Gaussian Processes

Dr. Francis Colas

28.10.2011

Linear Regression

Linear Regression:

- data points $(\mathbf{x}_n, t_n)$,
- set of basis functions $\boldsymbol{\phi}(\mathbf{x})$,
- model: $y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$,
- minimization of the SSE: $E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$
- regularization to avoid overfitting:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Finding best parameters $\mathbf{w}$.

Linear Regression

Linear Regression:

- data points $(\mathbf{x}_n, t_n)$,
- set of basis functions $\boldsymbol{\phi}(\mathbf{x})$,
- model: $y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$,
- minimization of the SSE: $E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$
- regularization to avoid overfitting:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Finding best parameters $\mathbf{w}$.

Linear Regression

Linear Regression:

- data points $(\mathbf{x}_n, t_n)$,
- set of basis functions $\phi(\mathbf{x})$,
- model: $y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \phi(\mathbf{x})$,
- minimization of the SSE: $E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$
- regularization to avoid overfitting:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Finding best parameters $\mathbf{w}$.

Linear Regression

Linear Regression:

- data points $(\mathbf{x}_n, t_n)$,
- set of basis functions $\phi(\mathbf{x})$,
- model: $y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \phi(\mathbf{x})$,
- minimization of the SSE: $E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2$
- regularization to avoid overfitting:

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Finding best parameters $\mathbf{w}$.

Probabilistic Regression

Equivalent probabilistic formalization:

- Gaussian noise on target values,
- maximum likelihood $\Leftrightarrow$ minimum of the sum of squared errors,
- Gaussian prior on $\|\mathbf{w}\|$,
- maximum a posteriori $\Leftrightarrow$ regularization.

Finding most probable parameters $\mathbf{w}$.

What about $y(\mathbf{x})$?

Probabilistic Regression

Equivalent probabilistic formalization:

- Gaussian noise on target values,
- maximum likelihood $\Leftrightarrow$ minimum of the sum of squared errors,
- Gaussian prior on $\|\mathbf{w}\|$,
- maximum a posteriori $\Leftrightarrow$ regularization.

Finding most probable parameters $\mathbf{w}$.

What about $y(\mathbf{x})$?

Prior

Let's look at $y(\mathbf{x})$:

$$\mathbf{y} = \mathbf{w}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}$$

with:

- $\mathbf{y} = (y(\mathbf{x}_1, \mathbf{w}), \cdots, y(\mathbf{x}_n, \mathbf{w}))$,
- $\mathbf{\Phi} = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \vdots & \phi_k(\mathbf{x}_n) & \vdots \\ \phi_1(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{pmatrix}$ (design matrix);

From a probabilistic point of view:

- $\mathbf{w}$ is a Gaussian distributed variable,
- $\mathbf{y}$ is a linear combination of $\mathbf{w}$,
- $\Rightarrow \mathbf{y}$ is a Gaussian distributed variable.

Prior

Let's look at $y(\mathbf{x})$:

$$\mathbf{y} = \mathbf{w}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}$$

with:

- $\mathbf{y} = (y(\mathbf{x}_1, \mathbf{w}), \cdots, y(\mathbf{x}_n, \mathbf{w}))$,
- $\mathbf{\Phi} = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \vdots & \phi_k(\mathbf{x}_n) & \vdots \\ \phi_1(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{pmatrix}$ (design matrix);

From a probabilistic point of view:

- $\mathbf{w}$ is a Gaussian distributed variable,
- $\mathbf{y}$ is a linear combination of $\mathbf{w}$,
- $\Rightarrow \mathbf{y}$ is a Gaussian distributed variable.

Distribution on $\mathbf{y}$

From:

- $\mathbf{y} = \mathbf{w}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}$,
- $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{w}}^2\mathbf{I})$;

We have:

- $E[\mathbf{y}] = E[\mathbf{w}]^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}} = \mathbf{0}$,
- $\mathrm{cov}[\mathbf{y}] = E[\mathbf{y}\mathbf{y}^{\mathrm{T}}] = \mathbf{\Phi}^{\mathrm{T}}E[\mathbf{w}^{\mathrm{T}}\mathbf{w}]\mathbf{\Phi} = \sigma_{\mathbf{w}}^2\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} = \mathbf{K}$

$\Leftrightarrow p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$

Joint Gaussian probability distribution for the evaluation of $y(\mathbf{x})$ in a finite set of points.

