

# Introduction to Gaussian Processes

Filipe P. de Farias, IC  
[filipepfarias@fisica.ufc.br](mailto:filipepfarias@fisica.ufc.br)

September 16, 2019

## 1 Linear Regression

- 1.1 Defining models
- 1.2 Optimizing the parameters
- 1.3 An uncertainty perspective

## 2 Bayesian Linear Regression

- 2.1 Bayesian statistics
- 2.2 Some Gaussian algebra
- 2.3 Change of Space

## 3 Gaussian Processes

- 3.1 Recap
- 3.2 Gaussian processes

# Linear Regression

# Defining models

## An initial curve fitting problem

- If we have a set of points in a space that comes from observations of an experiment and we want to predict other points, this could be done with **curve fitting**.
- So we could define some strategy to find our model.

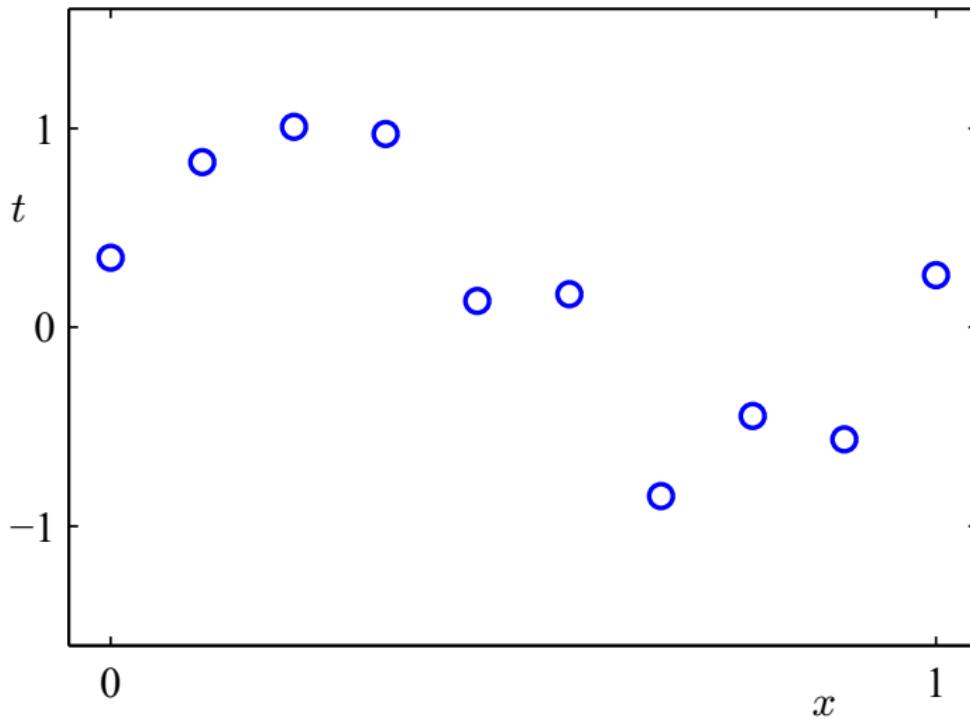
### Strategy

- 1 Purpose a **model**, e.g. functions like exponential, polynomial and others.
- 2 Train our model with the training data set, finding the **unknown parameters** or **weights**.

# Defining models

## An initial curve fitting problem

- Let's take the points below generated from the function  $y(x) = \sin(2\pi x)$  with addition of Gaussian noise with zero mean and 0.2 of standard deviation.



- We can express the curve with a polynomial, being the **model**

$$y(x, \mathbf{w}) = w_0x^0 + w_1x^1 + w_2x^2 + \dots + w_{M-1}x^{M-1} = \sum_{j=1}^{M-1} w_j x^j$$

- In general, we could write this **weighted sum** with any other function. In other words, we can put this in terms of  $\phi_n(x) = x^n$ , where  $\phi$  could be other **basis function**.
- e.g. we could have different  $y(x)$  for different basis functions, or **features**.

$$\begin{aligned} y(x, \mathbf{w}) &= w_0\phi_0(x) + w_1\phi_1(x) + w_2\phi_2(x) + \dots + w_{M-1}\phi_{M-1}(x) \\ &= w_0 \exp \left\{ -\frac{(x - \mu_0)^2}{2\sigma^2} \right\} + w_1 \exp \left\{ -\frac{(x - \mu_1)^2}{2\sigma^2} \right\} + \\ &\quad \dots + w_{M-1} \exp \left\{ -\frac{(x - \mu_{M-1})^2}{2\sigma^2} \right\} \\ &= w_0 \sin(0 \cdot x) + w_1 \cos(1 \cdot x) + \\ &\quad \dots + w_{M_2} \sin((M-2) \cdot x) + w_{M-1} \cos((M-1) \cdot x) \end{aligned}$$

