FACULTY OF
COMPUTER SCIENCE
IAȘI

‘Alexandru Ioan Cuza’
University of Iași,
Romania

Using the EM algorithm and Gaussian Processes to solve a gene expression problem

Sebastian Ciobanu

Liviu Ciortuz

Outline

- **Theory**
  - The EM algorithm
  - Gaussian processes
- **Application**
  - Regression and clustering. Gene expression

The EM algorithm

The EM algorithm (1)

- Actually, it is an algorithmic schema
- E = Expectation
- M = Maximization
- Why use it?
  - Function maximization
- Which function?
  - Log-likelihood function of (observed) data
- When to use it?
  - When one works with latent data

Likelihood function

- $f(h) = P(D|h)$
  - D = (observed) data
  - h = hypothesis/model
    - E.g.: parameters of a probability distribution:
      - $D = \{1,2,3\}$
      - $h_0 = \{mu = 0, sigma = 1\}$; Normal distribution
      - $f(h_0) = p(D|h_0) = p(1,2,3|mu = 0, sigma = 1)$

$$= p(1|mu = 0, sigma = 1)\ p(2|mu = 0, sigma = 1)\ p(3|mu = 0, sigma = 1)$$

$$= \ldots$$

$$= 5.78987e\text{-}05$$

Log-likelihood function

- $f(h) = \ln P(D|h)$
  - D = (observed) data
  - h = hypothesis/model
- Why?
  - $P(D|h)$ will be a product of numbers in [0,1]
  - We work with derivatives
  - The computer uses approximations

Data - example

- Complete Notation: Y = (X, Z)
  - Observed
  - Unobserved || Latent

Price	Type of product
2.1	1
3	2
4	3
5	1
3.1	2
2	3
7	3

Notation: X

Notation: Z

… log-likelihood of observed data…

How to maximize

… log-likelihood of observed data… ?

Ex:

$$\sum_{i=1}^{n} \ln \left(\sum_{j=1}^{k} p_{X|Z,h}(x_i|j,h) p_{Z|h}(j|h) \right)$$

1. Standard numerical methods (e.g.: the gradient ascent method)
2. **The EM algorithm**

The EM algorithm (2)

- Initialization: ? => [W,] h
- While(...)
  - W = eStep(X,h)
  - h = mStep(X,W)
  - (this implicitly increases P(X|h_var))

The EM algorithm (2)

- Initialization: ? => [W,] h
- While(...)
  - W = E[g(Z)|X,h]
  - h = ln P(X,Z|h_var)
  - (this implicitly increases P(X|h_var))

The EM algorithm (2)

- Initialization: ? => [W,] h
- While(...)
  - W = E[g(Z)|X,h]
  - h = $E_{P(g(Z)|X,h)}$ [ln P(X,Z|h_var)]
  - (this implicitly increases P(X|h_var))

The EM algorithm (2)

- Initialization: ? => [W,] h
- While(...)
  - $W = E[g(Z)|X,h]$
  - $h = \text{argmax}_{h\_var}\ E_{P(g(Z)|X,h)}\ [\ln P(X,Z|h\_var)]$
  - (this implicitly increases $P(X|h\_var)$)

The EM algorithm (2)

- Initialization: ? => [W,] h
- While(...)
  - W = E[g(Z)|X,h]
  - h = $\text{argmax}_{h\_var} \overbrace{E_{P(g(Z)|X,h)}\,[\ln P(X,Z|h\_var)]}^{\overset{\mathbf{Q(h\_var|h)}}{\text{not.}}}$
  - (this implicitly increases P(X|h_var))

Gaussian processes

Most of the ideas and notations are taken from Andrew Ng, Stanford University, ML course notes – Gaussian Processes

Stochastic (Random) process

- Indexed collection of random variables

$$f : \mathcal{X} \to \mathcal{F}(\Omega, \mathbb{R}) \text{ - stochastic process}$$
$$\mathcal{X} \text{ - index set}$$
$$\mathcal{F}(\Omega, \mathbb{R}) \text{ - set of random variables}$$

Stochastic (Random) process

- It induces a probability distribution over:
  - Functions: if $\mathcal{X}$ is finite
  - Function approximations: if $\mathcal{X}$ is infinite
    - We approximate $\mathcal{X}$ by a finite number of indexes
- So, we have:

$$\mathcal{X} = \{x_1, \ldots, x_n\} \text{ or } \mathcal{X} \approx \{x_1, \ldots, x_n\}$$