Distribution on $\mathbf{y}$

From:

- $\mathbf{y} = \mathbf{w}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}$,
- $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{w}}^2\mathbf{I})$;

We have:

- $E[\mathbf{y}] = E[\mathbf{w}]^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}} = \mathbf{0}$,
- $\mathrm{cov}[\mathbf{y}] = E[\mathbf{y}\mathbf{y}^{\mathrm{T}}] = \mathbf{\Phi}^{\mathrm{T}}E[\mathbf{w}^{\mathrm{T}}\mathbf{w}]\mathbf{\Phi} = \sigma_{\mathbf{w}}^2\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} = \mathbf{K}$

$\Leftrightarrow p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$

Joint Gaussian probability distribution for the evaluation of $y(\mathbf{x})$ in a finite set of points.

Distribution on $\mathbf{y}$

From:

- $\mathbf{y} = \mathbf{w}^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}}$,
- $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{w}}^2\mathbf{I})$;

We have:

- $E[\mathbf{y}] = E[\mathbf{w}]^{\mathrm{T}}\mathbf{\Phi}^{\mathrm{T}} = \mathbf{0}$,
- $\mathrm{cov}[\mathbf{y}] = E[\mathbf{y}\mathbf{y}^{\mathrm{T}}] = \mathbf{\Phi}^{\mathrm{T}}E[\mathbf{w}^{\mathrm{T}}\mathbf{w}]\mathbf{\Phi} = \sigma_{\mathbf{w}}^2\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} = \mathbf{K}$

$\Leftrightarrow p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$

Joint Gaussian probability distribution for the evaluation of $y(\mathbf{x})$ in a finite set of points.

Gram Matrix and Kernel Function

Gram matrix:

$$\mathbf{K} = \begin{pmatrix} \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_1)^T \boldsymbol{\phi}(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_1)^T \boldsymbol{\phi}(\mathbf{x}_N) \\ \vdots & \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_n)^T \boldsymbol{\phi}(\mathbf{x}_m) & \vdots \\ \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_N)^T \boldsymbol{\phi}(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_N)^T \boldsymbol{\phi}(\mathbf{x}_N) \end{pmatrix}$$

General term:

$$k_{n,m} = \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x}_n)^T \boldsymbol{\phi}(\mathbf{x}_m)$$

Kernel (function):

$$k(\mathbf{x}, \mathbf{x}') = \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}')$$

Gram Matrix and Kernel Function

Gram matrix:

$$\mathbf{K} = \begin{pmatrix} \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_N) \\ \vdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) & \vdots \\ \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_N) \end{pmatrix}$$

General term:

$$k_{n,m} = \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$$

Kernel (function):

$$k(\mathbf{x}, \mathbf{x}') = \sigma_{\mathbf{w}}^2 \phi(\mathbf{x})^T \phi(\mathbf{x}')$$

Gram Matrix and Kernel Function

Gram matrix:

$$\mathbf{K} = \begin{pmatrix} \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_N) \\ \vdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m) & \vdots \\ \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_1) & \cdots & \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_N) \end{pmatrix}$$

General term:

$$k_{n,m} = \sigma_{\mathbf{w}}^2 \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$$

Kernel (function):

$$k(\mathbf{x}, \mathbf{x}') = \sigma_{\mathbf{w}}^2 \phi(\mathbf{x})^T \phi(\mathbf{x}')$$

Definition

Definition: A Gaussian process is a *probability distribution over functions* $y(\mathbf{x})$ such as the vector $\mathbf{y}$ of $y(\mathbf{x})$ evaluated at an *arbitrary finite* set of points $\mathbf{x}_1, \cdots, \mathbf{x}_n$ is Gaussian distributed.

Specification:

- mean $E[y(\mathbf{x})]$ (most of the times, $\mathbf{0}$),
- covariance at any two values $E[y(\mathbf{x}_n)y(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m)$: *kernel*.

Kernel function instead of basis functions.

Definition

Definition: A Gaussian process is a *probability distribution over functions* $y(\mathbf{x})$ such as the vector $\mathbf{y}$ of $y(\mathbf{x})$ evaluated at an *arbitrary finite* set of points $\mathbf{x}_1, \cdots, \mathbf{x}_n$ is Gaussian distributed.

Specification:

- mean $E[y(\mathbf{x})]$ (most of the times, $\mathbf{0}$),
- covariance at any two values $E[y(\mathbf{x}_n)y(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m)$: *kernel*.

Kernel function instead of basis functions.

Definition

Definition: A Gaussian process is a *probability distribution over functions* $y(\mathbf{x})$ such as the vector $\mathbf{y}$ of $y(\mathbf{x})$ evaluated at an *arbitrary finite* set of points $\mathbf{x}_1, \cdots, \mathbf{x}_n$ is Gaussian distributed.