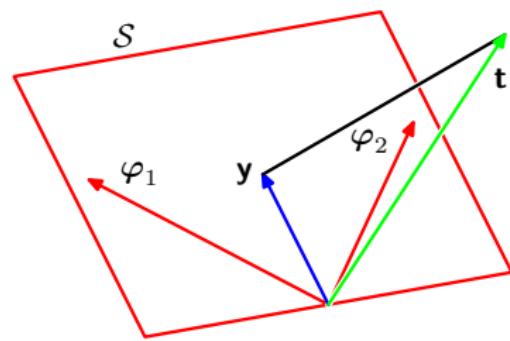
# Defining models

A non-linear model linear in parameters

- For simplicity, we'll carry this notation along.

$$\begin{aligned}y(x, \mathbf{w}) &= w_0\phi_0(x) + w_1\phi_1(x) + \dots \\&\quad + w_{M-1}\phi_{M-1}(x) \\&= \sum_{j=1}^{M-1} w_j\phi_j(x)\end{aligned}$$

- We'll evaluate  $\phi$  for all  $x$ , and then project it in the  $w$  vector space, the **feature-space**, then our model could be formed by **non-linear** functions. But, remaining **linear on parameters**.

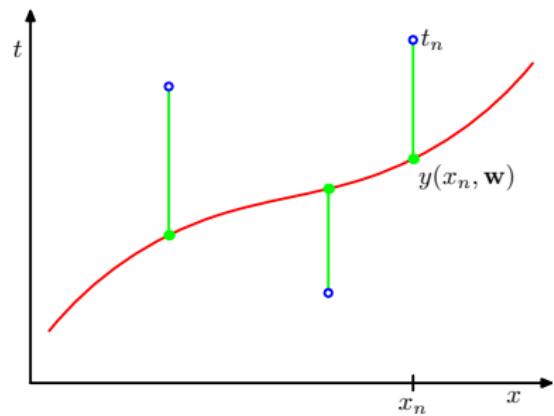


# Optimizing the parameters

## The model parameters

- The chosen model will give us some curve that is needed to adjust such that we'll **minimize its distance** to the **targets**  $t$ .
- This approach lead us to use the **least squares** to estimate the weights and minimize the **error**  $E$ .

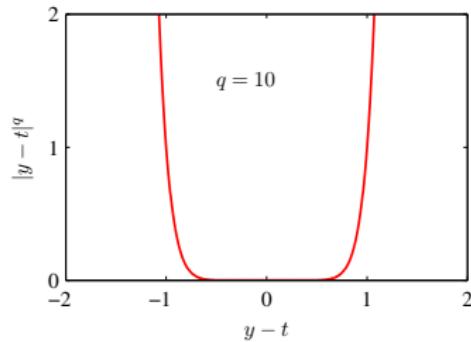
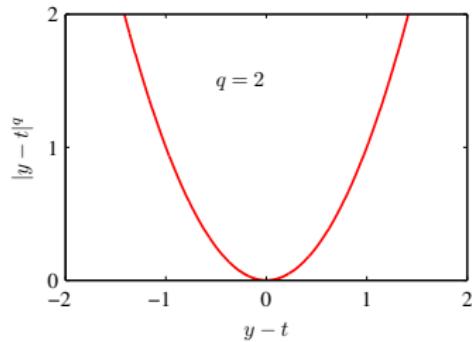
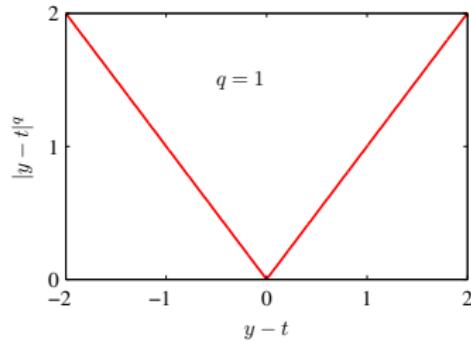
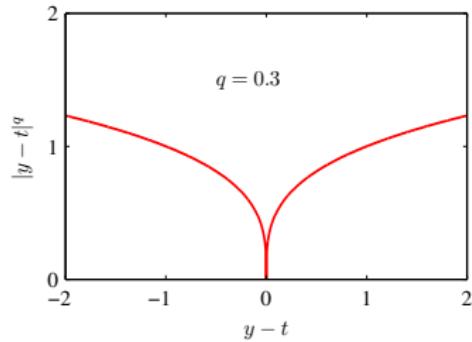
$$E(\mathbf{w}) \triangleq \frac{1}{2} \sum_{n=1}^N \{y_n - t_n\}^2$$



# Optimizing the parameters

## The model parameters

Why choose a quadratic norm distance?



