

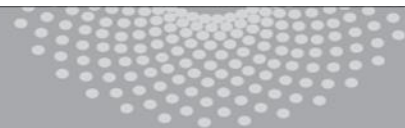
# Introduction to Machine Learning

## Active Learning

Barnabás Póczos



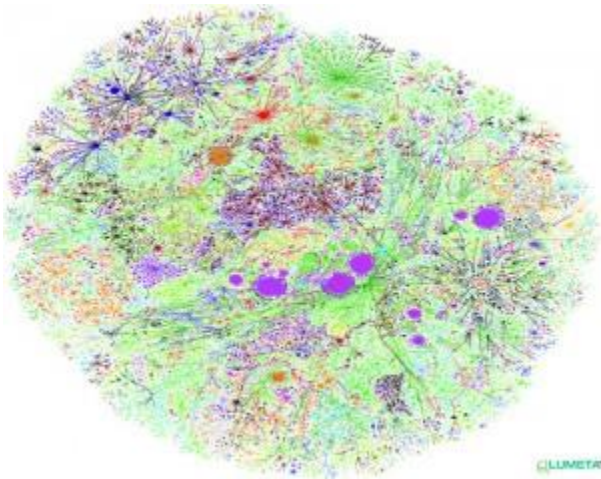
**MACHINE LEARNING** DEPARTMENT



**Carnegie Mellon.**  
School of Computer Science

# Classic Supervised Learning Paradigm is Insufficient Nowadays

Modern applications: **massive amounts** of raw data.  
Only a **tiny fraction** can be annotated by human experts.



Billions of webpages



Images



Sensor measurements

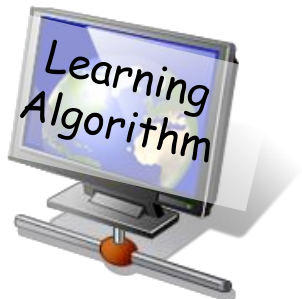
# Modern ML: New Learning Approaches

Modern applications: **massive amounts** of raw data.



**The Large Synoptic  
Survey Telescope**  
15 Terabytes of data ...  
every night

We need techniques that **minimize need for expert/human  
intervention** => Active Learning



# Contents

## ❑ Active Learning Intro

- Batch Active Learning vs Selective Sampling Active Learning
- Exponential Improvement on # of labels
- Sampling bias: Active Learning can hurt performance

## ❑ Active Learning with SVM

## ❑ Gaussian Processes

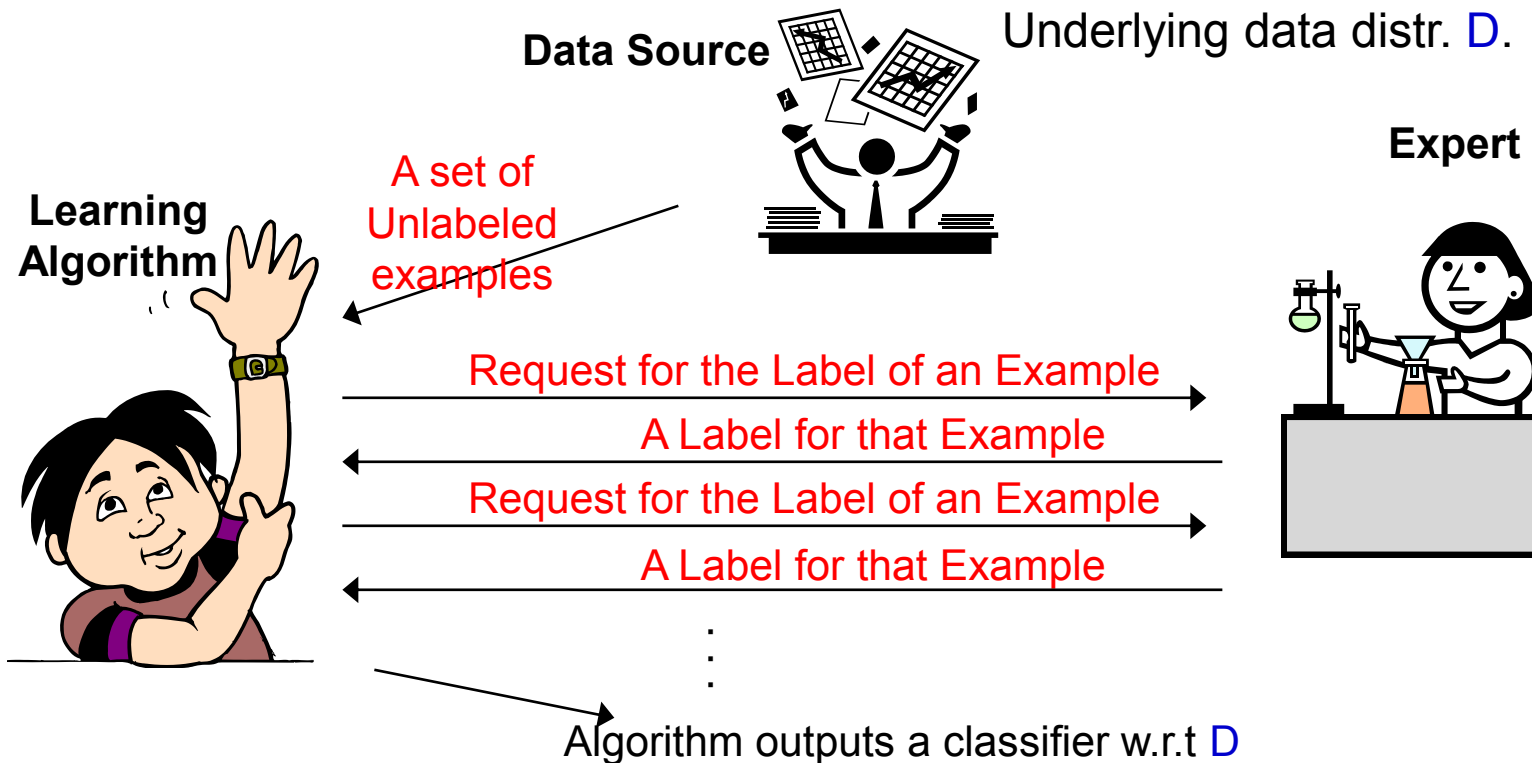
- Regression
- Properties of Multivariate Gaussian distributions
- Ridge regression
- GP = Bayesian Ridge Regression + Kernel trick

## ❑ Active Learning with Gaussian Processes

# Additional resources

- **Two faces of active learning.** Sanjoy Dasgupta. 2011.
- **Active Learning.** Bur Settles. 2012.
- **Active Learning.** Balcan-Urner. Encyclopedia of Algorithms. 2015

# Batch Active Learning

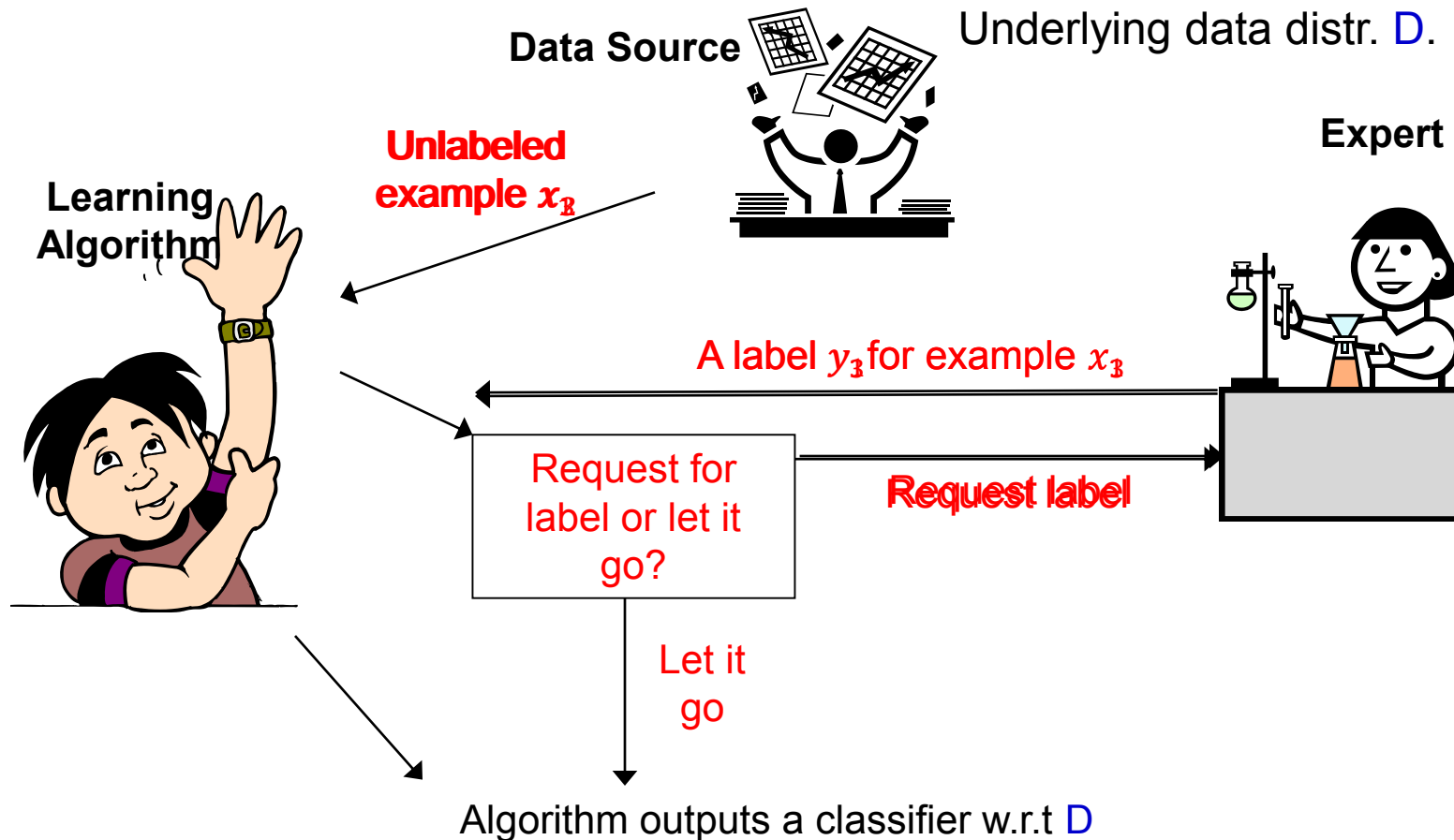


- **Learner can choose specific examples to be labeled.**

[pick **informative** examples to be labeled].

- **Goal:** use fewer labeled examples

# Selective Sampling Active Learning



- **Selective sampling AL (Online AL):** stream of unlabeled examples, when each arrives make a decision to ask for label or not.  
[pick **informative** examples to be labeled].
- **Goal:** use fewer labeled examples

# What Makes a Good Active Learning Algorithm?

- Guaranteed to output a relatively good classifier for most learning problems.
- Doesn't make too many label requests.  
Hopefully a lot less than passive learning.
- Need to choose the label requests carefully, to get **informative** labels.

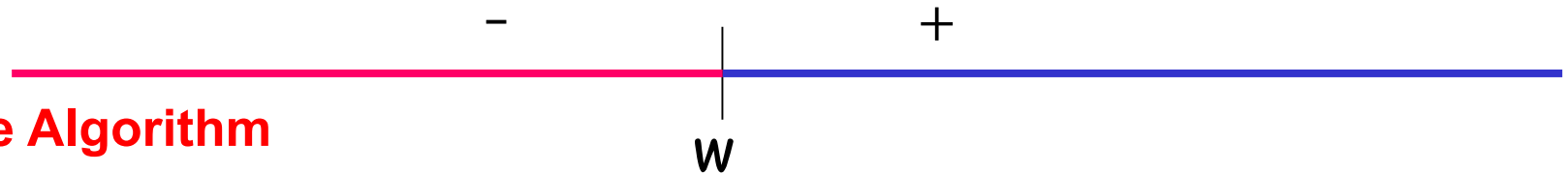


# Can adaptive querying really do better than passive/random sampling?

- YES! (sometimes)
- We often need far fewer labels for active learning than for passive.
- This is predicted by theory and has been observed in practice.

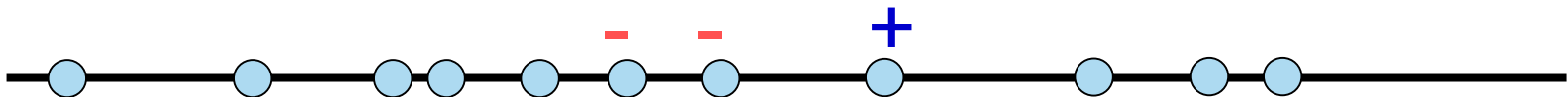
# Can adaptive querying help? [CAL92, Dasgupta04]

- Threshold fns on the real line:  $h_w(x) = 1(x \geq w)$ ,  $C = \{h_w: w \in \mathbb{R}\}$



## Active Algorithm

- Get  $N$  unlabeled examples
- How can we recover the correct labels with  $\ll N$  queries?
- Do binary search (query at half)! Just need  $O(\log N)$  labels!



- Output a classifier consistent with the  $N$  inferred labels.

- $N = O(1/\epsilon)$  we are guaranteed to get a classifier of error  $\leq \epsilon$ .

**Passive supervised:**  $\Omega(1/\epsilon)$  labels to find an  $\epsilon$ -accurate threshold

**Active:** only  $O(\log 1/\epsilon)$  labels. Exponential improvement.



# Active SVM

Uncertainty sampling in SVMs common and quite useful in practice.

E.g., [Tong & Koller, ICML 2000; Jain, Vijayanarasimhan & Grauman, NIPS 2010; Schohn Cohn, ICML 2000]

## Active SVM Algorithm

- At any time during the alg., we have a “current guess”  $w_t$  of the separator: the max-margin separator of all labeled points so far.
- Request the label of the example closest to the current separator.

# Active SVM

Active SVM seems to be quite useful in practice.

[Tong & Koller, ICML 2000; Jain, Vijayanarasimhan & Grauman, NIPS 2010]

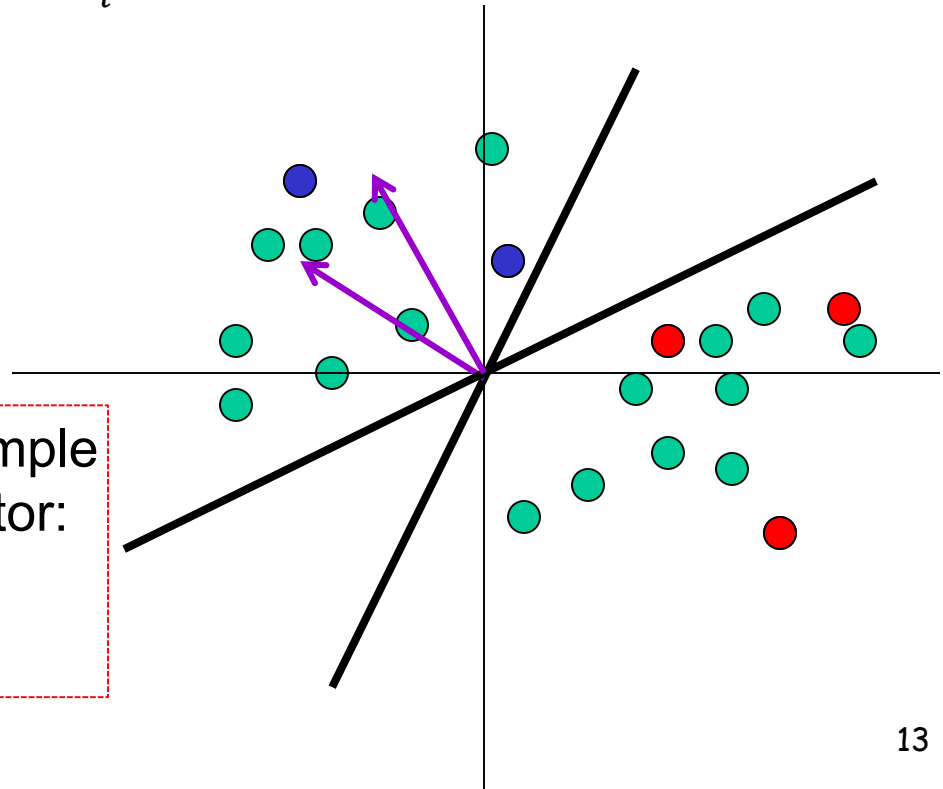
## Algorithm (batch version)

Input  $S_u = \{x_1, \dots, x_{m_u}\}$  drawn i.i.d from the underlying source  $D$

Start: query for the labels of a few random  $x_i$ s.

