymbol{\mu}))$$

$$x = \begin{bmatrix} x_A \\ x_B \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}.$$

1. **Normalization.** The density function normalizes, i.e.,

$$\int_x p(x; \mu, \Sigma)dx = 1.$$

2. **Marginalization.** The marginal densities,

$$p(x_A) = \int_{x_B} p(x_A, x_B; \mu, \Sigma)dx_B$$

$$p(x_B) = \int_{x_A} p(x_A, x_B; \mu, \Sigma)dx_A$$

$$x_A \sim \mathcal{N}(\mu_A, \Sigma_{AA})$$

$$x_B \sim \mathcal{N}(\mu_B, \Sigma_{BB}).$$

Gauss family provides inference $\Leftrightarrow$ linear algebra bridge

3. **Conditioning.** The conditional densities

$$p(x_A \mid x_B) = \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_A} p(x_A, x_B; \mu, \Sigma) dx_A}$$
$$p(x_B \mid x_A) = \frac{p(x_A, x_B; \mu, \Sigma)}{\int_{x_B} p(x_A, x_B; \mu, \Sigma) dx_B}$$

are also Gaussian:

$$x_A \mid x_B \sim \mathcal{N}\big(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}\big)$$
$$x_B \mid x_A \sim \mathcal{N}\big(\mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(x_A - \mu_A), \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}\big).$$

Gaussian process - formally

$f(x_0) = 2.5,\ f(x_1) = 3 \ldots f(x_m) = 4$ Let $\mathcal{H}$ be a set of all functions over m points (it is finite). Let's exploit correspondence between function $f(\cdot) \in \mathcal{H}$

$$p(\vec{f}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(f(x_i) - \mu_i)^2) \qquad \text{(The only important eq.)}$$

Gaussian process - formally

$f(x_0) = 2.5,\ f(x_1) = 3 \ldots f(x_m) = 4$ Let $\mathcal{H}$ be a set of all functions over m points (it is finite). Let's exploit correspondence between function $f(\cdot) \in \mathcal{H}$

$$p(\vec{f}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(f(x_i) - \mu_i)^2) \qquad \text{(The only important eq.)}$$

Gaussian Processes - more formally

A **Gaussian process** is a stochastic process such that any finite subcollection of R.V has a multivariate Gaussian distribution

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \sim \mathcal{N}\Big(\begin{bmatrix} f(m(x_1)) \\ \vdots \\ f(m(x_m)) \end{bmatrix} \begin{bmatrix} k(x_1,x_1) & \dots & k(x_1,x_m) \\ \vdots & \ddots & \dots \\ k(x_m,x_1) & \dots & k(x_m,x_m) \end{bmatrix} \Big) \tag{2}$$

Theorem (Mean and covariance of point)

$$m(x) = E[x]$$
$$k(x,x') = E[(x - m(x))(x' - m(x'))]$$

Kernel matrix

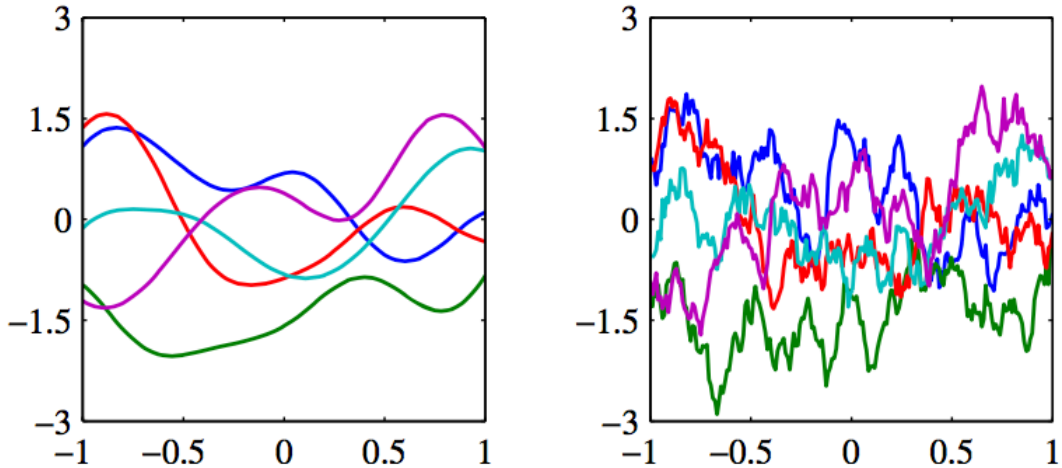
Kernels are *very* powerful, a kernel function could do complicated parsing during sentence comparison, or perform graph similarity when comparing compounds, and all of this is compatible with GP.