### Why choose a quadratic norm distance?<sup>1</sup>

- The first row figures could be used for the derivations, taking care with some **non-continuous derivatives**.
- We'll use the **quadratic norm** because its the minor integer  $q$  differentiable, and then the error measures  $E$  between the model  $y(x, \mathbf{w})$  and the targets  $t$  will be euclidean.
- More, increasing the value of  $q$ , the smallests than 1 and bigger than 0 errors between the model and the targets that become irrelevant for  $E$ .

---

<sup>1</sup>See Appendix ?

# Optimizing the parameters

## Matrix form

- Remembering that

$$y(x, \mathbf{w}) = w_0\phi_0(x) + w_1\phi_1(x) + w_2\phi_2(x) + \dots + w_{M-1}\phi_{M-1}(x)$$

- We'll evaluate for all  $x_i$  values, and then put  $y_n(x_i, \mathbf{w})$  in the matrix form and get

$$y_n = [\phi_0(x_n) \quad \phi_1(x_n) \quad \dots \quad \phi_{M-1}(x_n)] [w_0 \quad w_1 \quad \dots \quad w_{M-1}]^\top$$

- And then

$$\underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \dots & \phi_{M-1}(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_{M-1}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_{N-1}) & \phi_1(x_{N-1}) & \dots & \phi_{M-1}(x_{N-1}) \end{bmatrix}}_{\Phi} \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N-1} \end{bmatrix}}_{\mathbf{w}}$$

where  $\Phi$  is the **design matrix**.

- This represents the system  $\mathbf{y} = \Phi\mathbf{w}$ .

# Optimizing the parameters

## The cost function and minimization problem

- If  $E(\mathbf{w}) = \frac{1}{2} (\mathbf{y} - \mathbf{t})^\top (\mathbf{y} - \mathbf{t})$  where  $\mathbf{t} = [t_1 \quad t_2 \quad \dots \quad t_n]^\top$
- Then we'll have

$$\begin{aligned} E(\mathbf{w}) &= \frac{1}{2} \left( \mathbf{y}^\top \mathbf{y} - \mathbf{t}^\top \mathbf{y} - \mathbf{y}^\top \mathbf{t} + \mathbf{t}^\top \mathbf{t} \right) \\ &= \frac{1}{2} \left( (\Phi \mathbf{w})^\top (\Phi \mathbf{w}) - \mathbf{t}^\top (\Phi \mathbf{w}) - (\Phi \mathbf{w})^\top \mathbf{t} + \mathbf{t}^\top \mathbf{t} \right) \\ &= \frac{1}{2} \left( \mathbf{w}^\top \Phi^\top \Phi \mathbf{w} - 2\mathbf{t}^\top \Phi \mathbf{w} + \mathbf{t}^\top \mathbf{t} \right) \end{aligned}$$

- In sequence, we'll try to minimize it in terms of the weights ( $\mathbf{w}$ ) by

$$\begin{aligned} 0 &= \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = \frac{1}{2} \left( 2\mathbf{w}^\top \Phi^\top \Phi - 2\mathbf{t}^\top \Phi + 0 \right) \\ \mathbf{w}^\top &= \mathbf{t}^\top \Phi \left( \Phi^\top \Phi \right)^{-1} \\ \mathbf{w}^* &= \left( \Phi^\top \Phi \right)^{-1} \Phi^\top \mathbf{t} \end{aligned}$$

- Here, we've obtained the weights  $\mathbf{w}^*$  with the **best fit** of the curve.
- We could say that the model **learned** the parameters.