For  $t = 1, \dots,$

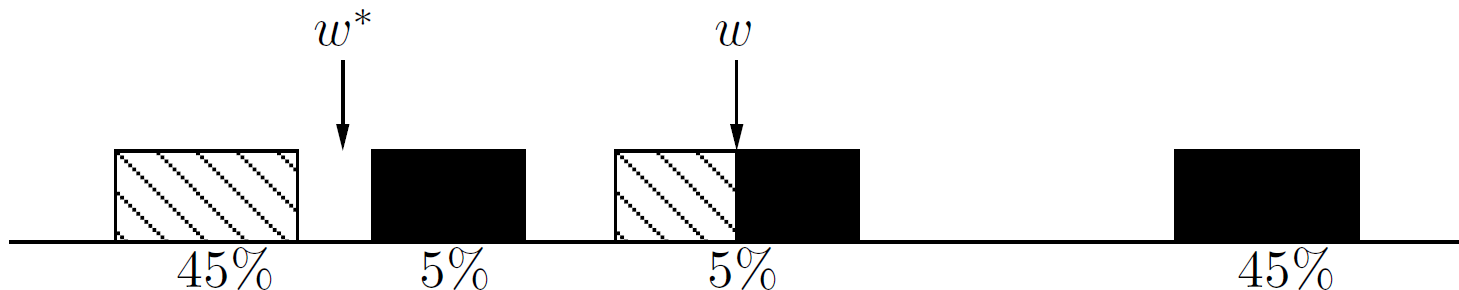
- Find  $w_t$  the max-margin separator of all labeled points so far.
- Request the label of the example closest to the current separator: minimizing  $|x_i \cdot w_t|$ .  
(highest uncertainty)



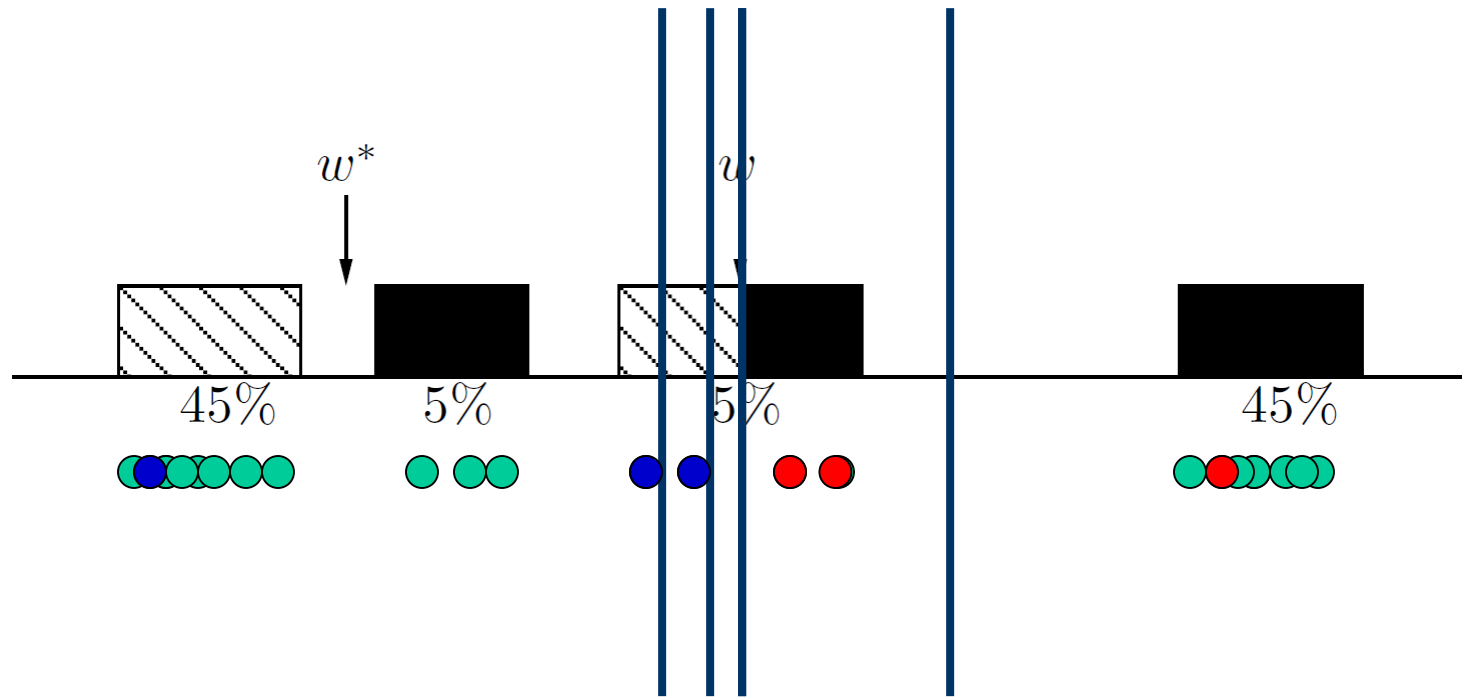
# DANGER!!!

- Uncertainty sampling works sometimes....
- **However, we need to be very very very careful!!!**
  - Myopic, greedy techniques can suffer from **sampling bias**.  
(The active learning algorithm samples from a different (x,y) distribution than the true data)
  - A bias created because of the querying strategy; as time goes on the sample is less and less representative of the true data source.

[Dasgupta10]



# DANGER!!!

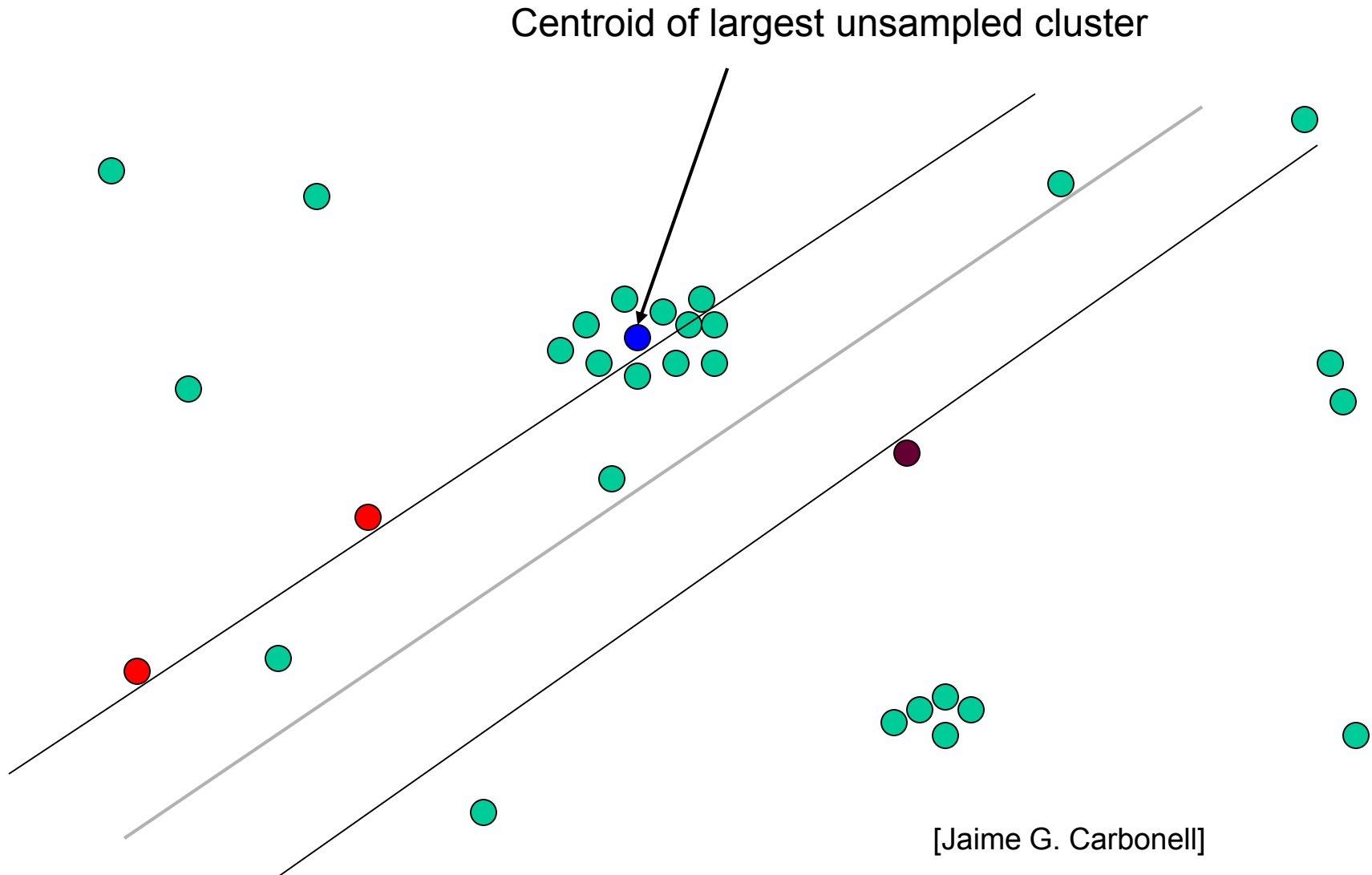


- Observed in practice too!!!!
- **Main tension:** want to choose informative points, but also want to guarantee that the classifier we output does well on true random examples from the underlying distribution.

# Other Interesting Active Learning Techniques used in Practice

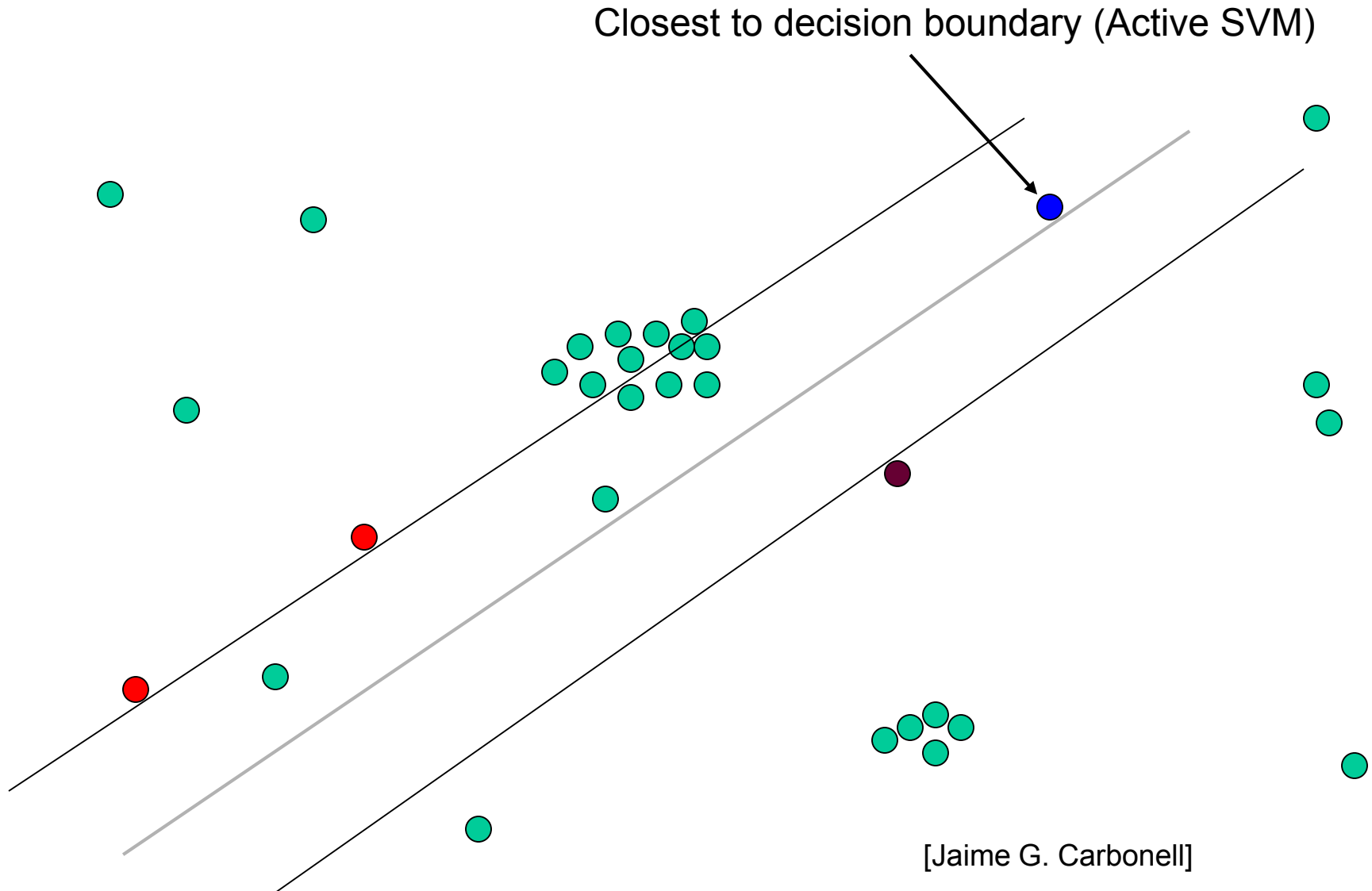
Interesting open question to analyze under what conditions they are successful.