Stochastic (Random) process

$$f \overset{\text{not.}}{=} \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix} \qquad v \overset{\text{not.}}{=} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

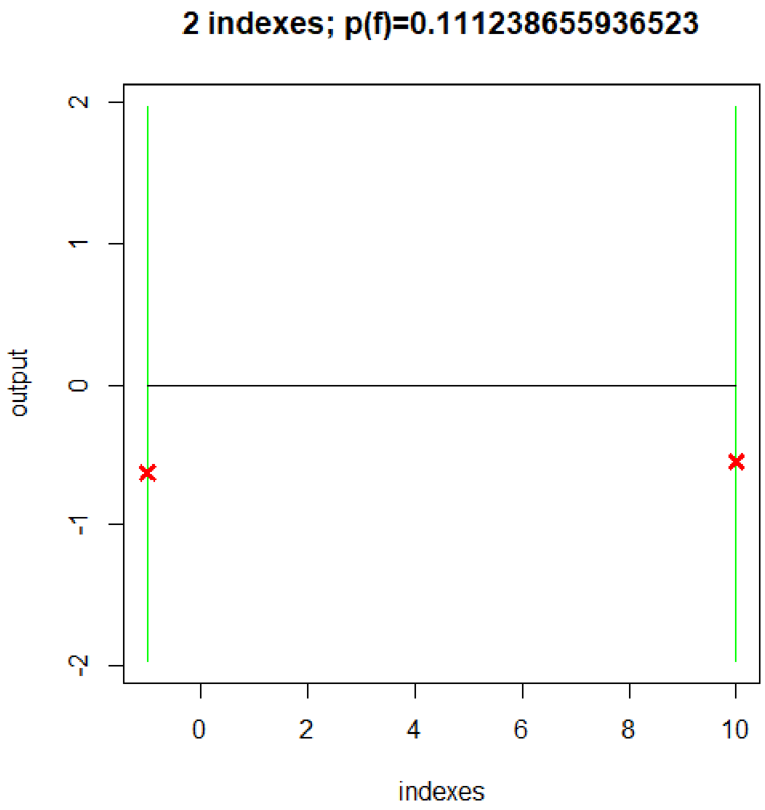
$$P(f = v) = P(f(x_1) = v_1, \ldots, f(x_n) = v_n)$$
$$p_f(v) = p_{f(x_1),\ldots,f(x_n)}(v_1, \ldots, v_n)$$

Gaussian process

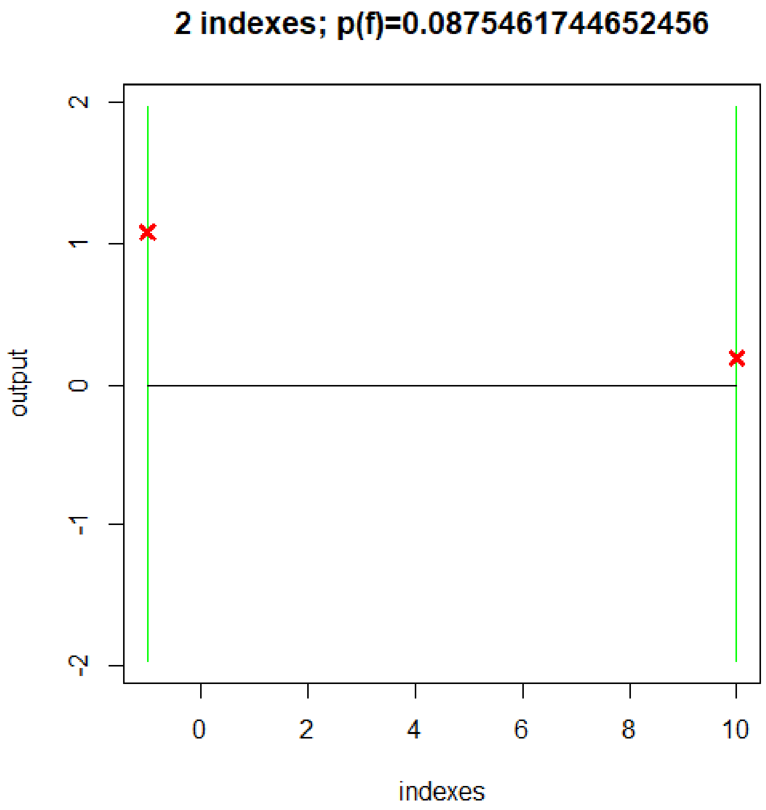
$\forall x_1, \ldots, x_n \in \mathcal{X} : (f(x_1), \ldots, f(x_n))$ has a multivariate normal distribution