### Why the prediction is so distant from the deterministic curve?

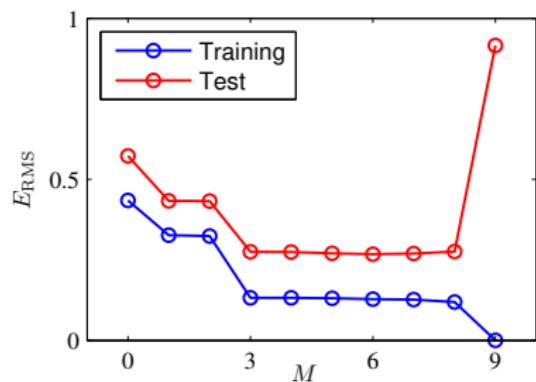
- A visible effect of the **increase of the complexity** of the model, is the increase of the **number of features**  $M$ .
- It's easy to see that our model start's to differ from the  $y$  and starts to interpolate the noise. We call this of **over-fitting**.
- This phenomenon illustrate a method of always search for the **best estimation of the parameters**.

# Optimizing the parameters

## Training and testing

Could be over-fitting a problem?

- We could **train** our model, it means evaluate  $\mathbf{w}^*$ , for only a part of our dataset.
- If the model be a good one, the error must be small when its **testing**, i.e. the error must be small when we evaluate all dataset with the  $\mathbf{w}^*$  of the trained part.
- But this in general does not occur and the **error increases**.



### How to control the over-fitting?

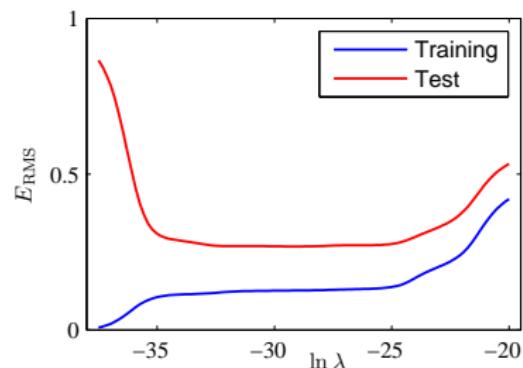
- With the increase of the model complexity, the value of  $\mathbf{w}^*$  increases too.
- A solution could be add a **penalty term** as the norm of the weights increases.
- To control the over-fitting, we try to **regularize** the weights by adding a penalty term  $\lambda$  to error function, by this we force the coefficients to not reach high values.

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - \mathbf{t})^\top(\mathbf{y} - \mathbf{t}) + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}$$
$$\Rightarrow \mathbf{w}_{\text{reg}}^* = (\Phi^\top\Phi + \lambda\mathbf{I})^{-1}\Phi^\top\mathbf{t}$$

### How to control the over-fitting?

- With the increase of the model complexity, the value of  $\mathbf{w}^*$  increases too.
- A solution could be add a **penalty term** as the norm of the weights increases.
- To control the over-fitting, we try to **regularize** the weights by adding a penalty term  $\lambda$  to error function, by this we force the coefficients to not reach high values.

$$\tilde{E}(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - \mathbf{t})^\top (\mathbf{y} - \mathbf{t}) + \frac{\lambda}{2} \mathbf{w}^\top \mathbf{w}$$
$$\Rightarrow \mathbf{w}_{\text{reg}}^* = (\Phi^\top \Phi + \lambda \mathbf{I})^{-1} \Phi^\top \mathbf{t}$$

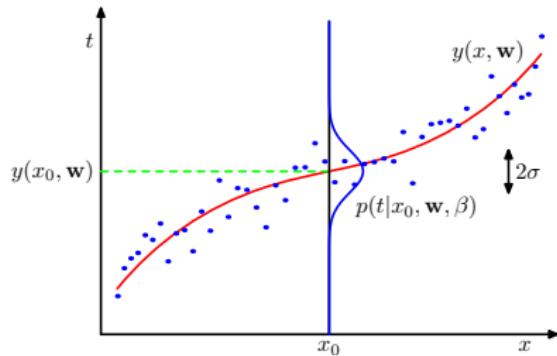


# An uncertainty perspective

## Looking back the regression

What if we assume not knowing the data exactly?

- Having an **uncertainty** in the measured value, we could represent it with a **probability distribution**.
- Now, each **target** could be expressed as a **random variable**.
- Its **mean** is given by  $y(x, \mathbf{w})$ , and the **variance** by  $1/\sigma^2 = \beta$ .
- $\beta$  is known as **precision parameter** too.
- For this case, we'll consider the distribution being **UniOrangeGaussian**.



- Being the random variables independent and identically distributed, we can say that our **joint probability** is given by

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^N p(t_n|x_n, \mathbf{w}, \beta)$$

known as **likelihood function**.