Techniques for Constructing New Kernels.

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels will also be valid:

$$k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}') \quad (6.13)$$
$$k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') \quad (6.14)$$
$$k(\mathbf{x}, \mathbf{x}') = q\left(k_1(\mathbf{x}, \mathbf{x}')\right) \quad (6.15)$$
$$k(\mathbf{x}, \mathbf{x}') = \exp\left(k_1(\mathbf{x}, \mathbf{x}')\right) \quad (6.16)$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \quad (6.17)$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') \quad (6.18)$$
$$k(\mathbf{x}, \mathbf{x}') = k_3\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')\right) \quad (6.19)$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}' \quad (6.20)$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a) + k_b(\mathbf{x}_b, \mathbf{x}'_b) \quad (6.21)$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a)k_b(\mathbf{x}_b, \mathbf{x}'_b) \quad (6.22)$$

Kernel matrix - intuition



[Code presentation]

Gaussian Process regression

Definition (GP regression)

$y^{(i)} = f(x^{(i)}) + \epsilon^{(i)}$, where $f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$

Same as before, just not $\boldsymbol{\theta}^T x$ but $f(\mathbf{x})$

Inference in GP - just use Gaussian ju–jitsu

We have trained on $\mathbf{X}, \mathbf{y}$ and want to make predictions on $\mathbf{X}_*, \mathbf{y}_*$ (to calculate new hyperparameters, for instance)

$$\begin{bmatrix} f \\ \vec{f}^* \end{bmatrix} | X, X_* \sim \mathcal{N}\Big(\vec{0}, \begin{bmatrix} k(X,X) & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\Big) \tag{3}$$

Add noise and we find distribution of y^*

Inference in GP - just use Gaussian ju–jitsu

We have trained on $\mathbf{X}, \mathbf{y}$ and want to make predictions on $\mathbf{X}_*, \mathbf{y}_*$ (to calculate new hyperparameters, for instance)

$$\begin{bmatrix} f \\ \vec{f}^* \end{bmatrix} | X, X_* \sim \mathcal{N}\Big(\vec{0}, \begin{bmatrix} k(X,X) & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) \end{bmatrix}\Big) \tag{3}$$

Add noise and we find distribution of y^*

$$\begin{bmatrix} y \\ y^* \end{bmatrix} | X, X_* \sim \begin{bmatrix} \vec{f} \\ \vec{f}^* \end{bmatrix} + \begin{bmatrix} \epsilon \\ \epsilon^* \end{bmatrix} \sim \mathcal{N}\Big(\vec{0}, \begin{bmatrix} k(X,X) + \sigma^2 I & k(X,X_*) \\ k(X_*,X) & k(X_*,X_*) + \sigma^2 I \end{bmatrix}\Big) \tag{4}$$

Inference in GP - just use Gaussian ju–jitsu #2

Now we want:

$$\mathbf{y}_*|\mathbf{y} = ? \tag{5}$$

Inference in GP - just use Gaussian ju–jitsu #2

Now we want:

$$\mathbf{y}_*|\mathbf{y} = ? \tag{5}$$

If we substitute into previous equations we will get:

$$\mathbf{y}_*|\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \tag{6}$$

, where

$$\boldsymbol{\mu}_* = K(\mathbf{X}_*, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1}\mathbf{y}$$
$$\boldsymbol{\Sigma}_* = K(\mathbf{X}_*, \mathbf{X}_*) + \sigma^2 I - K(\mathbf{X}_*, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1}K(\mathbf{X}, \mathbf{X}_*)$$