=> We will be able to compute P(f = x)

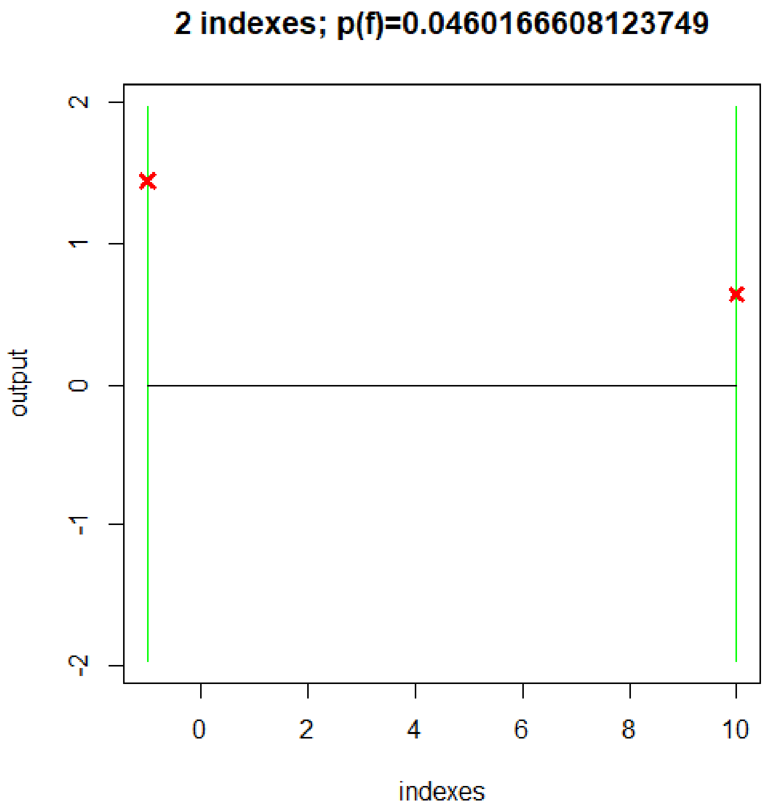
Gaussian process



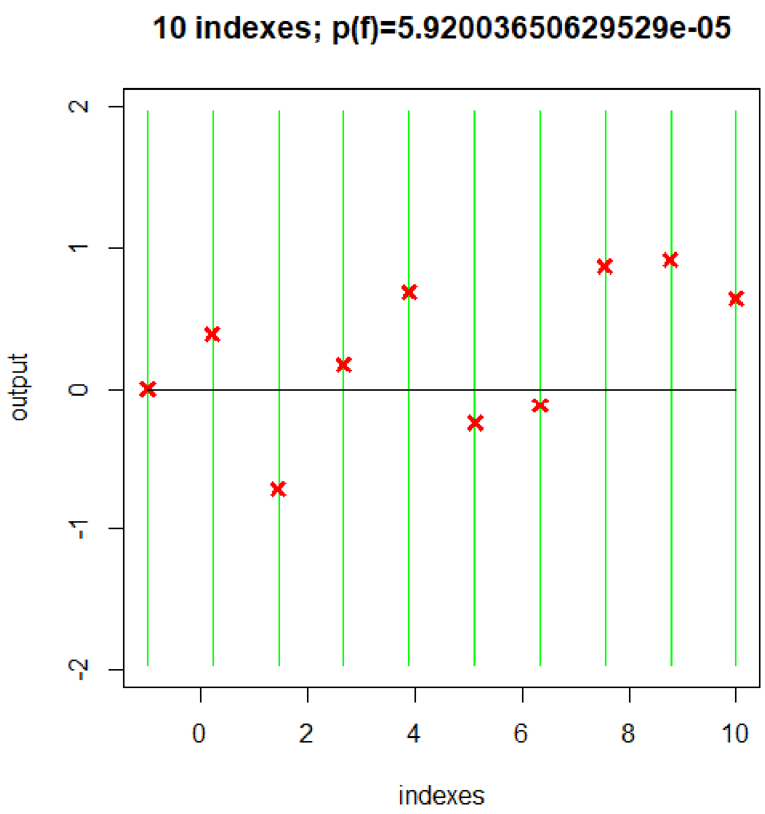