- Our goal is, given the **parameters**  $\mathbf{w}$ , maximize the **probability** of the **targets**.
- Before, consider a property of the probability distributions

$$\int_{-\infty}^{\infty} p(x)dx = 1 \text{ and } p(x) \geq 0$$

- Then, to avoid computational singularity and obtain a monotonically increasing function, we apply

$$\ln(p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)) = \sum_{n=1}^N \ln(p(t_n|x_n, \mathbf{w}, \beta))$$

# An uncertainty perspective

## Back to the cost function

- From the **joint probability** of the Gaussians distributions we have

$$\begin{aligned}\ln(p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)) &= \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right) \\ &= \sum_{n=1}^N -\frac{1}{2} \ln(2\pi) + \sum_{n=1}^N \frac{1}{2} \ln \beta - \sum_{n=1}^N \frac{\beta}{2} (t_n - y(x_n, \mathbf{w}))^2\end{aligned}$$

- If we make

$$\frac{\partial}{\partial \mathbf{w}} \ln(p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)) = 0$$

we'll obtain the **cost function** obtained before in the linear regression, then our assumptions are well grounded.

- With the maximization, we'll obtain the weights  $\mathbf{w}$  that **maximize** the log probability of the targets, given the parameters.
- This is called **maximum likelihood**, since we are looking for the **parameters** distribution that are more probable to had been generated the data.
- This is a initial step towards to a **Bayesian** approach.

# Bayesian Linear Regression

What if we assume not knowing the data exactly?

- The principle of the Bayesian statistics is express our **degree of belief** in an event.
- This belief could be based on some **prior** knowledge about the event or personal beliefs.
- This differs from frequentist statistics, where the probability is based on the number of trials.
- Suppose a die to be thrown once

### Frequentist

There's empiric evidence that similar dice thrown in past produce similar outcomes with the same frequency.

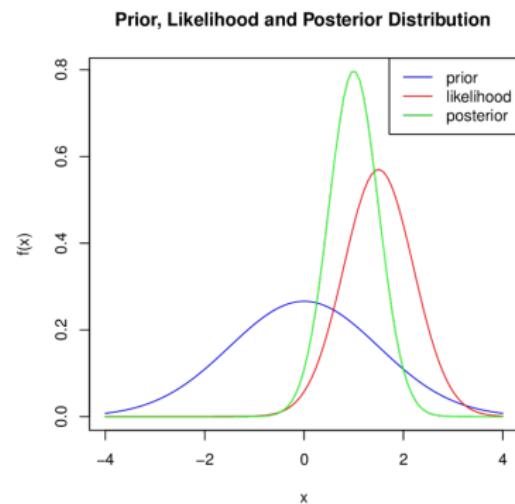


Figure: Aksu 2018

What if we assume not knowing the data exactly?

- The principle of the Bayesian statistics is express our **degree of belief** in an event.
- This belief could be based on some **prior** knowledge about the event or personal beliefs.
- This differs from frequentist statistics, where the probability is based on the number of trials.
- Suppose a die to be thrown once

## Bayesian

The last argument is right, but the **belief of the observer** it's important to the statistics.

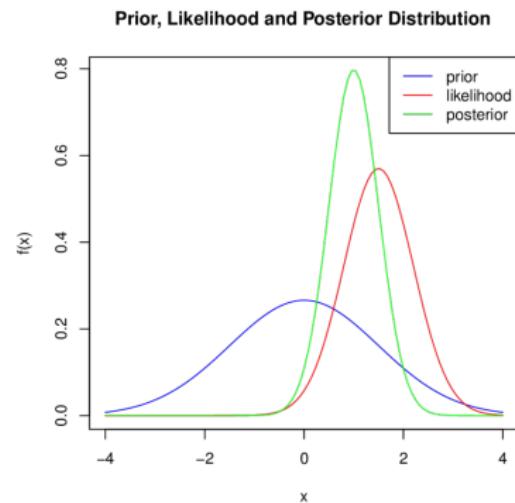


Figure: Aksu 2018

- The main idea of the Bayesian approach is put some **uncertainty** over the parameters and make **inferences**, i.e. obtain some statistics in light over the data.
- This principle is elucidated by the Bayes' Rule

$$\overbrace{p(\mathbf{w}|\mathbf{t})}^{\text{posterior}} = \frac{p(\mathbf{t}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{t})} = \frac{\overbrace{p(\mathbf{t}|\mathbf{w})}^{\text{likelihood}} \overbrace{p(\mathbf{w})}^{\text{prior}}}{\underbrace{\int p(\mathbf{t}|\mathbf{w})p(\mathbf{w})d\mathbf{w}}_{\text{marginal distribution}}}$$

where we assume some uncertainty over the parameters, i.e. a probability density function  $p(\mathbf{w})$ .