# Density-Based Sampling

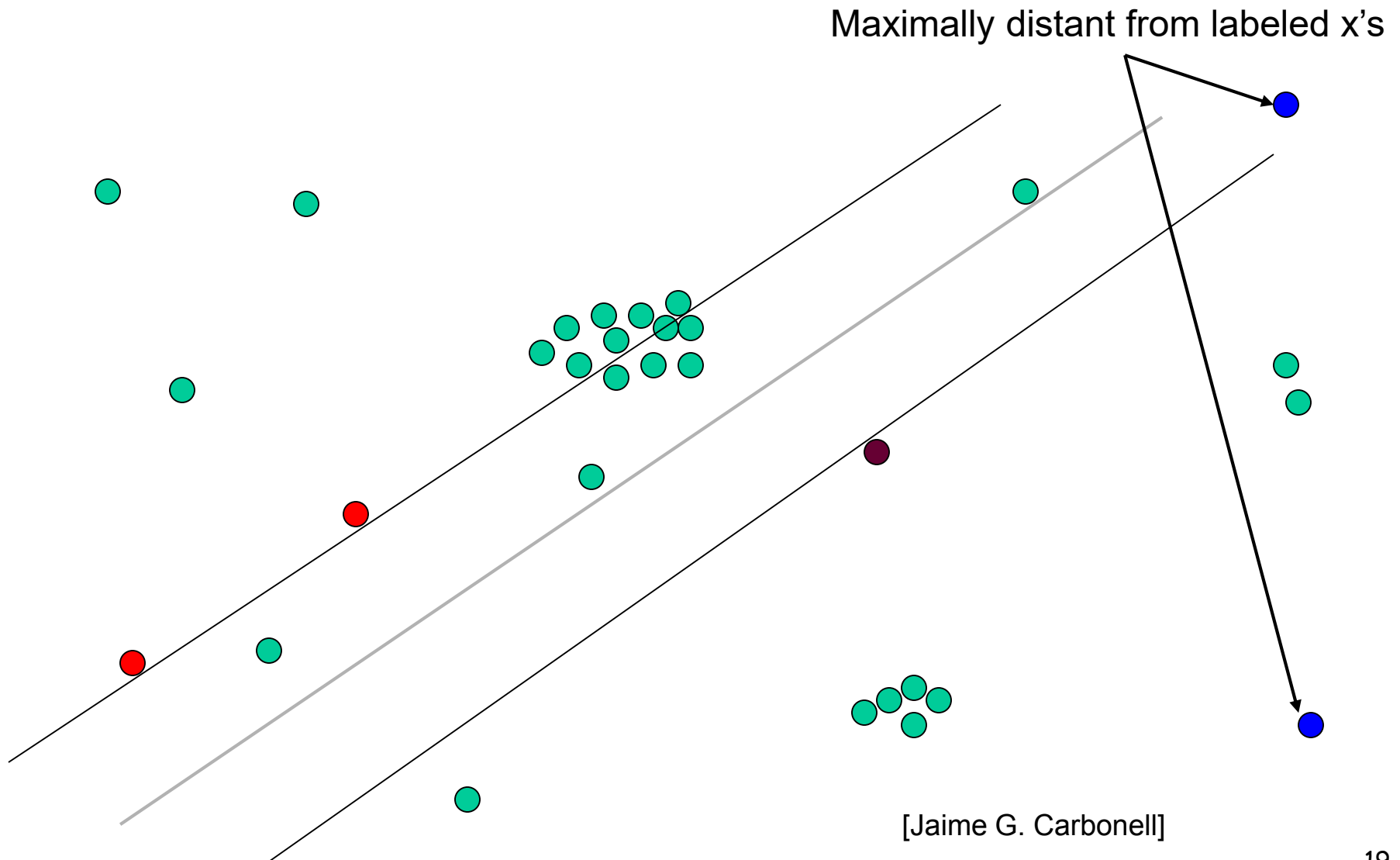




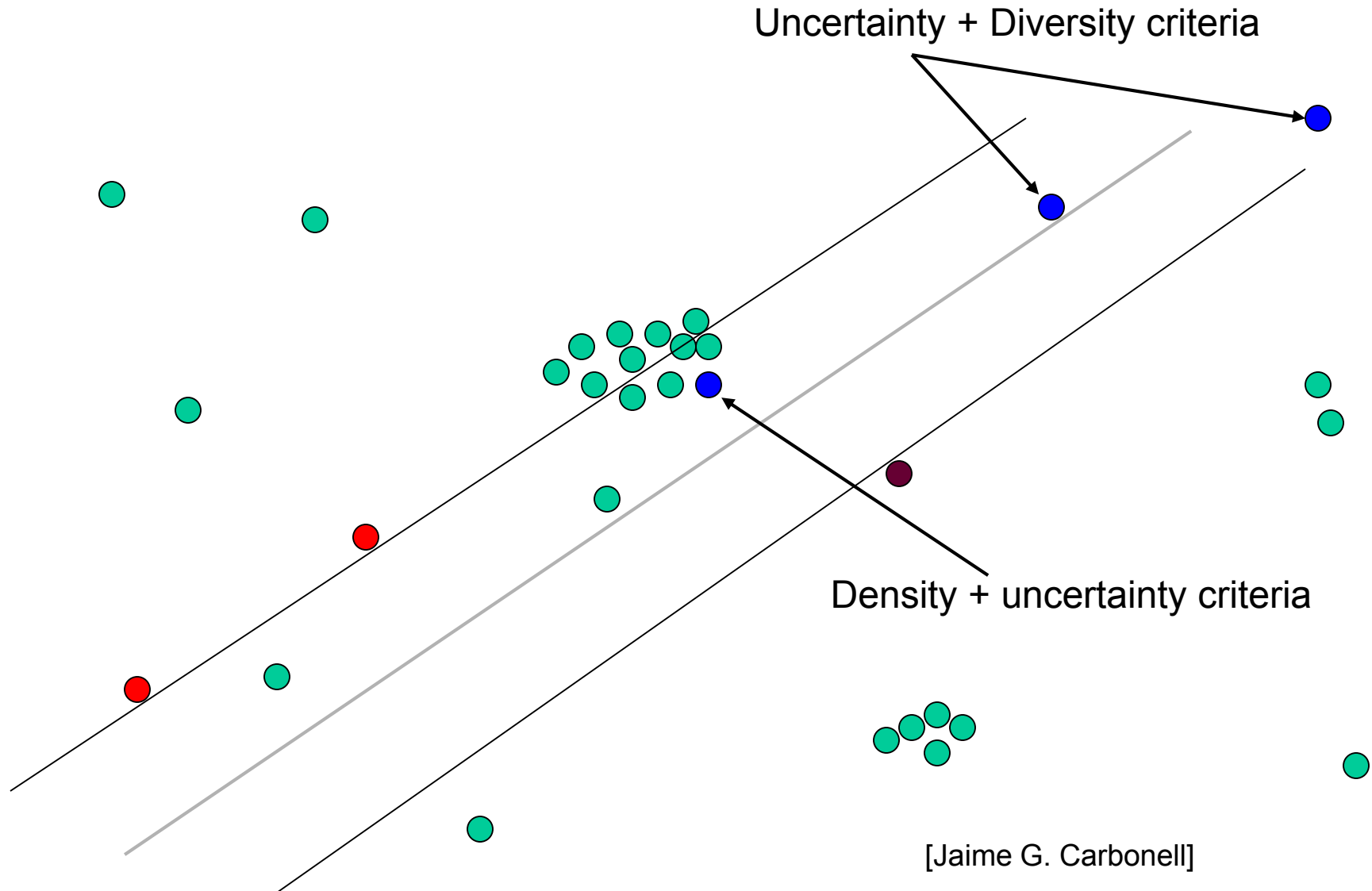
# Uncertainty Sampling



# Maximal Diversity Sampling



# Ensemble-Based Possibilities

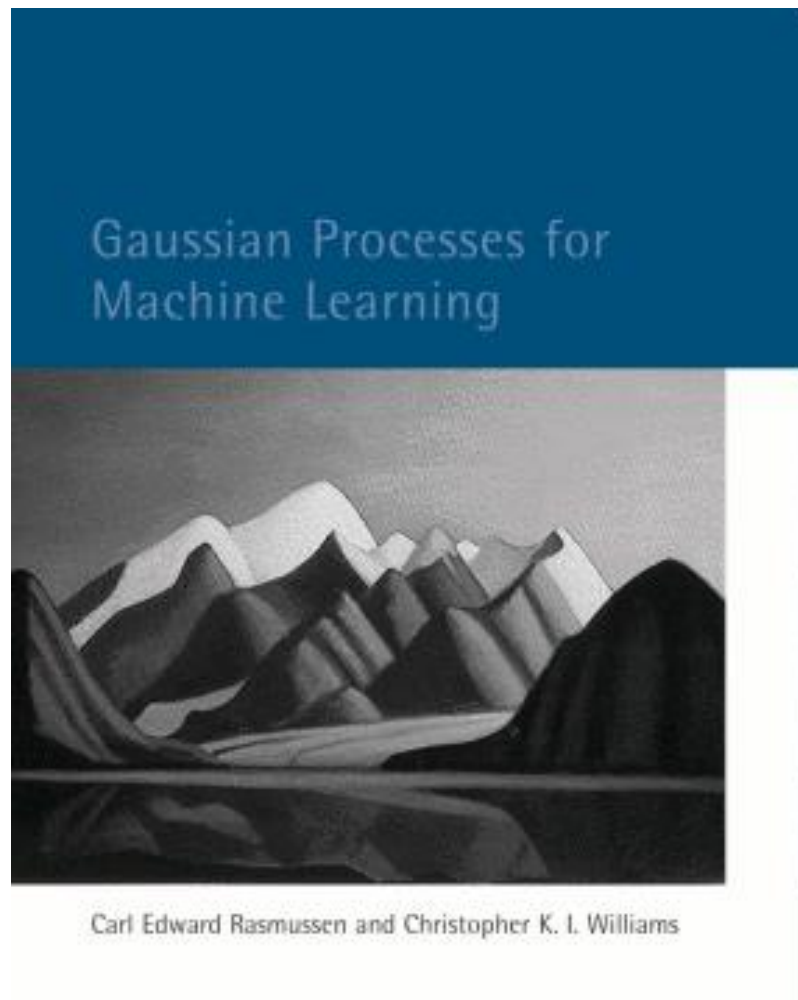


# What You Should Know so far

- ❑ **Active learning could be really helpful**, could provide exponential improvements in label complexity (both theoretically and practically)!
- ❑ **Need to be very careful due to sampling bias.**
- ❑ **Common heuristics**
  - (e.g., those based on uncertainty sampling).

# Gaussian Processes for Regression

# Additional resources



**<http://www.gaussianprocess.org/>**

Some of these slides are taken from D. Lizotte, R. Parr, C. Guesterin

# Additional resources

- **Nonmyopic Active Learning of Gaussian Processes: An Exploration–Exploitation Approach.** A.Krause and C. Guestrin, ICML 2007
- **Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies.** A.Krause, A. Singh, and C. Guestrin, Journal of Machine Learning Research 9 (2008)
- **Bayesian Active Learning for Posterior Estimation,** Kandasamy, K., Schneider, J., and Póczos, B, International Joint Conference on Artificial Intelligence (IJCAI), 2015

# Why GPs for Regression?

## Regression methods:

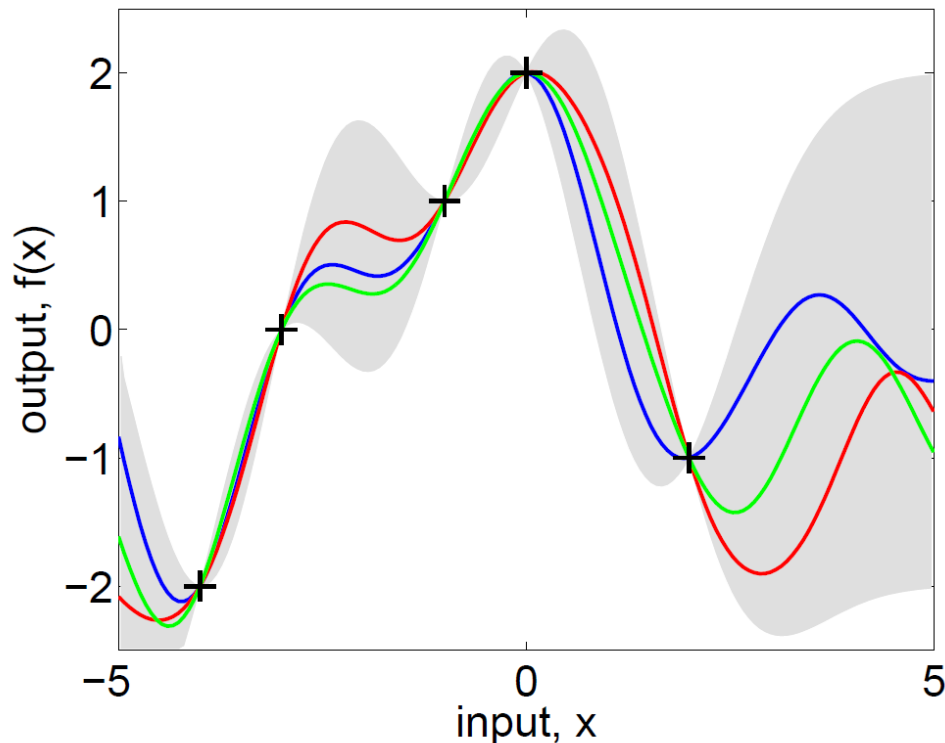
Linear regression, multilayer perceptron, ridge regression, support vector regression, kNN regression, etc...

## Motivation:

All the above regression methods give point estimates. We would like a method that could also provide confidence during the estimation.

## Application in Active Learning:

This method can be used for active learning: query the next point and its label where the uncertainty is the highest

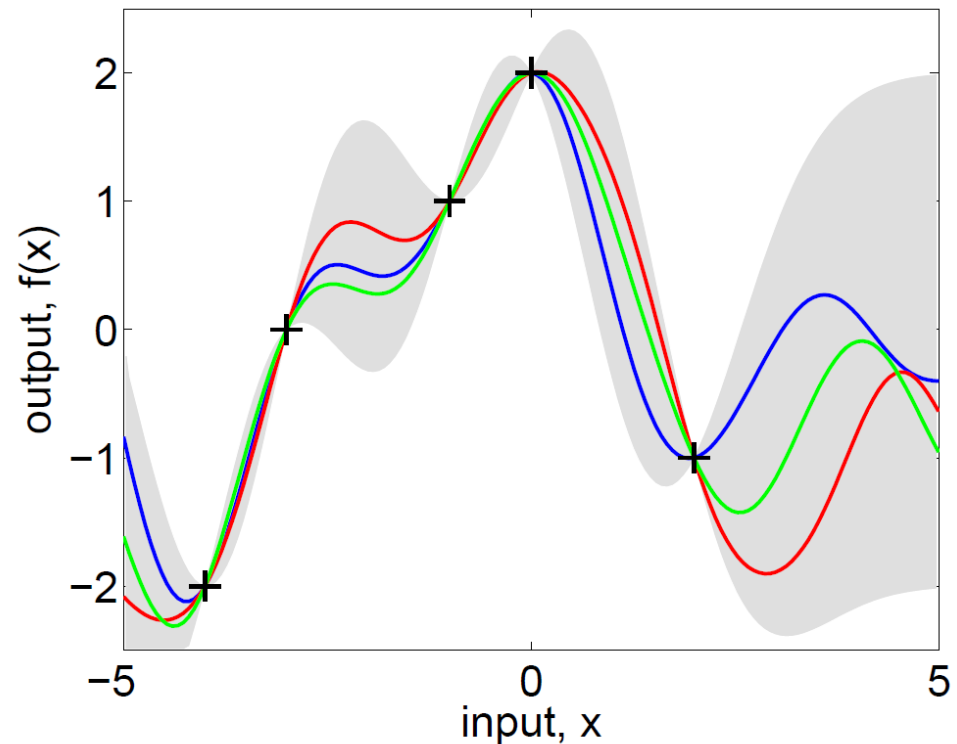




# Why GPs for Regression?

## GPs can answer the following questions:

- Here's where the function will **most likely be**.  
(expected function)
- Here are some **examples** of what it might look like.  
(sampling from the posterior distribution [blue, red, green functions])
- Here is a prediction of what you'll see if you evaluate your function at  $x'$ , **with confidence**

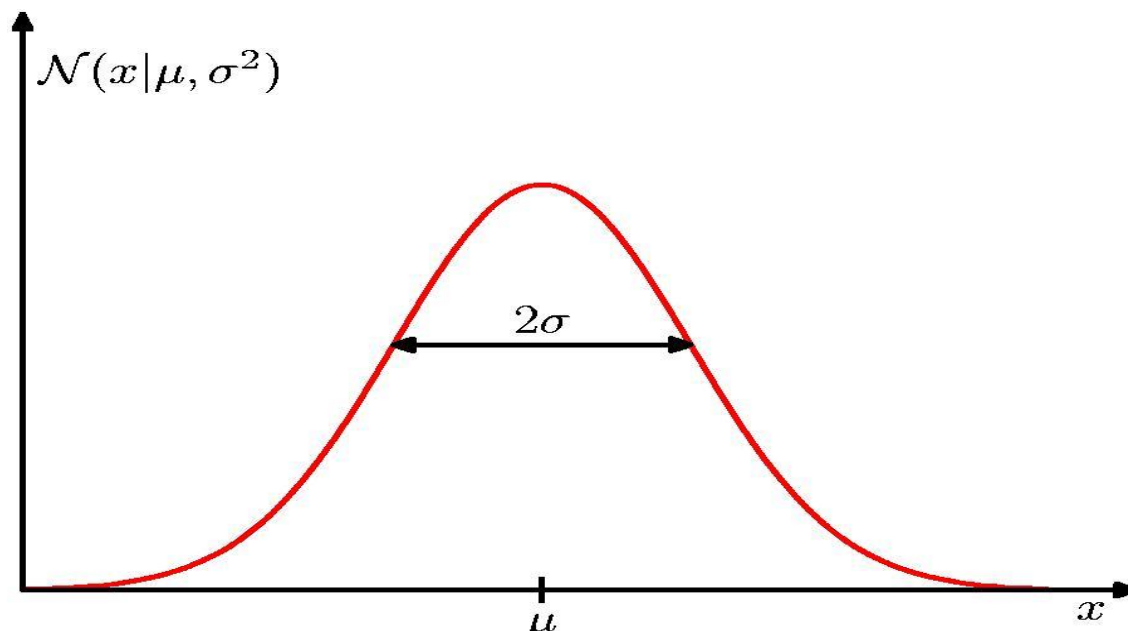


# Properties of Multivariate Gaussian Distributions

# 1D Gaussian Distribution

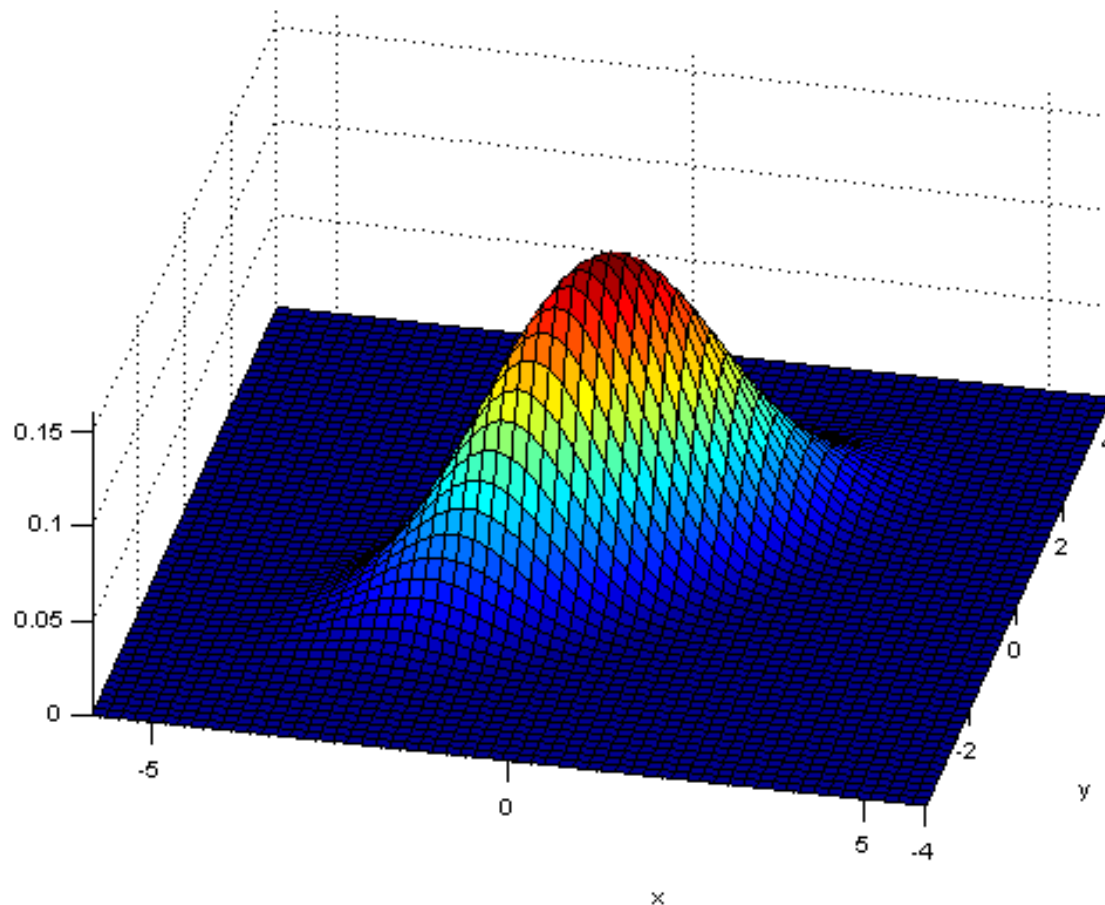
## Parameters

- Mean,  $\mu$
- Variance,  $\sigma^2$



$$P(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

# Multivariate Gaussian



$$p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{|\mathbf{2}\pi\boldsymbol{\Sigma}|}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