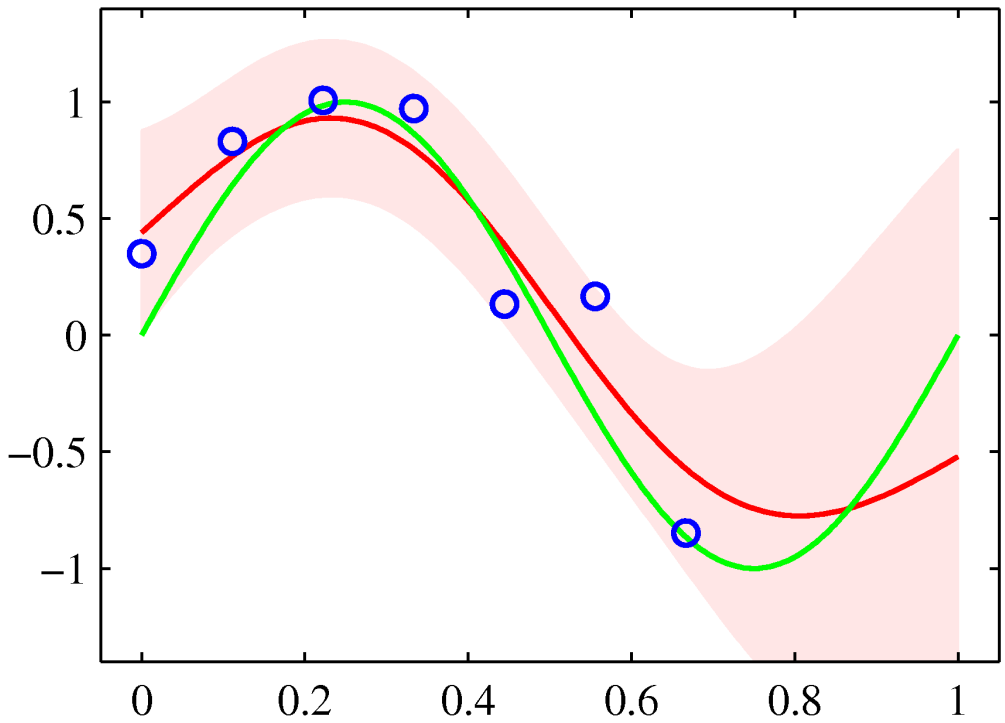
Specification:

- mean $E[y(\mathbf{x})]$ (most of the times, $\mathbf{0}$),
- covariance at any two values $E[y(\mathbf{x}_n)y(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m)$: *kernel*.

Kernel function instead of basis functions.

Example

Representation of a Gaussian process:



- green line: sinusoidal data source,
- blue circles: data points with Gaussian noise,
- red line: mean of the Gaussian process,
- shaded red area: 2σ confidence interval.

Kernel Trick

With basis functions:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_{\mathbf{w}}^2 \phi(\mathbf{x})^T \phi(\mathbf{x}')$$

$\Rightarrow$ kernel is a dot product of basis function; $\Rightarrow$ kernel must be positive, semi-definite, and symmetric;

Kernel trick:

- Mercer's theorem: positive, semi-definite, symmetric function corresponds to a dot product in some space;
- define kernel as any positive, semi-definite, symmetric function.

Kernel Trick

With basis functions:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_{\mathbf{w}}^2 \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}')$$

$\Rightarrow$ kernel is a dot product of basis function; $\Rightarrow$ kernel must be positive, semi-definite, and symmetric;

Kernel trick:

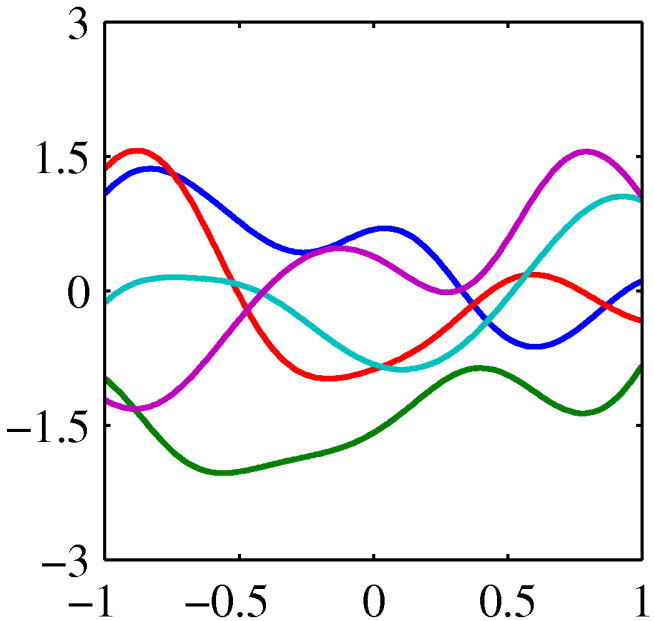
- Mercer's theorem: positive, semi-definite, symmetric function corresponds to a dot product in some space;
- define kernel as any positive, semi-definite, symmetric function.