- We'll use the knowledge about the data with the **likelihood function** and some previous knowledge, or **prior**, that we have about the parameters to obtain the knowledge considering these two, or **posterior**.
- This is called **Bayesian Inference**.

- We can introduce the **maximum *a posteriori*** (MAP) as the direct estimator for the Bayes' Rule.
- The approach is similar to what was done by maximizing the likelihood function, but now maximizing the posterior distribution of the parameters given the data.
- We consider the **marginal distribution**  $p(\mathbf{t})$  being a constant in the parameters

$$p(\mathbf{w}|\mathbf{x}, \mathbf{t}) \propto p(\mathbf{t}|\mathbf{w}, \mathbf{x}) p(\mathbf{w})$$

- Here, we'll consider our prior knowledge about the parameters being

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1} \mathbf{I})$$

- We obtain by the derivative w.r.t.  $\mathbf{w}$  of the negative log that

$$\sum_{n=1}^N \frac{\beta}{2} (t_n - y(x_n, \mathbf{w}))^2 + \frac{\alpha}{2} \mathbf{w}^\top \mathbf{w} + \text{const.}$$

what is similar to the regularized linear regression considering  $\lambda = \alpha/\beta$ .

### Aren't we ignoring possible solutions?

- The main idea in the Bayesian approach is that our knowledge is in the **statistics** and not in a singular value.
- With MAP we just consider the **most probable** value of a full distribution of possible values.
- In the next we'll obtain the statistics of the distributions involved in Bayes' Rule, including the posterior.
- But before, we need some tools...

### Partitioned Gaussians

Be  $\mathbf{x}$  a  $n$ -dimensional vector with a Gaussian distribution  $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ , then the partitioned will be

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix}, \quad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix}.$$

Preserved the symmetry  $\boldsymbol{\Sigma}^\top = \boldsymbol{\Sigma}$ , we say the covariance matrix is positive definite. And be the multivariate Gaussian

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2}} \frac{1}{(\det \boldsymbol{\Sigma})^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

We define too, just for convenience of work, the precision matrix  $\boldsymbol{\Lambda}$  by

$$\boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{pmatrix} \equiv \boldsymbol{\Sigma}^{-1}$$

assuming all matrices have inverses.

### Closure under linear transformations and marginalization

Being  $\mathbf{x}_b$  conditioned on  $\mathbf{x}_a$  and Gaussian distributed as

$$p(\mathbf{x}_a) = \mathcal{N}(\mathbf{x}_a | \boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a), \quad p(\mathbf{x}_b | \mathbf{x}_a) = \mathcal{N}(\mathbf{x}_b | \mathbf{M}\mathbf{x}_a + \mathbf{d}, \boldsymbol{\Sigma}_{b|a})$$

$\mathbf{M}$  a constant matrix and  $\mathbf{d}$  a constant vector, both with the appropriate dimensions. Then conditional distribution  $p(\mathbf{x}_a | \mathbf{x}_b)$  is given by

$$p(\mathbf{x}_a | \mathbf{x}_b) = \mathcal{N}(\mathbf{x}_a | \boldsymbol{\mu}_{a|b}, \boldsymbol{\Sigma}_{a|b})$$

with

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\Sigma}_{a|b} \left( \mathbf{M}^\top \boldsymbol{\Sigma}_{b|a}^{-1} (\mathbf{x}_b - \mathbf{d}) + \boldsymbol{\Sigma}_a^{-1} \boldsymbol{\mu}_a \right), \quad \boldsymbol{\Sigma}_{a|b} = \left( \boldsymbol{\Sigma}_a^{-1} + \mathbf{M}^\top \boldsymbol{\Sigma}_{b|a}^{-1} \mathbf{M} \right)^{-1}.$$

The marginal density of  $\mathbf{x}_b$  is given by

$$p(\mathbf{x}_b) = \mathcal{N}(\mathbf{x}_b | \boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b)$$

with

$$\boldsymbol{\mu}_b = \mathbf{M}\boldsymbol{\mu}_a + \mathbf{d}, \quad \boldsymbol{\Sigma}_b = \boldsymbol{\Sigma}_{b|a} + \mathbf{M}\boldsymbol{\Sigma}_a\mathbf{M}^\top.$$