Inference in GP - just use Gaussian ju–jitsu #2

Now we want:

$$\mathbf{y}_*|\mathbf{y} = ? \tag{5}$$

If we substitute into previous equations we will get:

$$\mathbf{y}_*|\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*) \tag{6}$$

, where

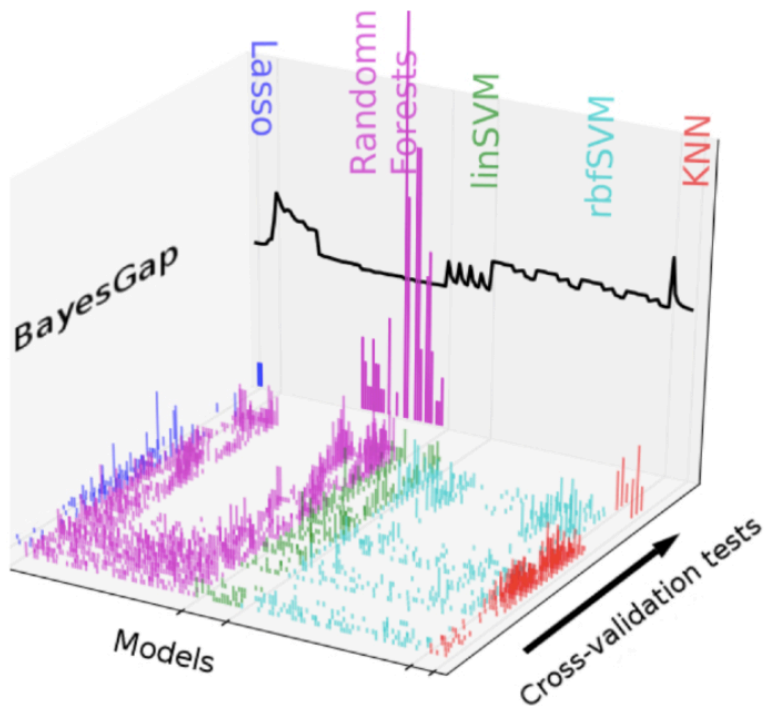
$$\boldsymbol{\mu}_* = K(\mathbf{X}_*, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1}\mathbf{y}$$
$$\boldsymbol{\Sigma}_* = K(\mathbf{X}_*, \mathbf{X}_*) + \sigma^2 I - K(\mathbf{X}_*, \mathbf{X})(K(\mathbf{X}, \mathbf{X}) + \sigma^2 I)^{-1}K(\mathbf{X}, \mathbf{X}_*)$$

Note: in practice we fit $\boldsymbol{\theta}_{MLE} = argmaxp(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$ (for instance kernel width)
[Show me some code]

Bayesian *Optimization*

Taking human out of the loop. Applications

- Picking parameters in different solvers (CPLEX has 72 parameters)
- Optimizing hyperparameters in ML algoirhtms
- A/B testing of website
- ...