Examples

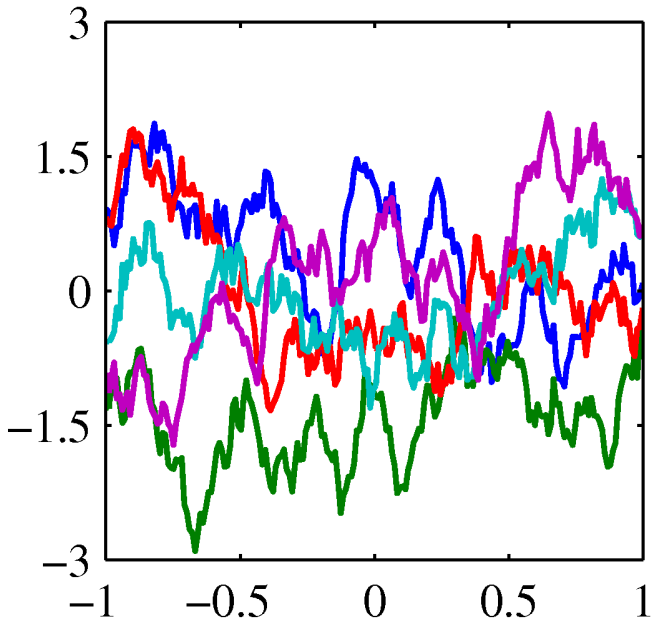
Covariance specified as a kernel:

- Gaussian kernel: $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|}{2\sigma^2}\right)$,
- exponential kernel: $k(x, x') = \exp\left(-\theta|x - x'|\right)$: *Ornstein-Uhlenbeck process* for Brownian motion.

Samples from Gaussian processes:



Gaussian kernel



Exponential kernel

Summary

Gaussian processes:

- probability distributions on functions,
- Gaussian distribution of samples,
- covariance defined by kernel,
- implicite feature space;

Example of two applications:

- regression,
- classification.

Back on Regression, Again

So far:

- regression expressed with respect to basis functions,
- focus on computing weight vector $\mathbf{w}$,
- with a Gaussian prior on $\mathbf{w}$; $y(\mathbf{x})$ is an example of Gaussian Process;

Now:

- come back to probabilistic formulation,
- get rid of $\mathbf{w}$ and $\phi(\mathbf{x})$,
- estimate t_{N+1} for new point $\mathbf{x}_{N+1}$.

Noisy Observations

As in aforementioned probabilistic formulation:

- added noise on target values,
- $t_n = y_n + \epsilon_n$,
- $y_n = y(\mathbf{x}_n)$ is nth component of $\mathbf{y}$;

Gaussian noise:

- $p(t_n \mid y_n) = \mathcal{N}(t_n | y_n, \frac{1}{\beta})$,
- $p(\mathbf{t} \mid \mathbf{y}) = \mathcal{N}(\mathbf{t} | \mathbf{y}, \frac{1}{\beta}\mathbf{I}_N)$ (with $\mathbf{I}_N$ the unit matrix of size $N \times N$),
- $p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | \mathbf{0}, \mathbf{K})$ (with $\mathbf{K}$ a Gram matrix based on a kernel function).

Posterior on $\mathbf{t}$.

We compute:

$$\begin{aligned} p(\mathbf{t}) &= \int p(\mathbf{t} \mid \mathbf{y}) p(\mathbf{y}) d\mathbf{y} \\ &= \int \mathcal{N}(\mathbf{t}|\mathbf{y}, \frac{1}{\beta}\mathbf{I}_N)\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) d\mathbf{y} \\ &= \int \mathcal{N}\left((\mathbf{y}, \mathbf{t})^T | (\mathbf{0}, \mathbf{0})^T, \begin{pmatrix} \mathbf{K} & \mathbf{K} \\ \mathbf{K} & \mathbf{K} + \frac{1}{\beta}\mathbf{I}_N \end{pmatrix}\right) d\mathbf{y} \\ &= \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C}) \end{aligned}$$

$$\mathbf{C} = \mathbf{K} + \frac{1}{\beta}\mathbf{I}_N = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) + \frac{1}{\beta} & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & k(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{\beta}\delta_{n,m} & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) + \frac{1}{\beta} \end{pmatrix}$$

The covariance just add: the Gaussian noises are independent.