### Substituting...

Being  $\mathbf{M}$  the design matrix  $\Phi$  and  $\mathbf{d}$  a zero vector, we have the posterior distribution

$$p(\mathbf{w}|\mathbf{t}, \alpha, \beta) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{\mathbf{w}|\mathbf{t}}, \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}})$$

being

$$\boldsymbol{\mu}_{\mathbf{w}|\mathbf{t}} = \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} (\beta \Phi^\top \mathbf{t} + \boldsymbol{\Sigma}_{\mathbf{w}}^{-1} \boldsymbol{\mu}_{\mathbf{w}}), \quad \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} = (\boldsymbol{\Sigma}_{\mathbf{w}}^{-1} + \beta \Phi^\top \Phi)^{-1}.$$

Assuming the prior

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1} \mathbf{I})$$

we have

$$\boldsymbol{\mu}_{\mathbf{w}|\mathbf{t}} = \beta \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} \Phi^\top \mathbf{t}, \quad \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} = (\alpha^{-1} \mathbf{I} + \beta \Phi^\top \Phi)^{-1}.$$

# Bayesian Linear Regression

## An example

### From Bayesian inference

- In the most of the cases, we are more interested in making predictions of  $\mathbf{t}$  than in the parameters  $\mathbf{w}$  in the space of the parameters, or **weight-space**, for the new values of  $\mathbf{x}$ . We'll define from the Bayes' Rule a **predictive distribution**

$$p(\mathbf{t}_* | \mathbf{x}_*, \mathbf{x}, \mathbf{t}) = \int p(\mathbf{t}_* | \mathbf{x}_*, \mathbf{w}) p(\mathbf{w} | \mathbf{x}, \mathbf{t}) d\mathbf{w}$$

- And turn to the **feature-space**  $f$

$$\begin{aligned} p(f_* | \mathbf{x}_*, \Phi, \mathbf{t}) &= \int p(f_* | \boldsymbol{\phi}_*^\top, \mathbf{w}) p(\mathbf{w} | \Phi, \mathbf{t}) d\mathbf{w} \\ &= \mathcal{N}\left(\beta \boldsymbol{\phi}_*^\top \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} \Phi \mathbf{t}, \boldsymbol{\phi}_*^\top \boldsymbol{\Sigma}_{\mathbf{w}|\mathbf{t}} \boldsymbol{\phi}(\mathbf{x}_*)\right) \end{aligned}$$

where  $f_* \triangleq f(\mathbf{x}_*)$ ,  $\boldsymbol{\phi}_* = \boldsymbol{\phi}(\mathbf{x}_*)$  at  $\mathbf{x}_*$  and  $\Phi = \Phi(\mathbf{x})$  at  $\mathbf{x}$ .

# Bayesian Linear Regression

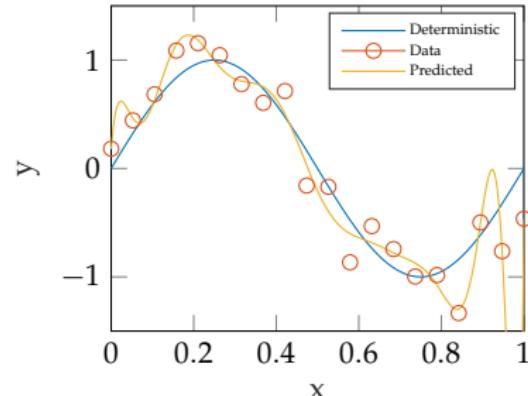
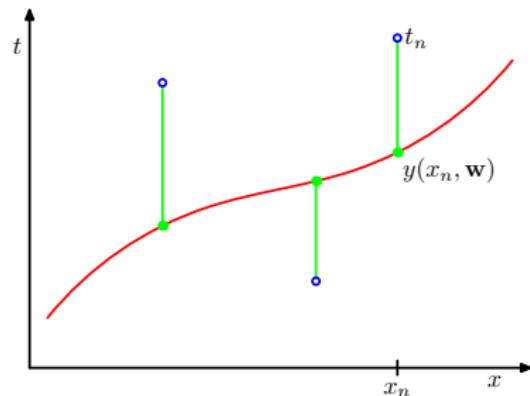
## Change of Space



# Gaussian Processes

### What was done until here?

- We assumed that our targets  $t$  were **i.i.d.** and given by  $t = y(\mathbf{x}) + \varepsilon$ , where  $\varepsilon \sim \mathcal{N}(0, \beta)$ .
- Our model is given by  $y(\mathbf{x}) = \Phi^\top \mathbf{w}$ , where  $\Phi$  is the **design matrix**, and this characterize our model as **linear in parameters**.
- The **design matrix** was defined as  $\phi_{i,j} = \phi_i(\mathbf{x}_j)$ .
- The **parameters** were given by  $\mathbf{w} = (\Phi^\top \Phi)^{-1} \Phi^\top \mathbf{t}$ .
- These **parameters** calculated at the minimum of the cost function are called **maximum likelihood**.