Gaussian process



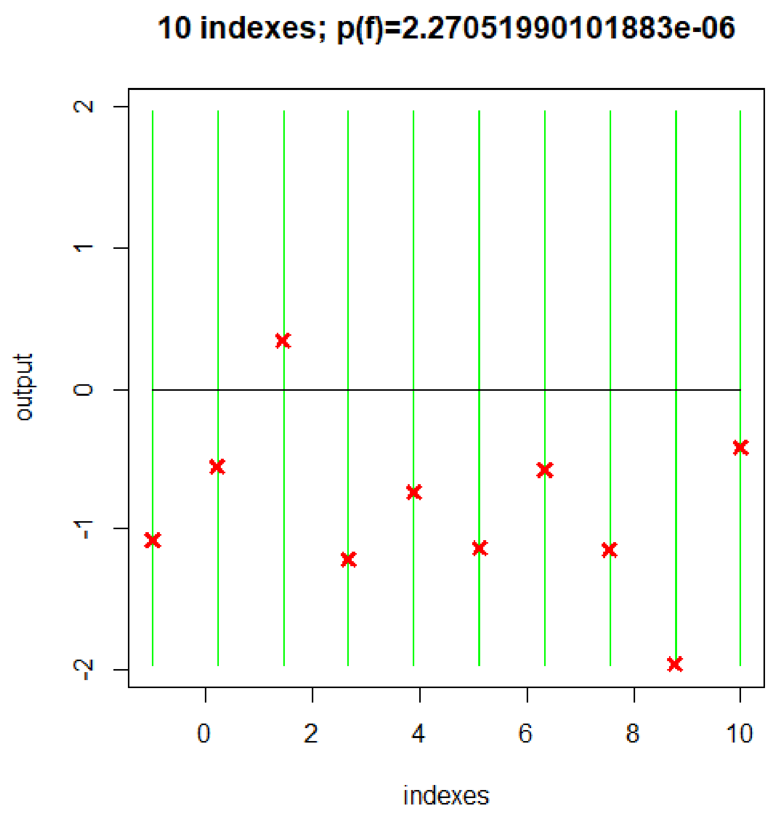
Gaussian process



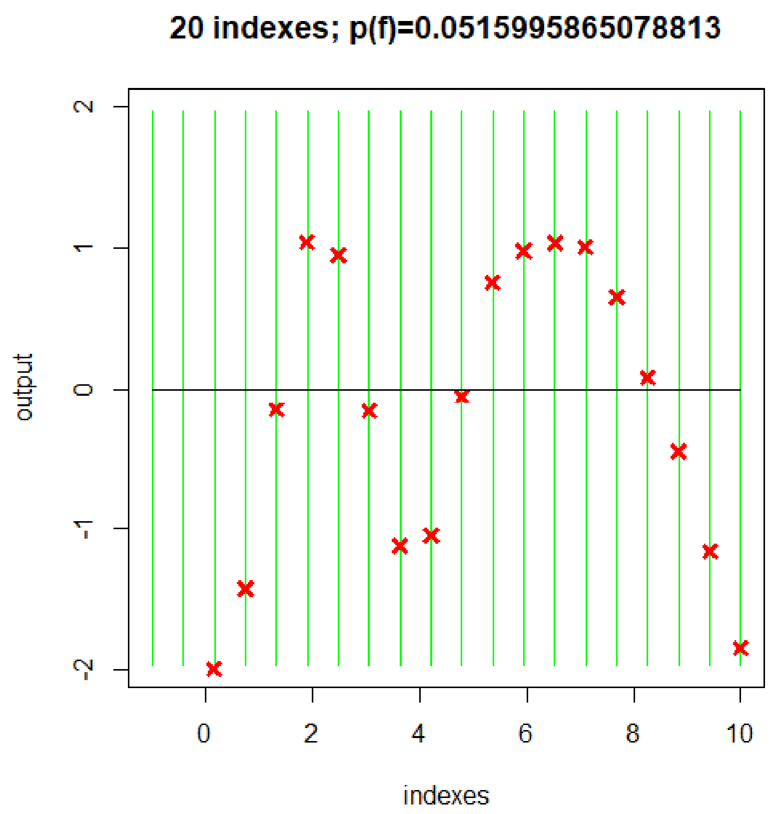