# Multivariate Gaussian

□ A 2-dimensional Gaussian is defined by

- a mean vector  $\mu = [\mu_1, \mu_2]$

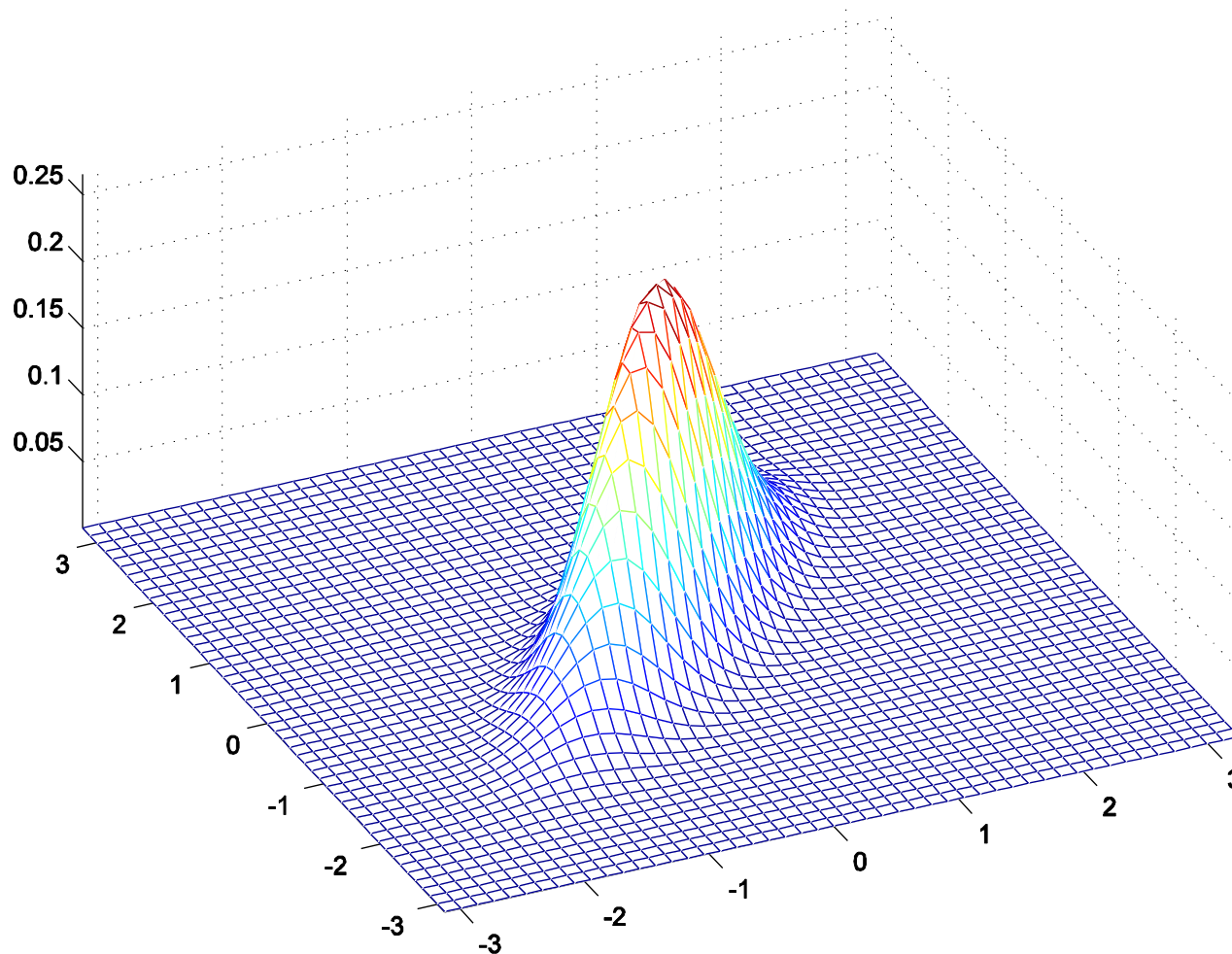
- a covariance matrix:  $\Sigma = \begin{bmatrix} \sigma_{1,1}^2 & \sigma_{2,1}^2 \\ \sigma_{1,2}^2 & \sigma_{2,2}^2 \end{bmatrix}$

where  $\sigma_{i,j}^2 = E[(x_i - \mu_i)(x_j - \mu_j)]$   
is (co)variance

□ Note:  $\Sigma$  is symmetric,

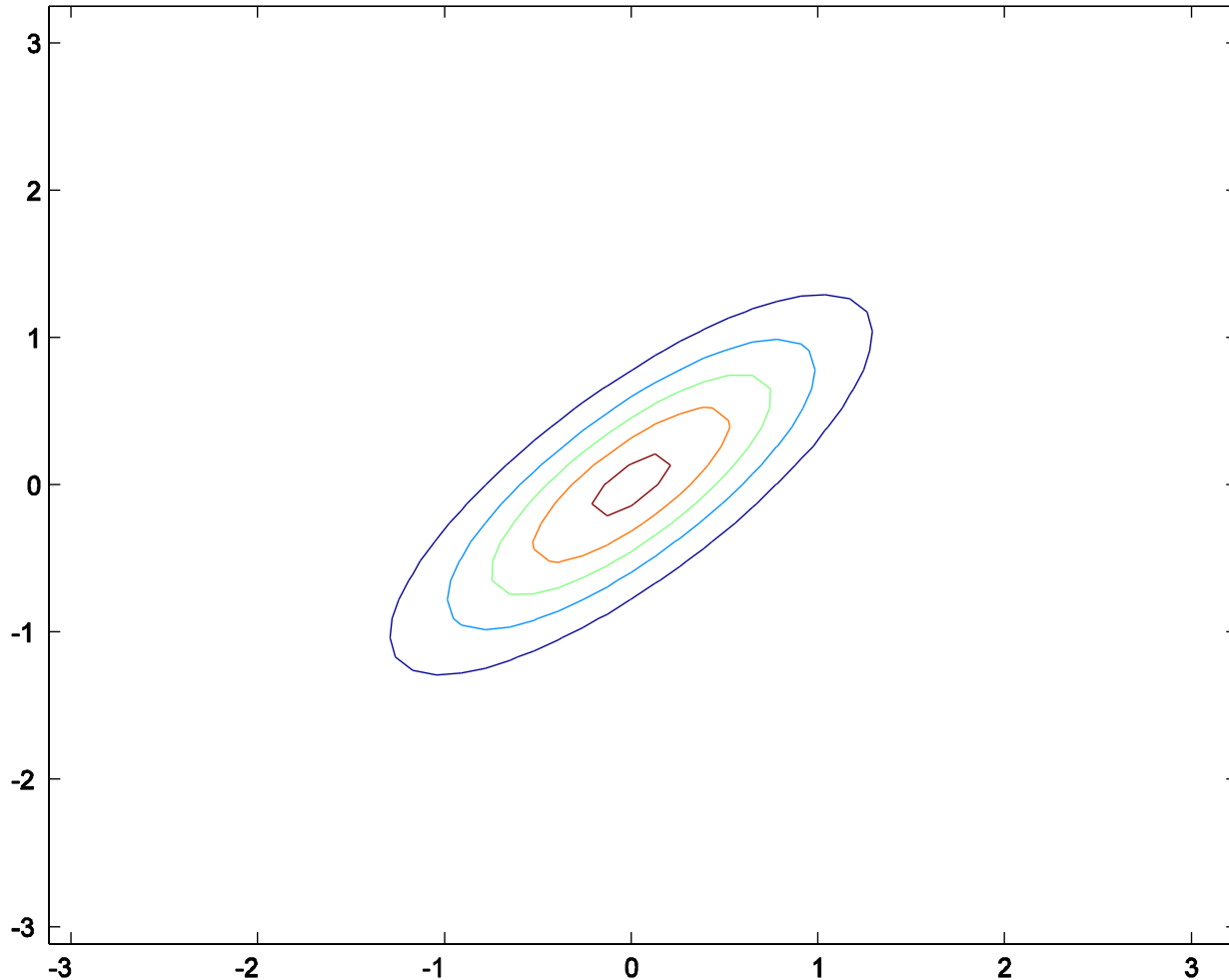
“positive semi-definite”:  $\forall \mathbf{x}: \mathbf{x}^T \Sigma \mathbf{x} \geq 0$

# Multivariate Gaussian examples



$$\mu = (0,0) \quad \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

# Multivariate Gaussian examples



$$\mu = (0,0) \quad \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

# Useful Properties of Gaussians

□ Marginal distributions of Gaussians are Gaussian

□ Given:

$$x = (x_a, x_b), \mu = (\mu_a, \mu_b)$$

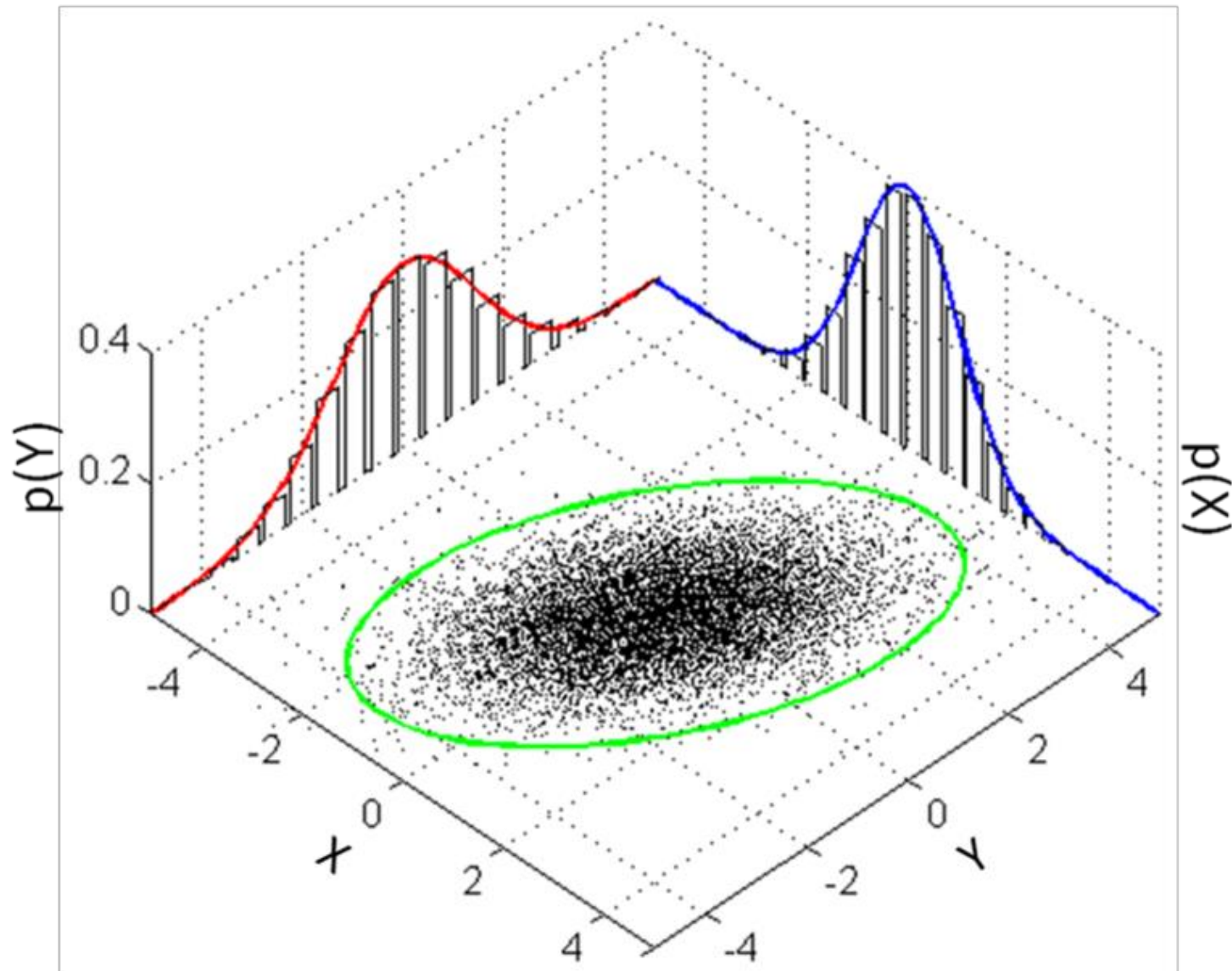
$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$$

The marginal distribution is:

$$p(X_a) = \mathcal{N}(x_a \mid \mu_a, \Sigma_{aa})$$



# Marginal distributions of Gaussians are Gaussian



# Block Matrix Inversion

## Theorem

$$\begin{aligned} \begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} &= \begin{bmatrix} (A - BD^{-1}C)^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} \\ &= \begin{bmatrix} S_D^{-1} & -A^{-1}BS_A^{-1} \\ -D^{-1}CS_D^{-1} & S_A^{-1} \end{bmatrix} \end{aligned}$$

## Definition: Schur complements

Schur complements of  $A$ :  $S_A = D - CA^{-1}B$

Schur complements of  $D$ :  $S_D = A - BD^{-1}C$

# Useful Properties of Gaussians

□ Conditional distributions of Gaussians are Gaussian

□ Notation:

$$\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \quad \Lambda = \Sigma^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

□ **Conditional Distribution:**

$$p(X_a | X_b) = \mathcal{N}(x_a \mid \mu_{a|b}, \Lambda_{aa}^{-1})$$

$$\mu_{a|b} = \mu_a - \Lambda_{aa}^{-1} \Lambda_{ab} (\mathbf{x}_b - \mu_b) = \mu_a + \Sigma_{ab} \Sigma_{bb}^{-1} (\mathbf{x}_b - \mu_b)$$

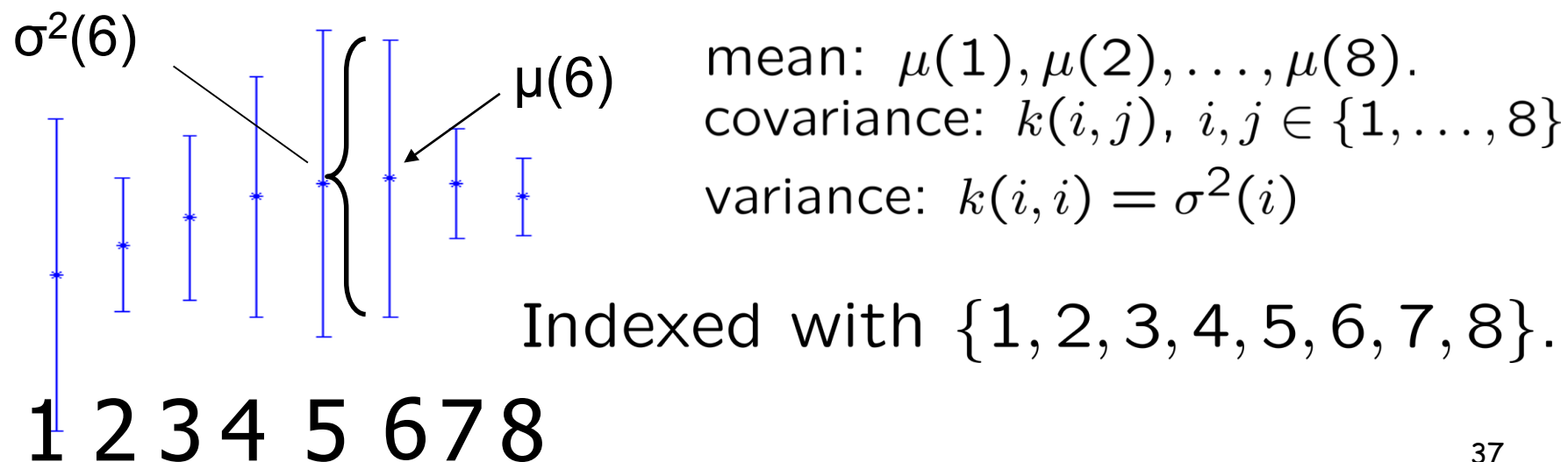
$$\Lambda_{aa}^{-1} = \Sigma_{aa} - \Sigma_{ab} \Sigma_{bb}^{-1} \Sigma_{ba}$$