Bayesian optimization algorithm

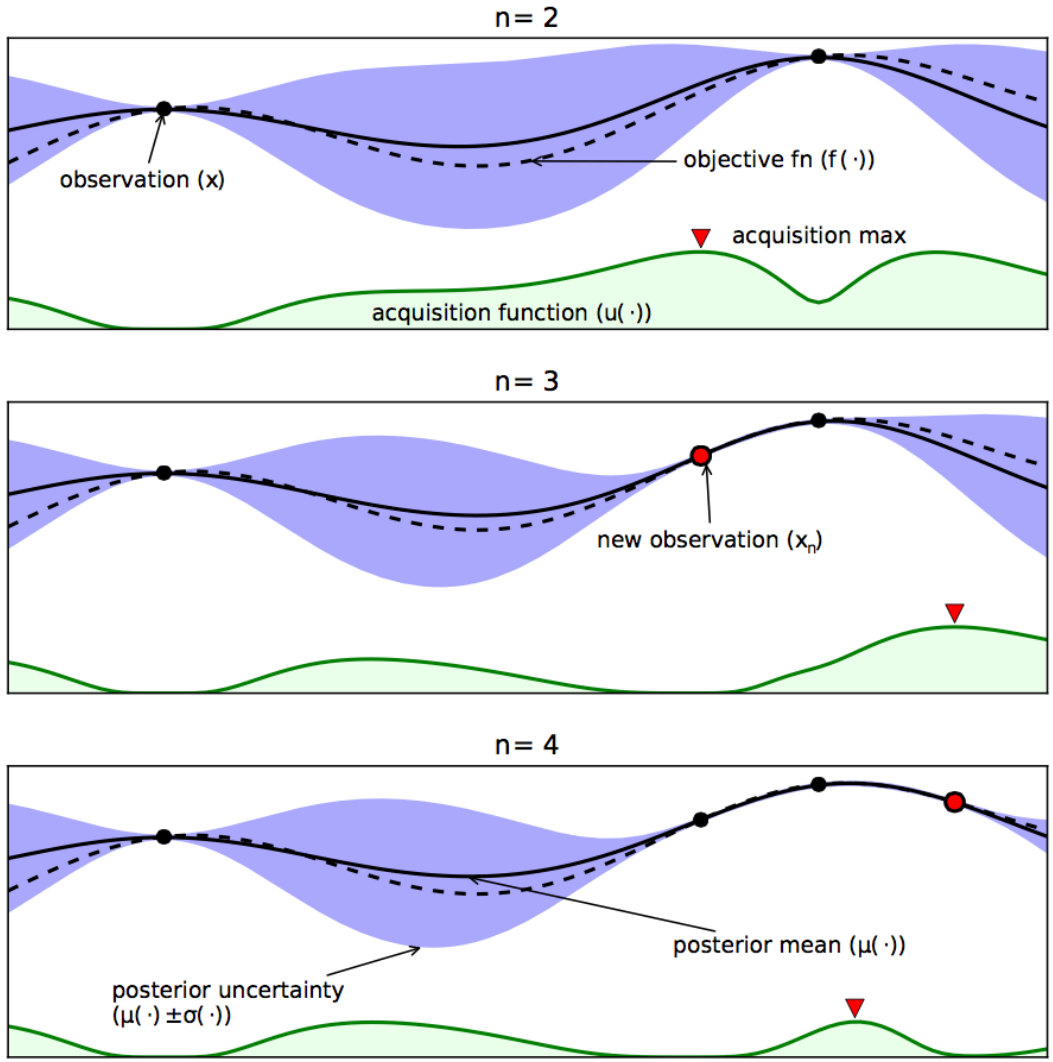
Algorithm 1 Bayesian optimization

1: **for** $n = 1, 2, \ldots$ **do**
2: select new $\mathbf{x}_{n+1}$ by optimizing acquisition function α

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$

3: query objective function to obtain y_{n+1}
4: augment data $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\mathbf{x}_{n+1}, y_{n+1})\}$
5: update statistical model
6: **end for**

Bayesian optimization algorithm



Acquisition functions

Having full probabilistic model you are only limited by your creativity as to how to pick next point.