Gaussian process



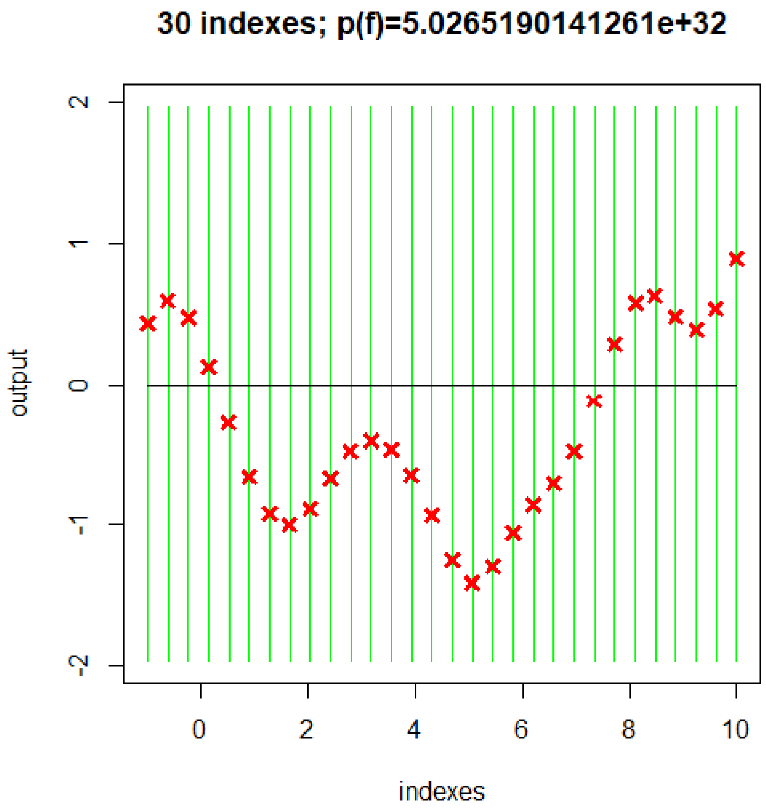
Gaussian process



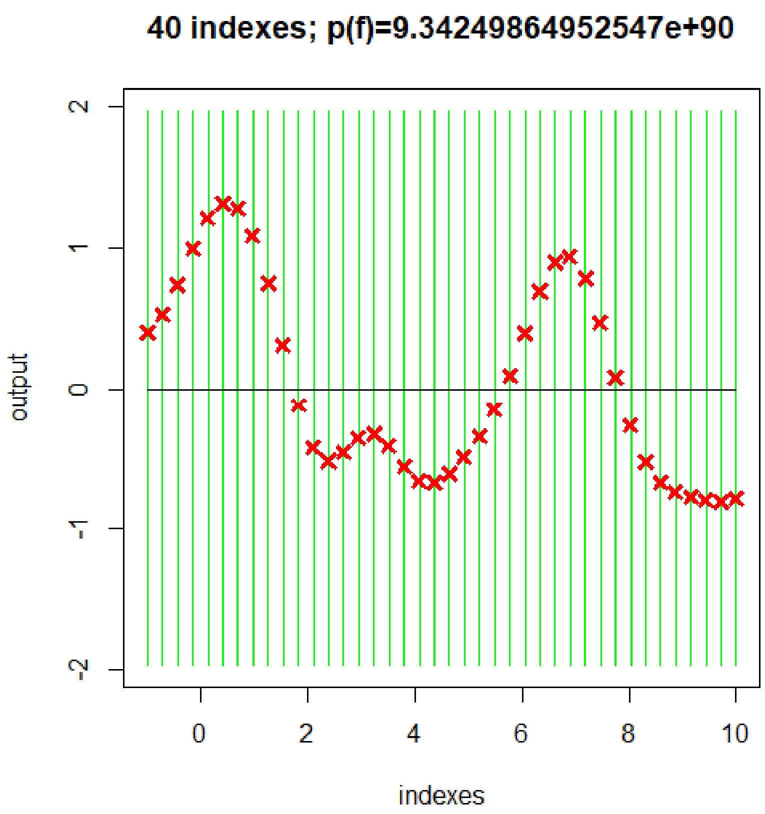