Schur complement of  $\Sigma_{bb}$  in  $\Sigma$

# Higher Dimensions

- ❑ **Visualizing  $> 3$  dimensional Gaussian random variables is... difficult**
- ❑ Means and variances of marginal variables are practical, but then we don't see correlations between those variables
- ❑ Marginals are Gaussian, e.g.,  $f(6) \sim N(\mu(6), \sigma^2(6))$

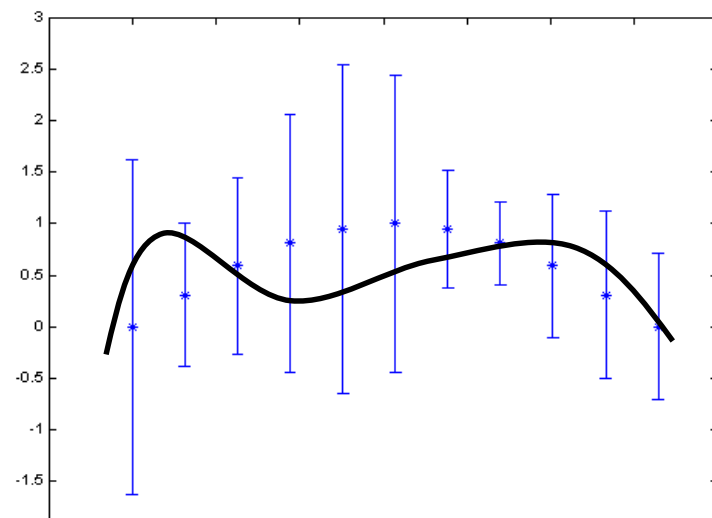
## Visualizing an 8-dimensional Gaussian variable $\mathbf{f}$ :



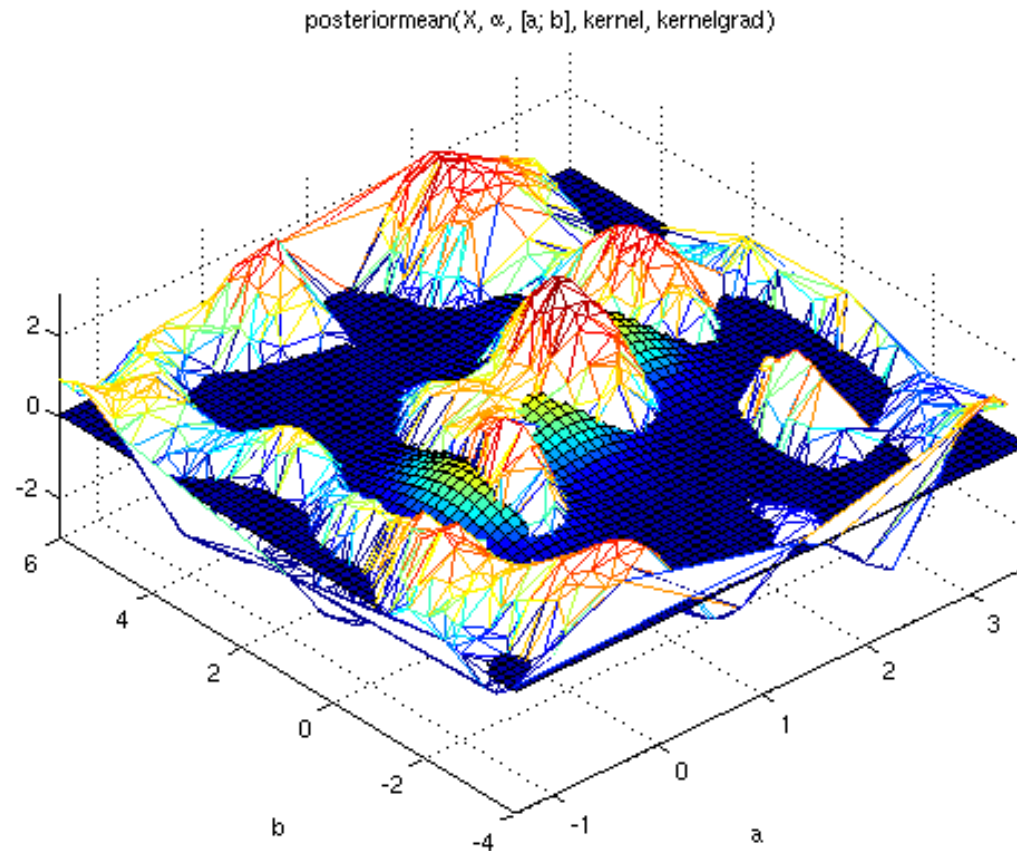
# Yet Higher Dimensions

## Why stop there?

- We indexed before with  $\{1, 2, 3, 4, 5, 6, 7, 8\}$ .
- Why not indexing with  $\mathbb{Z}$ , or  $\mathbb{R}$ ?
- Need functions  $\mu(x), k(x, z), \forall x, z \in \mathbb{R}$
- $x$  and  $z$  are indexes over the random variables
- $f$  is now an uncountably infinite dimensional vector
- Depending on  $\mu(\cdot)$  and  $k(\cdot, \cdot)$ ,  $f$  can be continuous or even infinitely differentiable too!



# Getting Ridiculous



## Why stop there?

- We indexed before with  $\mathbb{R}$ , why not with  $\mathbb{R}^D$ ?
- Need functions  $\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{z}), \forall \mathbf{x}, \mathbf{z} \in \mathbb{R}^D$

# Gaussian Process

## Definition of GP:

- ❑ Probability distribution *indexed by* an arbitrary set (integer, real, finite dimensional vector, etc)
- ❑ Each element (indexed by  $x$ ) is a Gaussian distribution over the reals with mean  $\mu(x)$
- ❑ These distributions are dependent/correlated as defined by  $k(x,z)$
- ❑ Any finite subset of indices defines a multivariate Gaussian distribution

# Gaussian Process

## □ Distribution over *functions*....

*If our regression model is a GP, then it won't be a point estimate anymore! It can provide regression estimates with confidence*

## □ Domain (index set) of the functions can be pretty much whatever

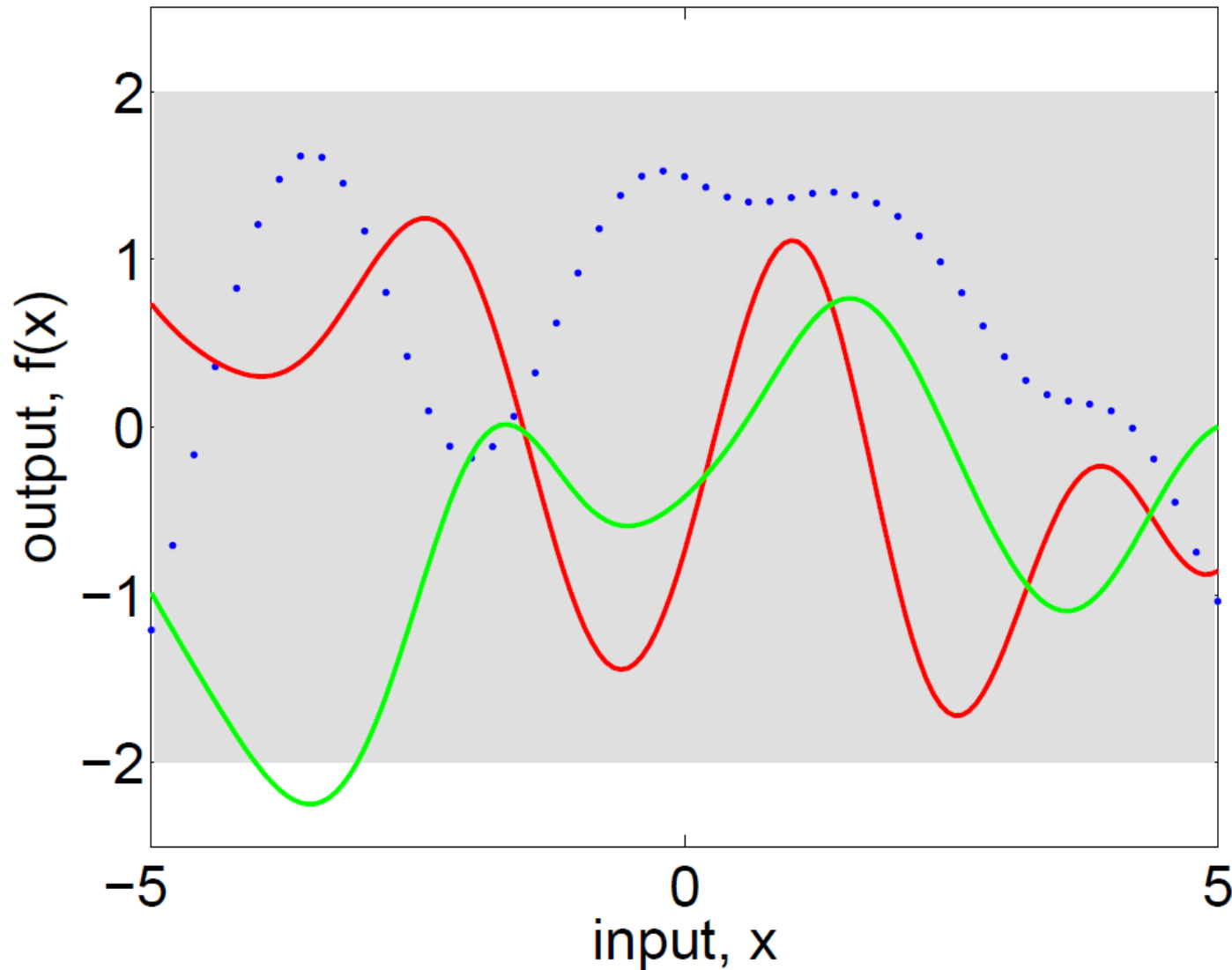
- Reals
- Real vectors
- Graphs
- Strings
- Sets
- ...



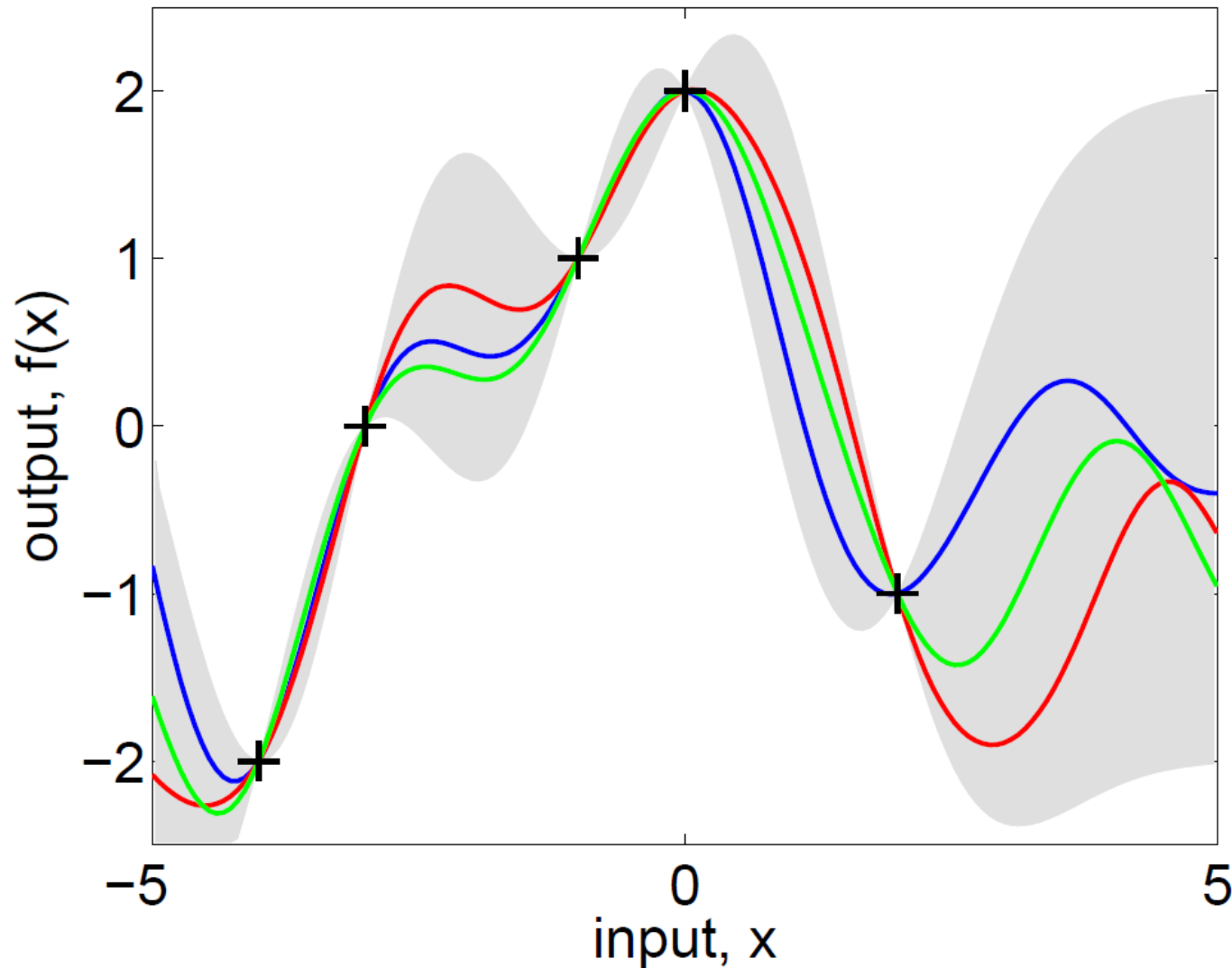
# Bayesian Updates for GPs

- How can we do regression and learn the GP from data?
- We will be Bayesians today:
  - Start with GP prior
  - Get some data
  - Compute a posterior