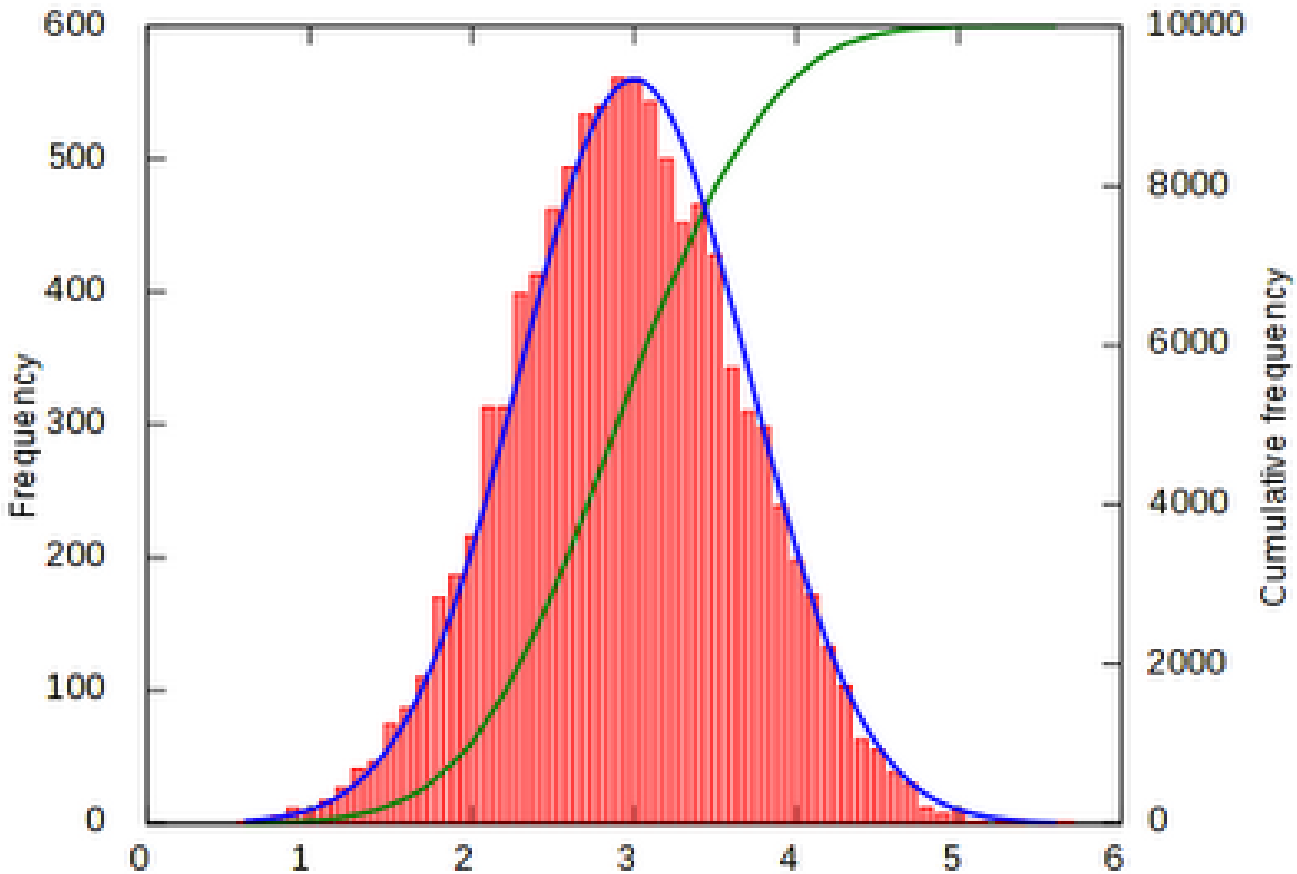
$$\alpha(x; \mathcal{D}_n) = \mathbb{E}_{v|x,\theta}[U(x, v, \theta)] \tag{7}$$

Let's assume we have observed some $\mathbf{X}$ and $\mathbf{y}$.

$$\mu_n(\mathbf{x}) = \mu_0(\mathbf{x}) + k(\mathbf{x})^T (K + \sigma^2 I)^{-1} (\mathbf{y} - m)$$
$$\sigma_n^2(\mathbf{x}) = k(\mathbf{x})^T (K + \sigma^2 I)^{-1} k(\mathbf{x})$$

Improvement-based

$$\alpha_{PI}(x; \mathcal{D}_n) = \Phi\left(\frac{\mu_n(x) - \tau}{\sigma_n(x)}\right) \tag{8}$$



Optimistic :)

$$\alpha_{UCB}(x; \mathcal{D}_n) = \mu_n(\mathbf{x}) + \beta\sigma_n^2(\mathbf{x}) \tag{9}$$