# Recap

## Bayesian Linear Regression

### What was done until here?

- We put an **uncertainty** over the targets  $t$  and the parameters  $\mathbf{w}$ .
- We assumed that targets being **distributed** as  $p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$ .
- By **Bayes' Rule** we obtained that  $p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta) \propto p(\mathbf{t}|\mathbf{w}, \mathbf{x}, \beta) p(\mathbf{w}|\alpha)$
- This allowed to make an **inference** to obtain a **prediction** of the parameters in the **weight-space**.

# Recap

A more clear way to see what is happening...

### What is kernel?

$$f_* | \mathbf{x}_*, \Phi, \mathbf{t} \sim \mathcal{N} \left( \boldsymbol{\phi}_*^\top \mathbf{S}_0 \Phi \left( K + \beta^{-2} I \right)^{-1} \mathbf{t}, \boldsymbol{\phi}_*^\top \mathbf{S}_0 \boldsymbol{\phi}_* - \boldsymbol{\phi}_*^\top \mathbf{S}_0 \Phi \left( K + \beta^{-2} I \right)^{-1} \Phi^\top \mathbf{S}_0 \boldsymbol{\phi}_* \right)$$

- We could observe the appearance of terms like  $\Phi^\top \mathbf{S}_0 \Phi$ ,  $\boldsymbol{\phi}_*^\top \mathbf{S}_0 \Phi$ , or  $\boldsymbol{\phi}_*^\top \mathbf{S}_0 \boldsymbol{\phi}_*$ .
- The common term between these operations is  $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{S}_0 \boldsymbol{\phi}(\mathbf{x}')$
- Then we define  $k(\cdot, \cdot)$  as **kernel function**
- This technique is particularly valuable in situations where it is more convenient to compute the kernel than the design matrix vectors themselves.

- Previously we make the inference in the **feature-space** and then we find the function distribution.
- Now we'll make the inference directly on **function-space**.
- Let's define

### Definition

A **Gaussian process** is a collection of random variables which any finite number of them have a joint Gaussian distribution.

### Mean and covariance function

- As the Gaussian distribution, the  $\mathcal{GP}$  is characterized by its **mean function**  $m(\mathbf{x})$  and its **covariance function**  $k(\mathbf{x}, \mathbf{x}')$  of a real process  $f(\mathbf{x})$ .
- For a Gaussian processes

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

- We have

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$$



# Bibliography I

- Anil Aksu (Jan. 2018). *Decision Theory and Bayesian Analysis*.
- C.M. Bishop (2016). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer New York. ISBN: 9781493938438. URL:  
<https://books.google.com.br/books?id=kOXDtAEACAAJ>.
- P.J. Dhrymes (2013). *Mathematics for Econometrics*. SpringerLink : Bücher. Springer New York. ISBN: 9781461481454. URL:  
<https://books.google.com.br/books?id=HIK8BAAAQBAJ>.
- J. L. Doob (Sept. 1944). "The Elementary Gaussian Processes". In: *Ann. Math. Statist.* 15.3, pp. 229–282. DOI: 10.1214/aoms/1177731234. URL:  
<https://doi.org/10.1214/aoms/1177731234>.
- F.A. Graybill (2001). *Matrices with Applications in Statistics*. Duxbury Classic Series. Brooks/Cole. ISBN: 9780534401313. URL:  
<https://books.google.com.br/books?id=BV3CAAAACAAJ>.
- Philipp Hennig (Sept. 2013). *Animating Samples from Gaussian Distributions*. Technical Report 8. Spemannstraße, 72076 Tübingen, Germany: Max Planck Institute for Intelligent Systems.
- T.B. Schön (2011). *Manipulating the Multivariate Gaussian Density*. URL:  
<http://user.it.uu.se/~thosc112/pubpdf/schonl2011.pdf>.