# Samples from the prior distribution

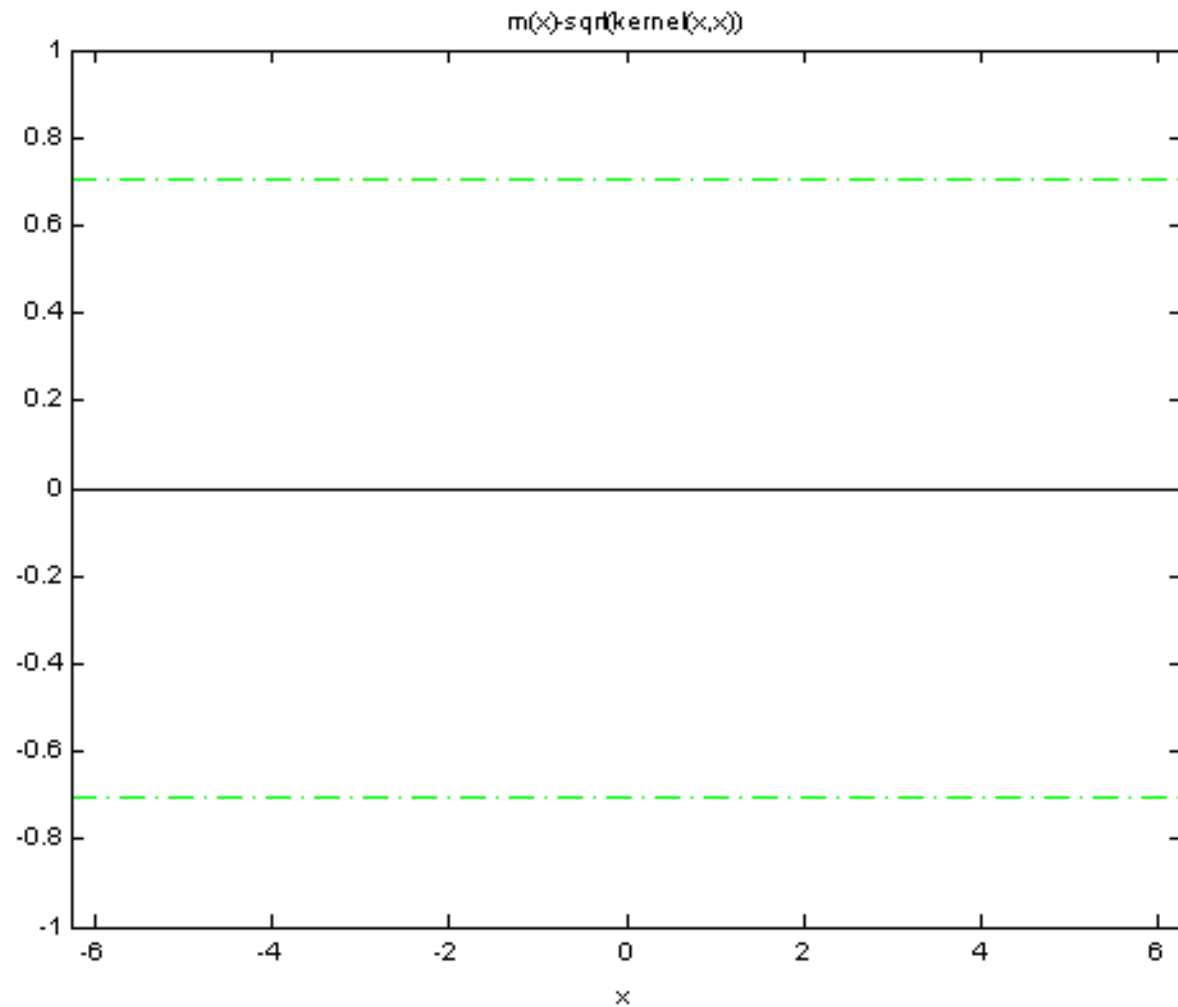


# Samples from the posterior distribution

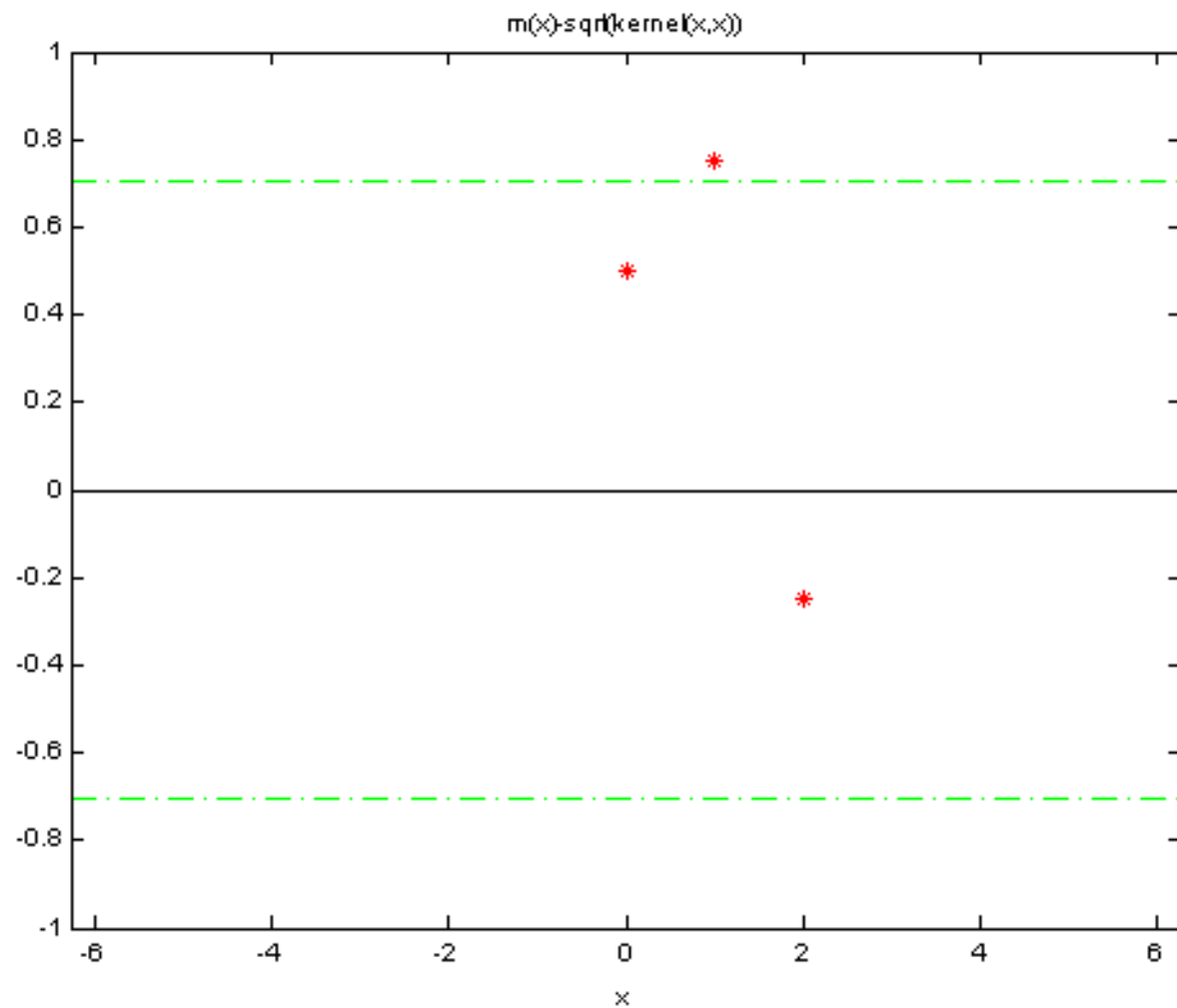


# Prior

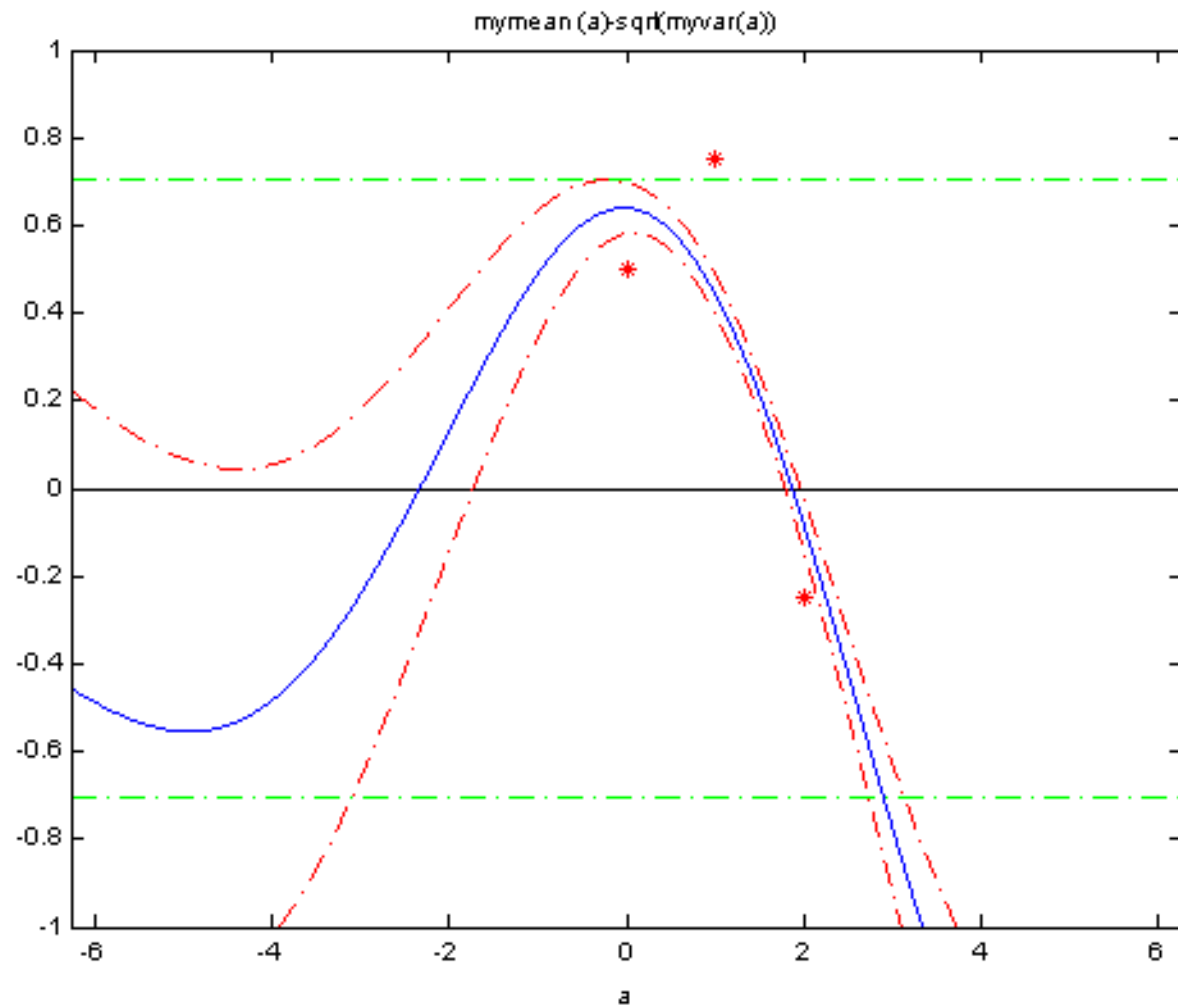
Zero mean Gaussians with covariance  $k(x,z)$



# Data



# Posterior



# Ridge Regression

Training set:  $\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1, \dots, n\}$ ,  $\mathbf{x}_i \in \mathbb{R}^D$ ,  $y_i \in \mathbb{R}$

**Linear regression:**  $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w} \in \mathbb{R}^D} \sum_{i=1}^n (y_i - \langle \mathbf{x}_i, \mathbf{w} \rangle)^2$$

**Ridge regression:**

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w} \in \mathbb{R}^D} \sum_{i=1}^n (y_i - \langle \mathbf{x}_i, \mathbf{w} \rangle)^2 + \lambda \|\mathbf{w}\|^2$$

**The Gaussian Process is a Bayesian Generalization  
of the kernelized ridge regression**

# Weight Space View

**GP = Bayesian ridge regression in feature space  
+ Kernel trick to carry out computations**

Training set:  $\mathcal{D} = \{(\mathbf{x}_i, y_i) | i = 1, \dots, n\}$ ,  $\mathbf{x}_i \in \mathbb{R}^D$ ,  $y_i \in \mathbb{R}$

$$\left. \begin{aligned} X &= \begin{bmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^{D \times n}, \text{ design matrix} \\ y &= \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n \end{aligned} \right\} \text{The training data}$$



# Bayesian Analysis of Linear Regression with Gaussian noise

Linear regression:  $f(\mathbf{x}) = \mathbf{x}^T \mathbf{w} \in \mathbb{R}$ ,  $\mathbf{x}, \mathbf{w} \in \mathbb{R}^D$

Linear regression with noise:  $y = f(\mathbf{x}) + \epsilon = \mathbf{x}^T \mathbf{w} + \epsilon \in \mathbb{R}$   
 $\epsilon \sim \mathcal{N}(0, \sigma^2) \in \mathbb{R}$

Let us calculate the likelihood:

$$P(\mathbf{y} | X, \mathbf{w}) = \prod_{i=1}^n P(y_i | \mathbf{x}_i^T \mathbf{w})$$

and then put  $\mathbf{w} \sim \mathcal{N}_{\mathbf{w}}(0, \Sigma_p)$  prior over parameters  $\mathbf{w}$ .

# Bayesian Analysis of Linear Regression with Gaussian noise

**The likelihood:**

$$\begin{aligned} P(\mathbf{y}|\mathbf{X}, \mathbf{w}) &= \prod_{i=1}^n P(y_i|\mathbf{x}_i^T \mathbf{w}) \\ &= \prod_{i=1}^n \mathcal{N}_{y_i}(\mathbf{x}_i^T \mathbf{w}, \sigma^2) \\ &= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[ \frac{-(y_i - \mathbf{x}_i^T \mathbf{w})^2}{2\sigma^2} \right] \\ &= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp \left[ \frac{-1}{2\sigma^2} \|\mathbf{y} - \mathbf{X}^T \mathbf{w}\|^2 \right] \\ &= \mathcal{N}_{\mathbf{y}}(\mathbf{X}^T \mathbf{w}, \sigma^2 \mathbf{I}_n) \end{aligned}$$

# Bayesian Analysis of Linear Regression with Gaussian noise

**The prior:**

$$\mathbf{w} \sim \mathcal{N}_{\mathbf{w}}(0, \Sigma_p)$$

**Now, we can calculate the posterior:**

$$\begin{aligned} P(\mathbf{w}|X, \mathbf{y}) &= \frac{P(\mathbf{y}|X, \mathbf{w})P(\mathbf{w})}{P(\mathbf{y}|X)} \\ &= \frac{P(\mathbf{y}|X, \mathbf{w})P(\mathbf{w})}{\int P(\mathbf{y}|X, \mathbf{w})d\mathbf{w}} \\ &= \frac{\mathcal{N}_{\mathbf{y}}(X^T \mathbf{w}, \sigma^2 \mathbf{I}_n) \mathcal{N}_{\mathbf{w}}(0, \Sigma_p)}{\int \mathcal{N}_{\mathbf{y}}(X^T \mathbf{w}, \sigma^2 \mathbf{I}_n) \mathcal{N}_{\mathbf{w}}(0, \Sigma_p) d\mathbf{w}} \\ &\sim \mathcal{N}_{\mathbf{y}}(X^T \mathbf{w}, \sigma^2 \mathbf{I}_n) \mathcal{N}_{\mathbf{w}}(0, \Sigma_p) \end{aligned}$$

# Bayesian Analysis of Linear Regression with Gaussian noise

$$\begin{aligned} P(\mathbf{w}|X, \mathbf{y}) &\sim \mathcal{N}_{\mathbf{y}}(X^T \mathbf{w}, \sigma^2 \mathbf{I}_n) \mathcal{N}_{\mathbf{w}}(0, \Sigma_p) && \text{Ridge Regression} \\ &\sim \exp\left\{\frac{-1}{2\sigma^2}(\mathbf{y} - X^T \mathbf{w})^T (\mathbf{y} - X^T \mathbf{w})\right\} \exp\left\{\frac{-1}{2} \mathbf{w}^T \Sigma_p^{-1} \mathbf{w}\right\} \\ &\sim \exp\left\{\frac{-1}{2}(\mathbf{w} - \bar{\mathbf{w}})^T \underbrace{\left(\frac{1}{\sigma^2} X X^T + \Sigma_p^{-1}\right)}_A (\mathbf{w} - \bar{\mathbf{w}})\right\} \\ &\boxed{\sim \mathcal{N}_{\mathbf{w}}(\bar{\mathbf{w}}, A^{-1})} && \text{After "completing the square"} \end{aligned}$$

where  $\bar{\mathbf{w}} \doteq \sigma^{-2} \underbrace{\left(\sigma^{-2} X X^T + \Sigma_p^{-1}\right)^{-1}}_{A^{-1} \in \mathbb{R}^{D \times D}} X \mathbf{y} \in \mathbb{R}^D$  **MAP estimation**

$$\boxed{A \doteq \left(\sigma^{-2} X X^T + \Sigma_p^{-1}\right) \in \mathbb{R}^{D \times D}}$$

# Bayesian Analysis of Linear Regression with Gaussian noise

We want to use  $P(\mathbf{w}|X, \mathbf{y}) = N_{\mathbf{w}}(\bar{\mathbf{w}}, A^{-1})$  posterior for predicting  $f$  in a test point  $\mathbf{x}_*$ .

$$f_* \doteq f(\mathbf{x}_*) \quad f(\mathbf{x}) = \mathbf{x}^T \mathbf{w} \in \mathbb{R}, \quad \mathbf{x}, \mathbf{w} \in \mathbb{R}^D$$
$$y = f(\mathbf{x}) + \epsilon = \mathbf{x}^T \mathbf{w} + \epsilon \in \mathbb{R}$$

$$P(\underbrace{f_*}_{\mathbf{x}_*^T \mathbf{w}} | \mathbf{x}_*, X, \mathbf{y}) = \int \underbrace{P(f_* | \mathbf{x}_*, \mathbf{w})}_{\delta(f_*, \mathbf{x}_*^T \mathbf{w})} \underbrace{P(\mathbf{w} | \mathbf{y}, X)}_{N_{\mathbf{w}}(\bar{\mathbf{w}}, A^{-1})} d\mathbf{w}$$
$$= \mathcal{N}_{f_*}(\mathbf{x}_*^T \bar{\mathbf{w}}, \mathbf{x}_*^T A^{-1} \mathbf{x}_*)$$

This posterior covariance matrix doesn't depend on the observations  $\mathbf{y}$ ,  
A strange property of Gaussian Processes  $\mathbf{y}^T = [y_1, \dots, y_n]$

# Projections of Inputs into Feature Space

The Bayesian linear regression suffers from  
limited expressiveness



To overcome the problem  $\Rightarrow$   
go to a feature space and do linear regression there

a., **explicit** features  $\phi(\mathbf{x}) = [x_1, x_1x_2^2, x_1 - x_2, \dots]^T$

b., **implicit** features (kernels)  $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2)$

# Explicit Features

$$\phi(\mathbf{x}) = [x_1, x_1x_2^2, x_1 - x_2, \dots]^T \in \mathbb{R}^N \quad \phi : \mathbb{R}^D \rightarrow \mathbb{R}^N$$

$$\phi(X) = \begin{bmatrix} \phi(\mathbf{x}_1) & \phi(\mathbf{x}_2) & \dots & \phi(\mathbf{x}_n) \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w} \in \mathbb{R}, \quad \phi(\mathbf{x}), \mathbf{w} \in \mathbb{R}^N$$

$$y = f(\mathbf{x}) + \epsilon = \phi(\mathbf{x})^T \mathbf{w} + \epsilon \in \mathbb{R}$$