Information theoretical

$$\alpha_{ES}(x; \mathcal{D}_n) = H(\mathbf{x}^*|\mathcal{D}_n) - H_{y|\mathcal{D}_n,x}(\mathbf{x}^*|\mathcal{D}_n \cup \{(\mathbf{x}, \mathbf{y})\}) \tag{10}$$

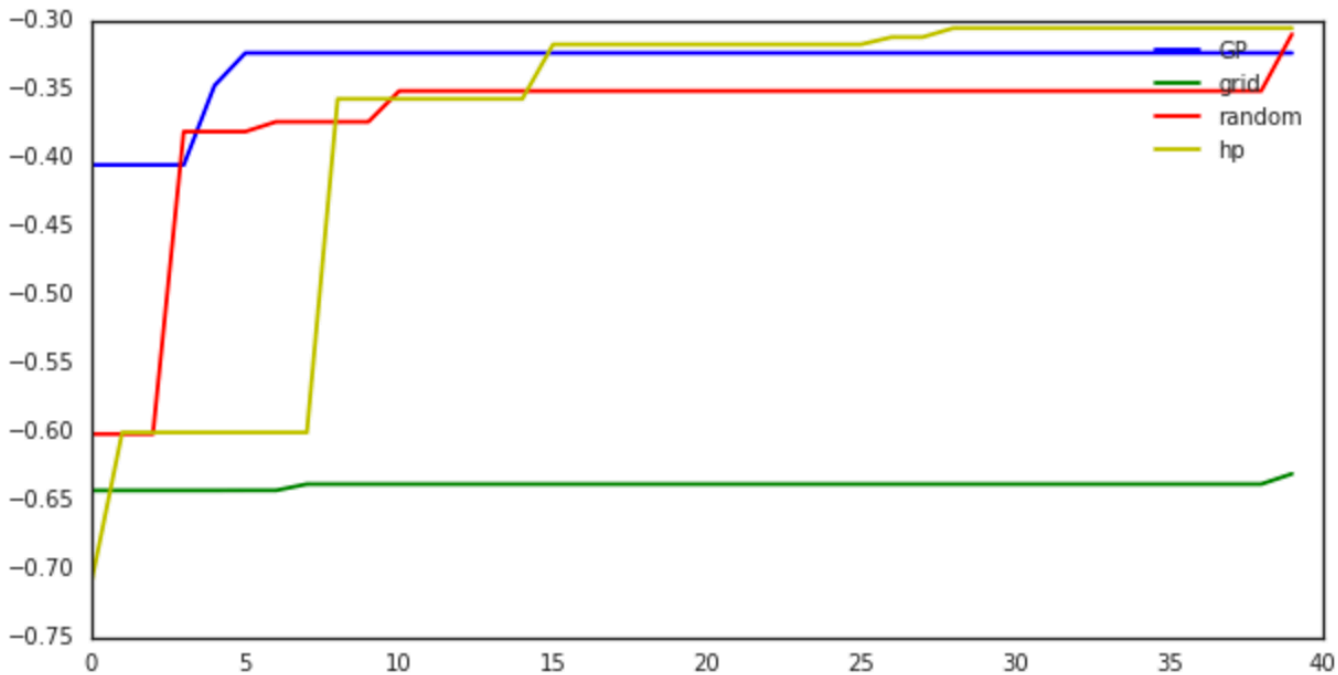
Note: we need here posterior distribution $p_*(\mathbf{x}|\mathcal{D}_n)$ which is very hard to compute.

Caveats and practical considerations of GPs in BO

- Hard to handle conditional hyperparameters (hyperopt uses tree based prob. estimation)
- $O(n^3)$ inference complexity (reducible using many tricks)
- Most BO algorithms are sequential
- No overall *best* acquisition function

Simple experiment on virtual screening

Fitting SVM RBF for 10 values of γ and 12 values of C to 5-HT2a with KRFP fingerprint (using *dr.GMUM*)



Thank you