Kernels

Kernel:

- $\mathbf{C} = \mathbf{K} + \frac{1}{\beta}\mathbf{I}$ must be symmetric positive definite (covariance matrix),
- $\mathbf{K}$ just need to be symmetric positive *semi-definite*;

Choice:

- depends on application,
- either built with feature functions,
- or defined directly,
- in general: for more similar points $\mathbf{x}_n$ and $\mathbf{x}_m$, $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ are more correlated.

Example

Kernel for regression on a line:

- $\phi(\mathbf{x}) = \begin{pmatrix} 1 \\ x \end{pmatrix}$,
- $k(\mathbf{x}, \mathbf{x}') = \frac{1}{\alpha}\phi(\mathbf{x})^t\phi(\mathbf{x}') = \frac{1}{\alpha}(1 + xx')$;

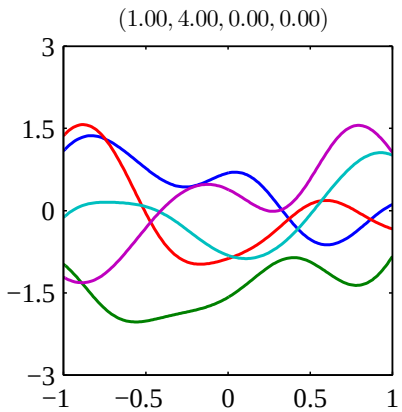
Common kernel for Gaussian process regression:

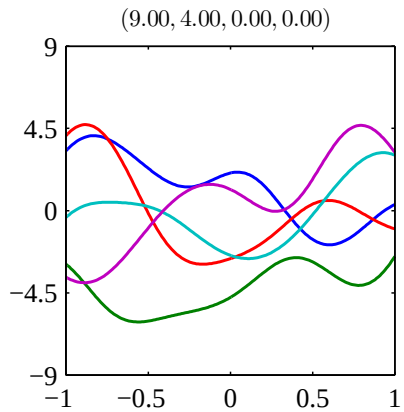
$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left\{-\frac{\theta_1}{2}\|\mathbf{x}_n - \mathbf{x}_m\|^2\right\} + \theta_2 + \theta_3 \mathbf{x}_n \mathbf{x}_m$$

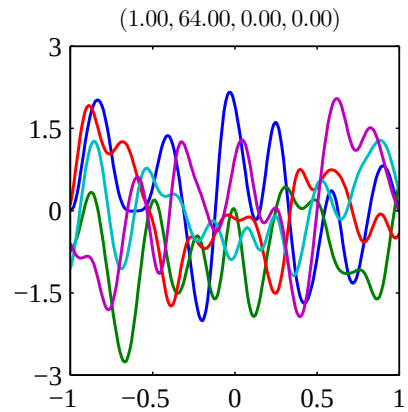
- θ_0: relevance of proximity,
- θ_1: proximity scale,
- θ_2: noise,
- θ_3: linear parametric form.

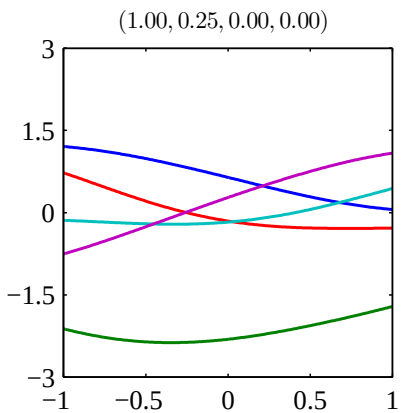
Samples from Gaussian Processes

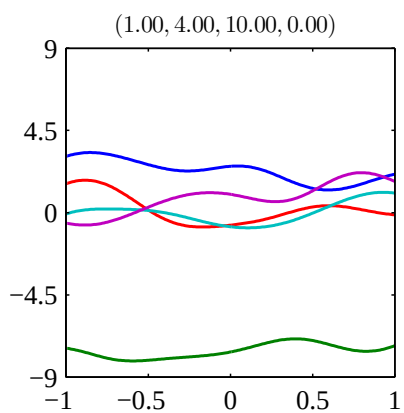
Using kernel $k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left\{-\frac{\theta_1}{2}\|\mathbf{x}_n - \mathbf{x}_m\|^2\right\} + \theta_2 + \theta_3 \mathbf{x}_n \mathbf{x}_m$:

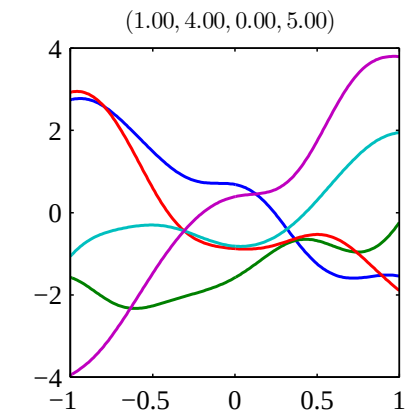








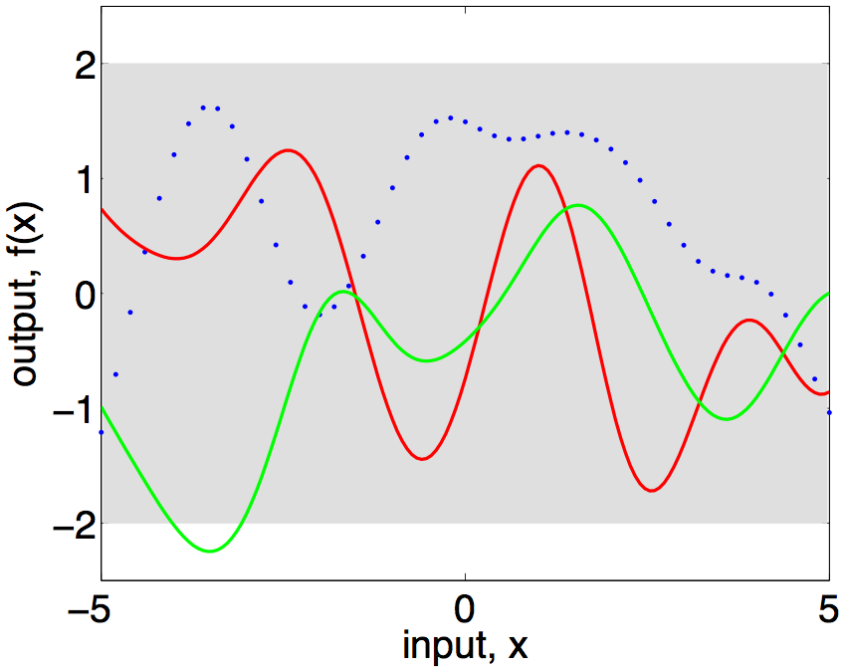




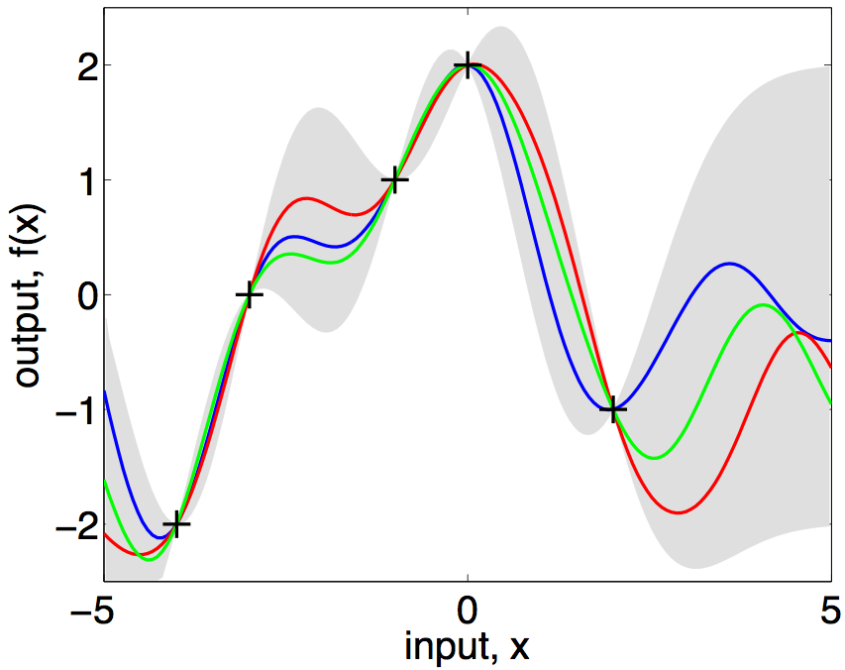
Regression: from Prior to Posterior

Data points constrain the function distribution:

Prior



Posterior



Gray area: 95% confidence interval.