## Linear regression in the feature space

Let us repeat all the previous calculations with  $\phi(\mathbf{x}) \in \mathbb{R}^N$  (instead of  $\mathbf{x}$ )

# Explicit Features

**Reminder:** This is what we had without feature maps:

$$P(\underbrace{f_*}_{\mathbf{x}_*^T \bar{\mathbf{w}}} | \mathbf{x}_*, X, \mathbf{y}) = \mathcal{N}_{f_*}(\mathbf{x}_*^T \bar{\mathbf{w}}, \mathbf{x}_*^T A^{-1} \mathbf{x}_*)$$

$$\text{where } \bar{\mathbf{w}} \doteq \sigma^{-2} \underbrace{\left( \sigma^{-2} X X^T + \Sigma_p^{-1} \right)^{-1}}_{A^{-1} \in \mathbb{R}^{D \times D}} X \mathbf{y} \in \mathbb{R}^D$$
$$A \doteq \left( \sigma^{-2} X X^T + \Sigma_p^{-1} \right) \in \mathbb{R}^{D \times D}$$

**The predictive distribution after feature map:**

$$P(\underbrace{f_*}_{\phi(\mathbf{x}_*)^T \bar{\mathbf{w}}} | \mathbf{x}_*, X, \mathbf{y}) = \mathcal{N}_{f_*} \left( \phi(\mathbf{x}_*)^T \bar{\mathbf{w}}, \phi(\mathbf{x}_*)^T A^{-1} \phi(\mathbf{x}_*) \right)$$

$$\text{where } \bar{\mathbf{w}} \doteq \sigma^{-2} \underbrace{\left( \sigma^{-2} \phi(X) \phi(X)^T + \Sigma_p^{-1} \right)^{-1}}_{A^{-1} \in \mathbb{R}^{N \times N}} \underbrace{\phi(X)}_{\in \mathbb{R}^{N \times n}} \underbrace{\mathbf{y}}_{\in \mathbb{R}^{n \times 1}} \in \mathbb{R}^N$$

$$A \doteq \left( \sigma^{-2} \phi(X) \phi(X)^T + \Sigma_p^{-1} \right) \in \mathbb{R}^{N \times N}$$



# Explicit Features

## Shorthands:

$$\begin{aligned}\phi_* &\doteq \phi(\mathbf{x}_*) \in \mathbb{R}^N & N &= \text{dim of feature space} \\ \phi &\doteq \phi(X) = \begin{bmatrix} \phi(\mathbf{x}_1) & \phi(\mathbf{x}_2) & \dots & \phi(\mathbf{x}_n) \end{bmatrix} \in \mathbb{R}^{N \times n} \\ A &\doteq (\sigma^{-2} \phi \phi^T + \Sigma_p^{-1}) \in \mathbb{R}^{N \times N} \\ \bar{\mathbf{w}} &\doteq \underbrace{\sigma^{-2} (\sigma^{-2} \phi \phi^T + \Sigma_p^{-1})^{-1}}_{A^{-1} \in \mathbb{R}^{N \times N}} \phi \mathbf{y} \in \mathbb{R}^N\end{aligned}$$

## The predictive distribution after feature map:

$$P(\underbrace{f_*}_{\phi_*^T \mathbf{w} \in \mathbb{R}} | \mathbf{x}_*, X, \mathbf{y}) = \mathcal{N}_{f_*}(\phi_*^T \bar{\mathbf{w}}, \phi_*^T A^{-1} \phi_*)$$

# Explicit Features

**The predictive distribution after feature map:**

$$\begin{aligned}
 P(\underbrace{f_*}_{\phi_*^T \mathbf{w} \in \mathbb{R}} | \mathbf{x}_*, X, \mathbf{y}) &= \mathcal{N}_{f_*} \left( \underbrace{\phi_*^T \bar{\mathbf{w}}}_{\mathbb{R}}, \underbrace{\phi_*^T A^{-1} \phi_*}_{\mathbb{R}^{N \times N}} \right) \quad (*) \\
 &= \mathcal{N}_{f_*} \left( \underbrace{\sigma^{-2} \phi_*^T [\sigma^{-2} \phi \phi^T + \Sigma_p^{-1}]^{-1} \phi \mathbf{y}}_{\mathbb{R}}, \underbrace{\phi_*^T [\sigma^{-2} \phi \phi^T + \Sigma_p^{-1}]^{-1} \phi_*}_{\mathbb{R}^{N \times N}} \right)
 \end{aligned}$$

A problem with (\*) is that it needs an  $N \times N$  matrix inversion...

Let  $K \doteq \phi^T \Sigma_p \phi \in \mathbb{R}^{n \times n}$

**Theorem:**

(\*) can be rewritten:  $P(f_* | \mathbf{x}_*, X, \mathbf{y}) =$

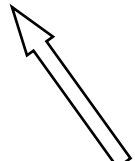
$$\boxed{
 \underbrace{\mathcal{N}_{f_*}}_{\mathbb{R}^{1 \times n}} \left( \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{\mathbf{y}}_{\mathbb{R}^{n \times 1}}, \underbrace{(\phi_*^T \Sigma_p \phi_*)}_{\mathbb{R}^{1 \times 1}} - \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{(\phi^T \Sigma_p \phi_*)}_{\mathbb{R}^{n \times 1}} \right)$$

# Proofs

- **Mean expression.** We need:

$$\sigma^{-2} \phi_*^T \underbrace{[\sigma^{-2} \phi \phi^T + \Sigma_p^{-1}]^{-1}}_{A^{-1}} \phi y = (\phi_*^T \underbrace{\Sigma_p \phi}_{\sigma^{-2} A^{-1} \phi}) (K + \sigma^2 \mathbf{I}_n)^{-1} y$$

**Lemma:**

$$\sigma^{-2} \phi (K + \sigma^2 \mathbf{I}_n) = \sigma^{-2} \phi (\phi^T \Sigma_p \phi + \sigma^2 \mathbf{I}_n) = A \Sigma_p \phi$$


- **Variance expression.** We need:

$$\phi_*^T [\sigma^{-2} \phi \phi^T + \Sigma_p^{-1}]^{-1} \phi_* = (\phi_*^T \Sigma_p \phi_*) - (\phi_*^T \Sigma_p \phi) (K + \sigma^2 \mathbf{I}_n)^{-1} (\phi^T \Sigma_p \phi_*)$$

**Matrix inversion Lemma:**

$$(\underbrace{U}_{\phi} \underbrace{W}_{\sigma^{-2}} \underbrace{V^T}_{\phi^T} + \underbrace{Z}_{\Sigma_p^{-1}})^{-1} = Z^{-1} - Z^{-1} U (W^{-1} + \underbrace{V^T Z^{-1} U}_K)^{-1} V^T Z^{-1}$$

# From Explicit to Implicit Features

$$P(f_*|\mathbf{x}_*, X, \mathbf{y}) =$$

$$\mathcal{N}_{f_*} \left( \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{\mathbf{y}}_{\mathbb{R}^{n \times 1}}, \underbrace{(\phi_*^T \Sigma_p \phi_*)}_{\mathbb{R}^{1 \times 1}} - \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{(\phi^T \Sigma_p \phi_*)}_{\mathbb{R}^{n \times 1}} \right)$$

**We only have to work with  $n \times n$  matrices, and not  $N \times N$**

# From Explicit to Implicit Features

$$P(f_*|\mathbf{x}_*, X, \mathbf{y}) =$$

$$\mathcal{N}_{f_*} \left( \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1} \mathbf{y}}_{\mathbb{R}^{n \times 1}}, \underbrace{(\phi_*^T \Sigma_p \phi_*)}_{\mathbb{R}^{1 \times 1}} - \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{(\phi^T \Sigma_p \phi_*)}_{\mathbb{R}^{n \times 1}} \right)$$

**The feature space always appears in the form of:**

$$(\phi_*^T \Sigma_p \phi_*), (\phi_*^T \Sigma_p \phi), (\phi^T \Sigma_p \phi), (\in \mathbb{R}^{n \times n} \text{ matrices})$$

$$\text{Let } k(\mathbf{x}, \tilde{\mathbf{x}}) \doteq \phi(\mathbf{x})^T \Sigma_p \phi(\tilde{\mathbf{x}})$$

**No need to know the explicit N dimensional features.**

**Their inner product is enough.**

**Lemma:**

$k(\mathbf{x}, \tilde{\mathbf{x}})$  is an inner product in the feature space:  $\psi(\mathbf{x}) \doteq \Sigma_p^{1/2} \phi(\mathbf{x})$  62

# GP pseudo code

## Inputs:

$$X = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times D}, \text{ } n \text{ training inputs}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n, \text{ } n \text{ training targets}$$

$k(\cdot, \cdot) : \mathbb{R}^{D \times D} \rightarrow \mathbb{R}$  covariance function (kernel)

$\mathbf{x}_* \in \mathbb{R}^D$  test input

$\sigma^2 > 0$  noise level on the observations

$$[y(\mathbf{x}) = f(\mathbf{x}) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma^2)]$$

# GP pseudo code (continued)

$$P(f_* | \mathbf{x}_*, X, \mathbf{y}) =$$

$$\mathcal{N}_{f_*} \left( \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1} \mathbf{y}}_{\mathbb{R}^{n \times n} \mathbb{R}^{n \times 1}} \underbrace{(\phi_*^T \Sigma_p \phi_*)}_{\mathbb{R}^{1 \times 1}} - \underbrace{(\phi_*^T \Sigma_p \phi)}_{\mathbb{R}^{1 \times n}} \underbrace{(K + \sigma^2 \mathbf{I}_n)^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{(\phi^T \Sigma_p \phi_*)}_{\mathbb{R}^{n \times 1}} \right)$$

1.,  $K \in \mathbb{R}^{n \times n}$  Gram matrix.  $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$

$$k(\mathbf{x}_*) = k_* = k(X, \mathbf{x}_*) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_*) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}_*) \end{bmatrix} \in \mathbb{R}^n$$

2.,  $\alpha = (K + \sigma^2 \mathbf{I}_n)^{-1} \mathbf{y}$

3.,  $\bar{f}_* = k_*^T \alpha \in \mathbb{R}$  (Posterior mean at  $x_*$ )

$$4., \text{cov}(f_*) = \underbrace{k(\mathbf{x}_*, \mathbf{x}_*)}_{\mathbb{R}} - \underbrace{k_*^T}_{\mathbb{R}^{1 \times n}} \underbrace{[K + \sigma^2 I_n]^{-1}}_{\mathbb{R}^{n \times n}} \underbrace{k_*}_{\mathbb{R}^n} \in \mathbb{R}$$

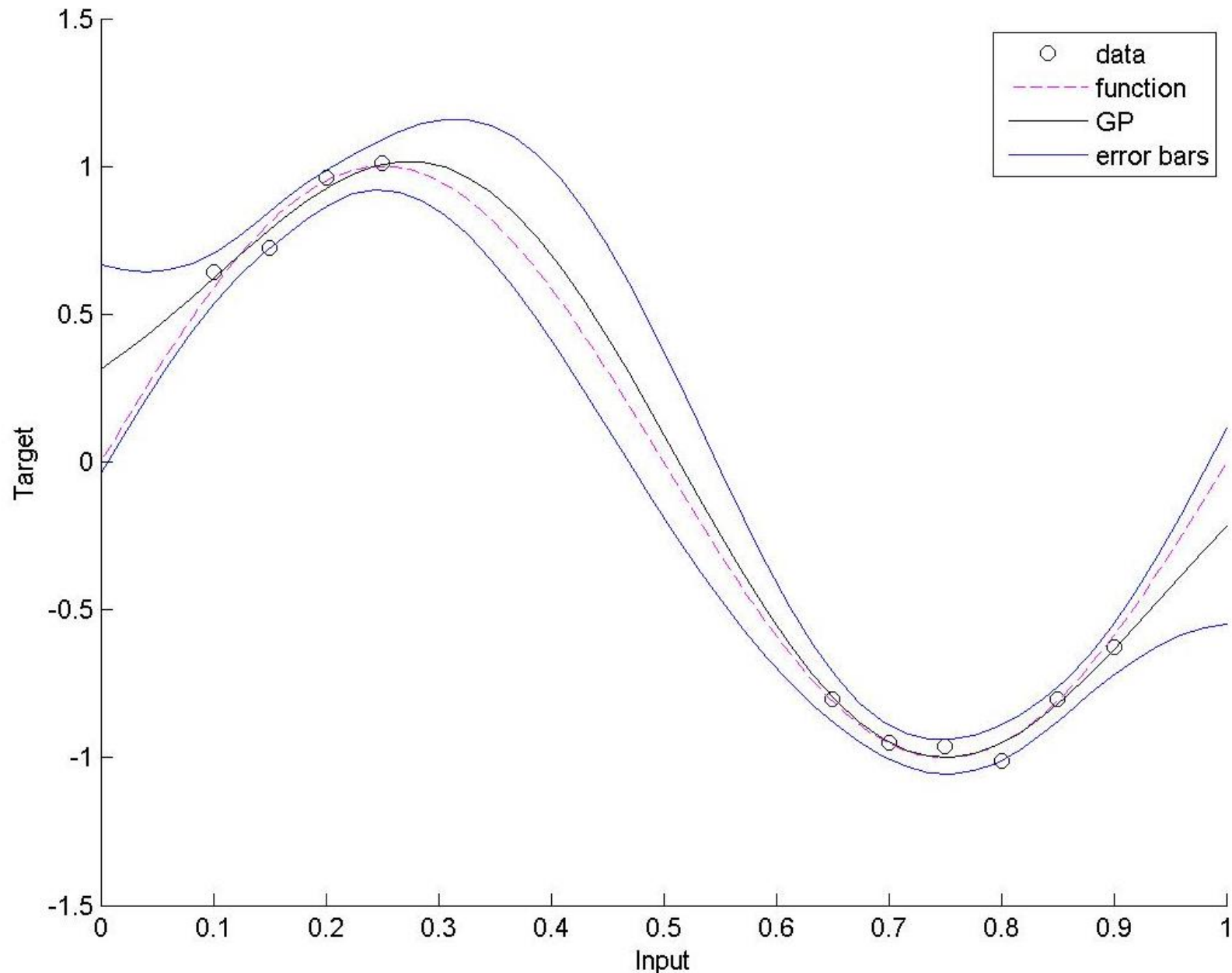
(Posterior covariance at  $x_*$ )

**Outputs:**  $\bar{f}_*, \text{cov}(f_*)$

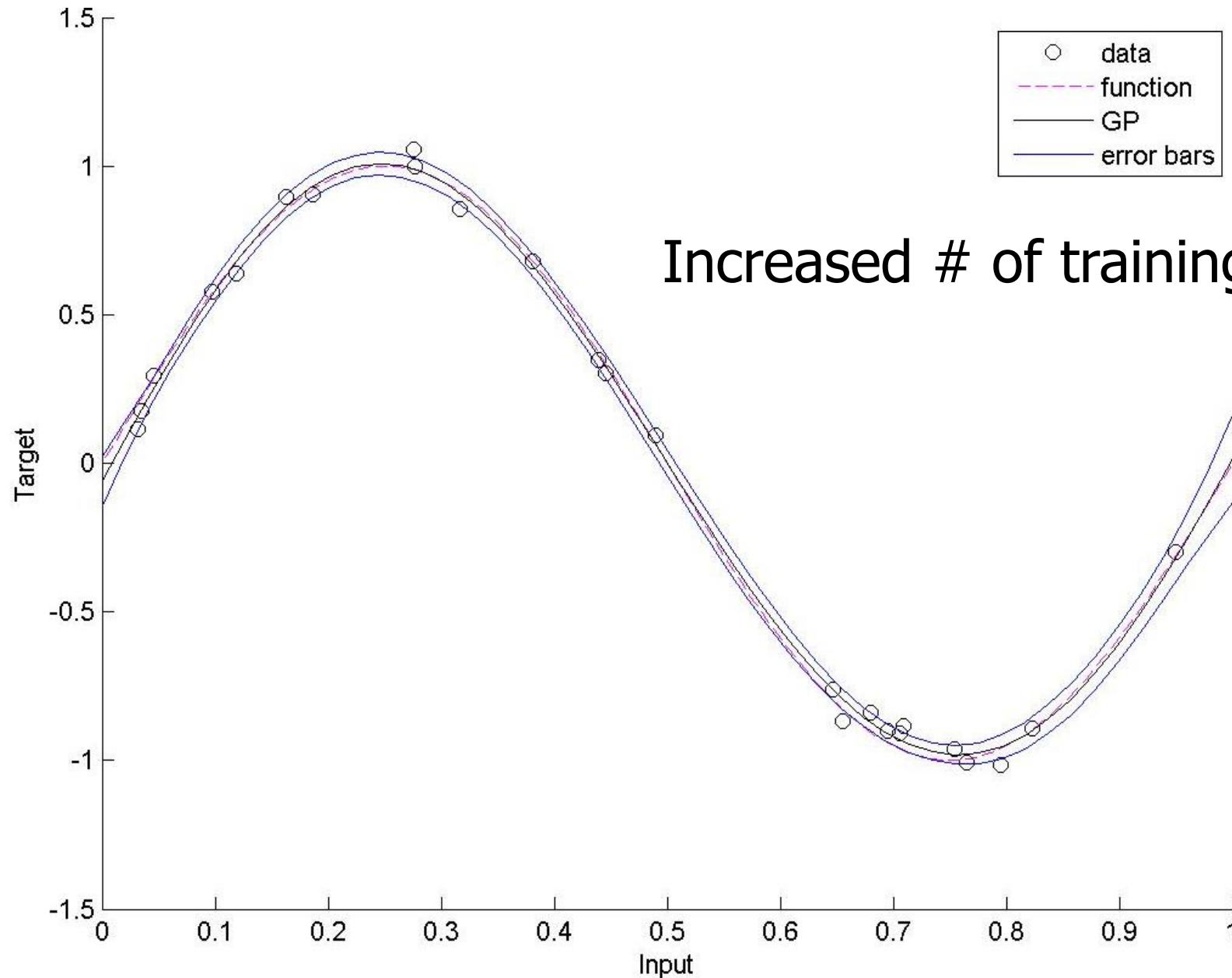
# Results



# Results using Netlab , Sin function

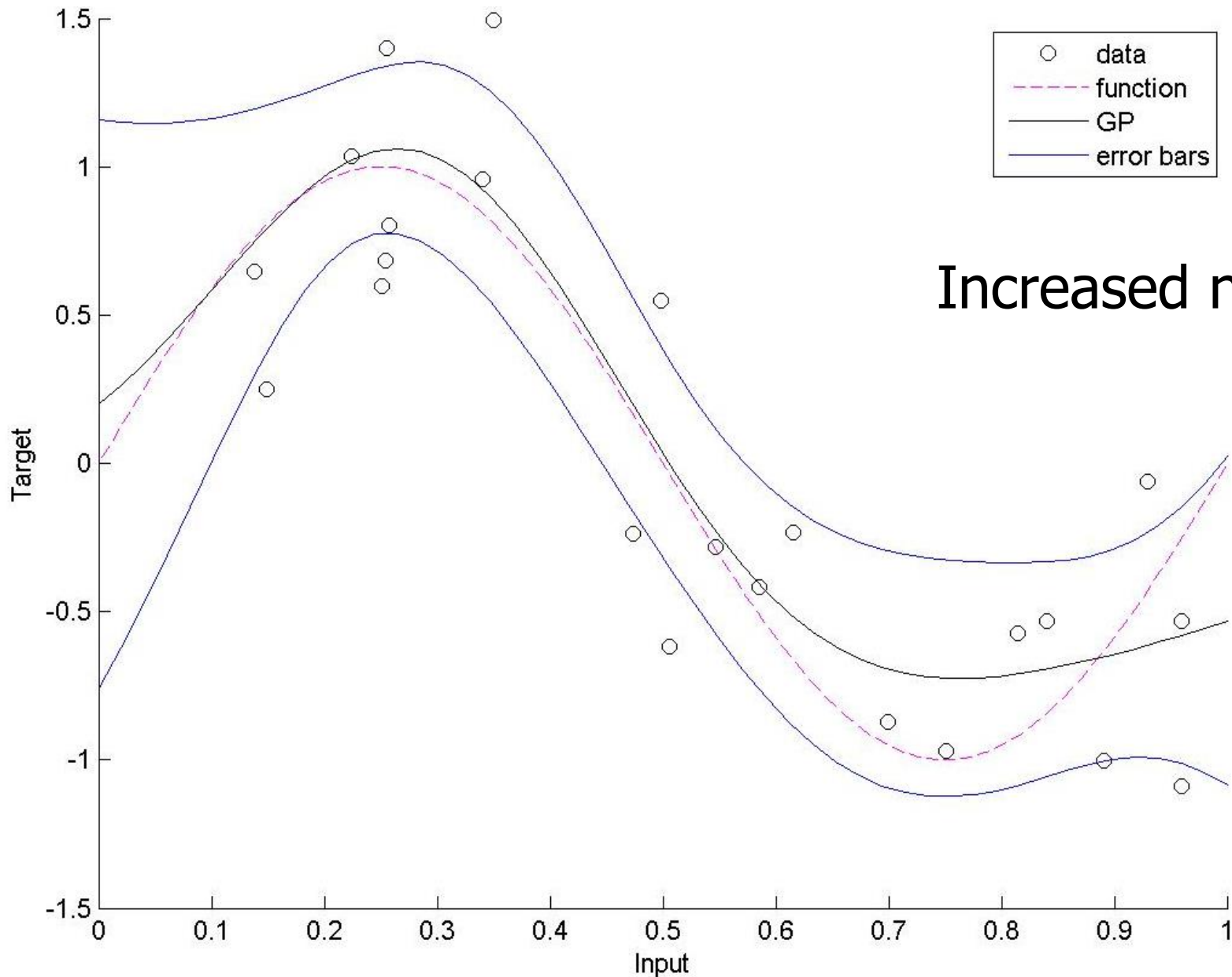


# Results using Netlab, Sin function



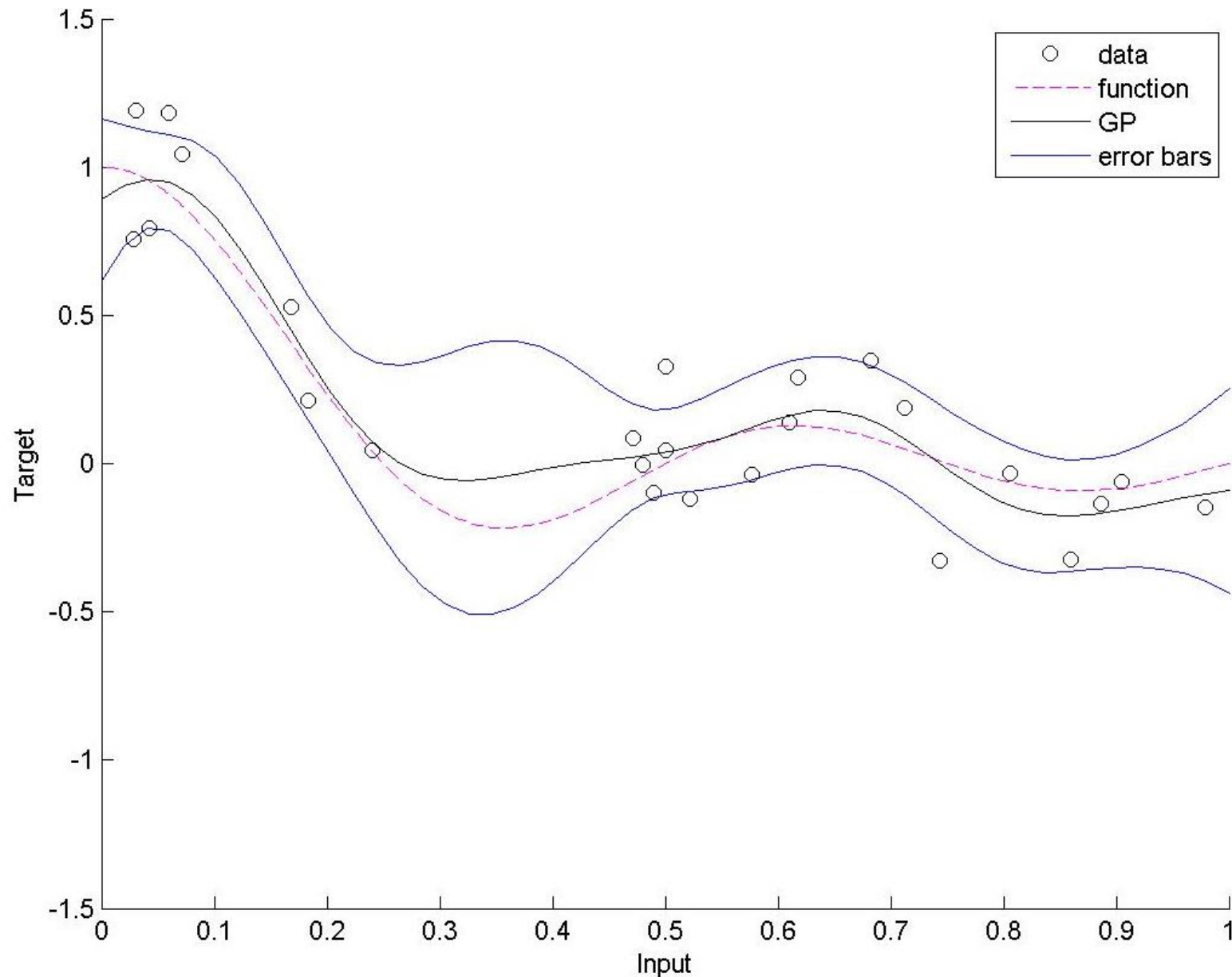
Increased # of training points

# Results using Netlab, Sin function

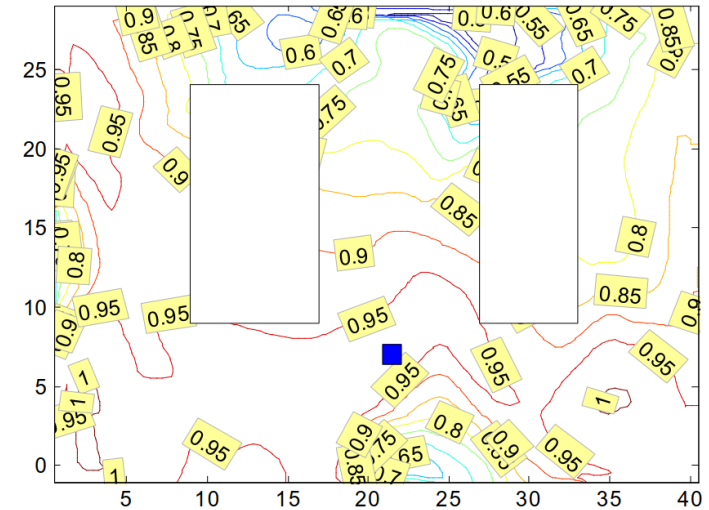
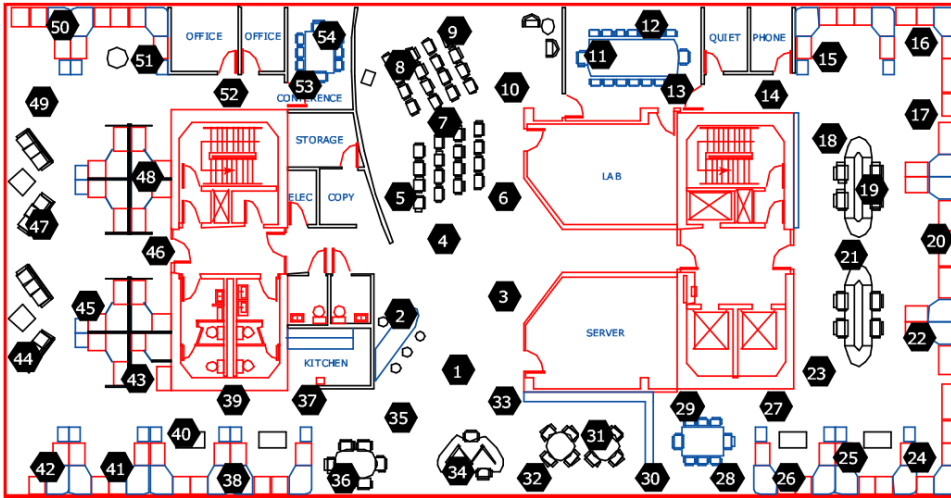


Increased noise

# Results using Netlab, Sinc function

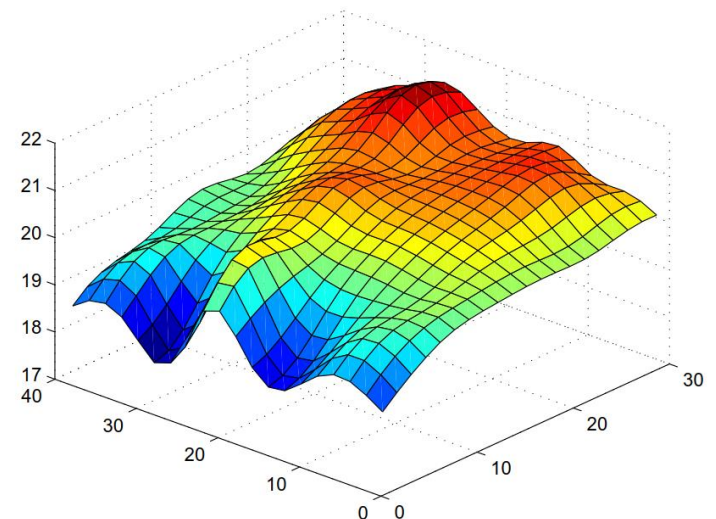


# Applications: Sensor placement



## Temperature modeling with GP

**Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies.** A. Krause, A. Singh, and C. Guestrin, Journal of Machine Learning Research (2008)



# Applications: Sensor placement

$\mathcal{A}$ : sensors are placed in these locations

$\mathcal{V}$ : all possible sensor locations

## Entropy criterion:

$$\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A} \subset \mathcal{V}: |\mathcal{A}|=k} H(\mathcal{X}_{\mathcal{V} \setminus \mathcal{A}} \mid \mathcal{X}_{\mathcal{A}}).$$

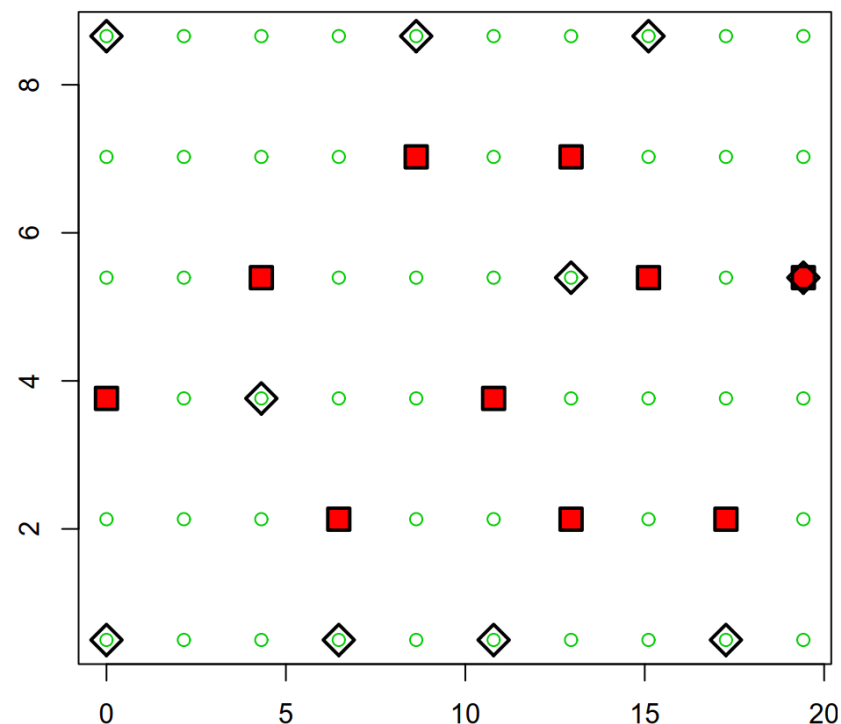
$$= \operatorname{argmax}_{\mathcal{A} \subset \mathcal{V}: |\mathcal{A}|=k} H(\mathcal{X}_{\mathcal{A}})$$

$$H(\mathcal{X}_{\mathcal{V} \setminus \mathcal{A}} \mid \mathcal{X}_{\mathcal{A}}) = H(\mathcal{X}_{\mathcal{V}}) - H(\mathcal{X}_{\mathcal{A}})$$

## Mutual information criterion:

$$\mathcal{A}^* = \operatorname{argmin}_{\mathcal{A} \subset \mathcal{V}: |\mathcal{A}|=k} I(\mathcal{X}_{\mathcal{A}}; \mathcal{X}_{\mathcal{V} \setminus \mathcal{A}})$$

$$\underbrace{H(\mathcal{X}_{\mathcal{V} \setminus \mathcal{A}}) - H(\mathcal{X}_{\mathcal{V} \setminus \mathcal{A}} \mid \mathcal{X}_{\mathcal{A}})}$$



An example of placements chosen using entropy and mutual information criteria on temperature data. Diamonds indicate the positions chosen using entropy; squares the positions chosen using MI.

# What You Should Know

- ❑ **Properties of Multivariate Gaussian distribution**
- ❑ **Gaussian process = Bayesian Ridge Regression**
- ❑ **GP Algorithm**
- ❑ **GP application in active learning**

Thanks for the Attention! 😊