From Rasmussen & Williams, Gaussian Processes for Machine Learning

`http://www.gaussianprocess.org/gpml/`

Prediction

More than a distribution on $\mathbf{t}$ we want to make predictions:

- new input vector $\mathbf{x}_{N+1}$,
- $\mathbf{t}_N = (t_1, \ldots, t_N)^T$,
- prediction of t_{N+1}:

$$p(t_{N+1} \mid \mathbf{t}_N) = \frac{p(\mathbf{t}_{N+1})}{p(\mathbf{t}_N)}$$

with:

$$p(\mathbf{t}_N) = \mathcal{N}(\mathbf{t}_N|\mathbf{0}, \mathbf{C}_N)$$

and

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

Covariance for prediction

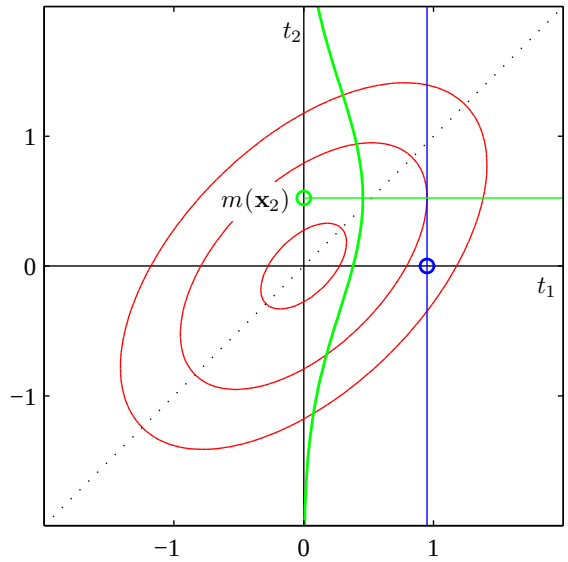
$$\mathbf{C}_{N+1} = \begin{pmatrix} & \vdots & \\ \mathbf{C}_N & & k(\mathbf{x}_n, \mathbf{x}_{N+1}) \\ & \vdots & \\ \cdots k(\mathbf{x}_n, \mathbf{x}_{N+1}) \cdots & & k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \frac{1}{\beta} \end{pmatrix} = \begin{pmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^T & c \end{pmatrix}$$

Then:

- $p(t_{N+1} \mid \mathbf{t}_N)$ is Gaussian with:
- mean:

$$\begin{aligned} m(\mathbf{x}_{N+1}) &= \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}_N \\ &= \sum_{n-1}^{N} a_n k(\mathbf{x}_n, \mathbf{x}_{N+1}) \end{aligned}$$

where a_n is the nth component of $\mathbf{C}_N^{-1}\mathbf{t}_N$,



Complexity

Regression with Gaussian Processes:

- $p(t_{N+1} \mid \mathbf{t}_N) = \mathcal{N}(t_{N+1} \mid \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}_N, c - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k})$
- inversion of a $N \times N$ positive semi-definite matrix,
- Cholesky decomposition: $A = LL^T$ with L lower triangular,
- complexity in $O(N^3)$;

Standard Regression:

- solve a linear system of size M,
- complexity in $O(M^3)$;

Useful for feature space of higher dimension than number of data points.

Summary

Regression with Gaussian Processes:

- function viewpoint instead of parameters,
- probability distribution over functions,
- data points constrain the distribution,
- definition of a kernel for the covariance,
- implicit feature space,
- complexity in terms of data points.

From Regression to Classification

Regression:

- mapping from $\mathbf{x}_n$ to real-valued t_n,
- Gaussian appropriate;

Classification:

- mapping from $\mathbf{x}_n$ to discrete class labels t_n,
- Gaussian inappropriate.

Classification using Gaussian Processes

Workaround:

- evaluate the probability $y(\mathbf{x}) \in (0, 1)$ of being part of t_n,
- introduce Gaussian process $a(\mathbf{x}) \in \boldsymbol{R}$,
- transform the output $y = \sigma(a) \in (0, 1)$.

Sigmoid:

- logistic:

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

- cumulative density of Gaussian distribution:

$$\boldsymbol{\Phi}(a) = \int_{-\infty}^{a} \mathcal{N}(x|0, 1)dx$$

Bernoulli Distribution (can be generalized):

$$p(t \mid a) = \sigma(a)^t(1 - \sigma(a))^{1-t}$$

Prediction (1/2)

Prediction:

- training set $(\mathbf{x}_n, t_n)$,
- new point $\mathbf{x}_{N+1}$,
- Gaussian Process prior over $\mathbf{a}_{N+1} = (a(\mathbf{x}_1), \ldots, a(\mathbf{x}_{N+1}))$:

$$p(\mathbf{a}_{N+1}) = \mathcal{N}(\mathbf{a}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

with $\mathbf{C}_{N+1}$ from $C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \nu\delta_{n,m}$

(We could assume that the classification is not noisy, but we have to have a symmetric *definite* positive covariance matrix, whereas kernels can be only semi-definite.)

Prediction (2/2)

For binary classification:

$$p(t_{N+1} = 1|\mathbf{t}_N)$$

which can be expressed as:

$$p(t_{N+1} = 1|\mathbf{t}_N) = \int p(t_{N+1} = 1 \mid a_{N+1}) p(a_{N+1} \mid \mathbf{t}_N) da_{N+1}$$

With:

- $p(t_{N+1} = 1 \mid |a_{N+1}) = \sigma(a_{N+1})$,
- $p(a_{N+1} \mid \mathbf{t}_N)$ not even Gaussian;

$\Rightarrow$ analytically intractable.

Approximations

Two approximations:

- $\int \sigma(a)\mathcal{N}(a|\mu, \sigma^2)da$: $\sigma(\kappa(\sigma^2)\mu)$,
- $p(a_{N+1} \mid \mathbf{t}_N)$: Gaussian;

Usually:

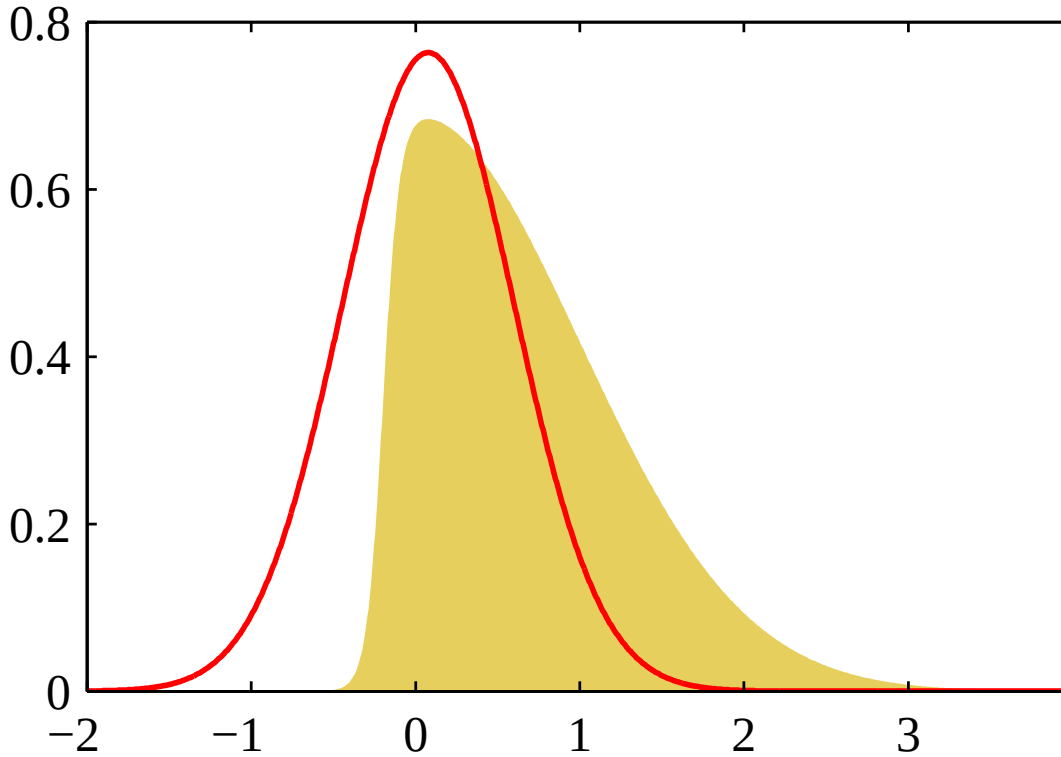
- central limit theorem,
- noise;

$\Rightarrow$ Laplace approximation.

Laplace Approximation (1/2)

Idea:

- center on the mode,
- logarithm,
- order-2 Taylor expansion;



Laplace Approximation (2/2)

Approximate $p(z)$ with $q(z) = \mathcal{N}(z|\mu, \sigma^2)$:

- mode z_0 such that $\frac{dp}{dz}(z_0) = 0$,
- $\ln q(z) = -\frac{1}{2}\ln 2\pi\sigma^2 - \frac{1}{2}\frac{(z-\mu)^2}{\sigma^2}$,
- order-2 Taylor expansion of $p(z)$ in z_0:

$$\begin{aligned} \ln p(z) &\approx \ln p(z_0) - \frac{1}{2}A(z-\mu)^2 \\ p(z) &\approx \sqrt{\frac{A}{2\pi}}\exp\left\{-\frac{A}{2}(z-z_0)^2\right\} \end{aligned}$$

- $\Rightarrow q(z) = \mathcal{N}(z|z_0, A^{-1})$.

Summary

Gaussian processes:

- probability distribution over functions,
- vectors of values are Gaussian,
- use kernel for the covariance,
- implicit feature space;

Regression:

- mean for the regression value,
- covariance for uncertainty information;

Classification:

- need a remapping,
- approximate inference.