Gaussian process



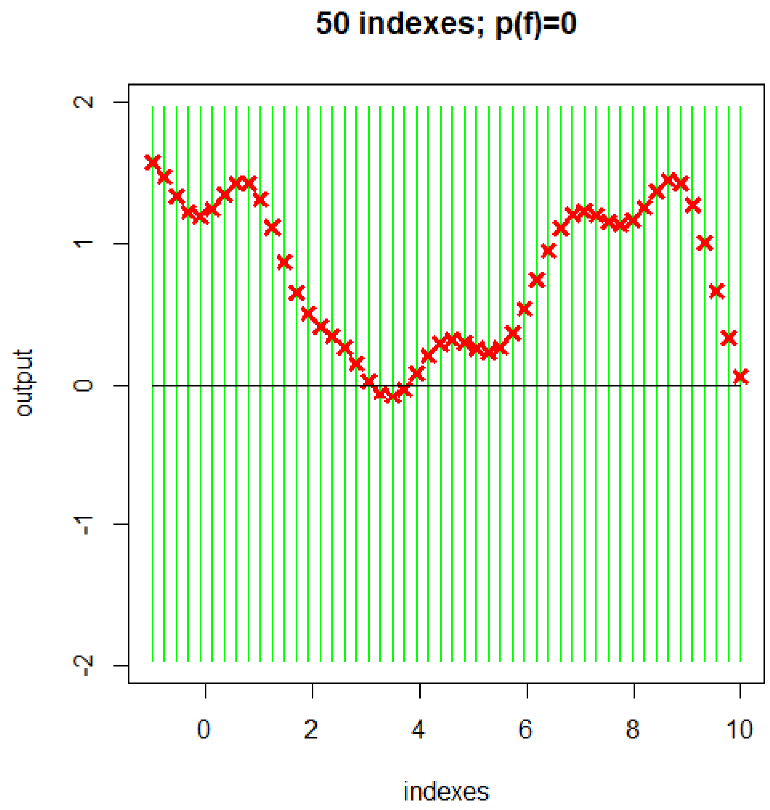
Gaussian process



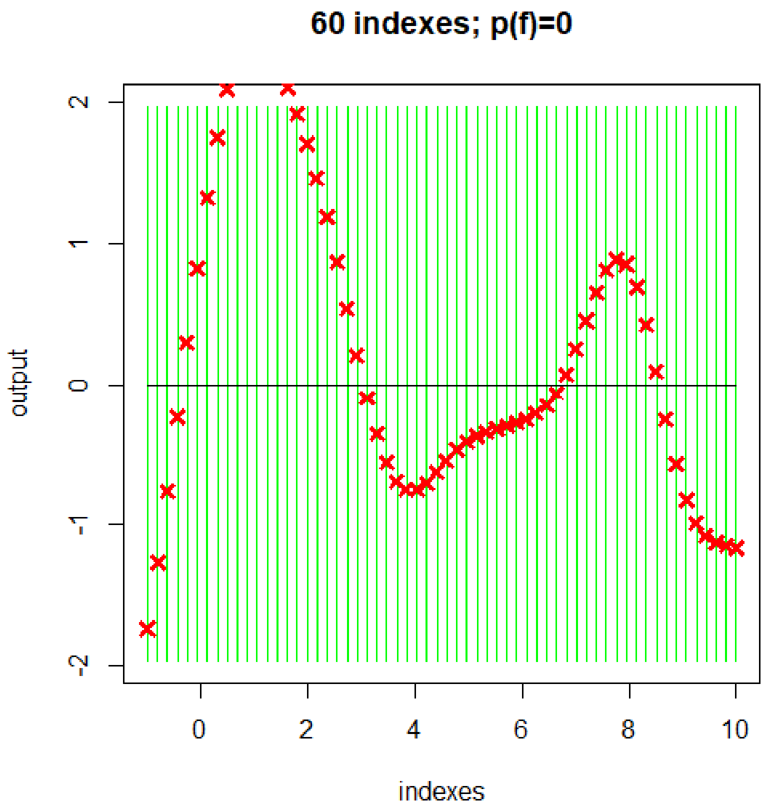
Gaussian process



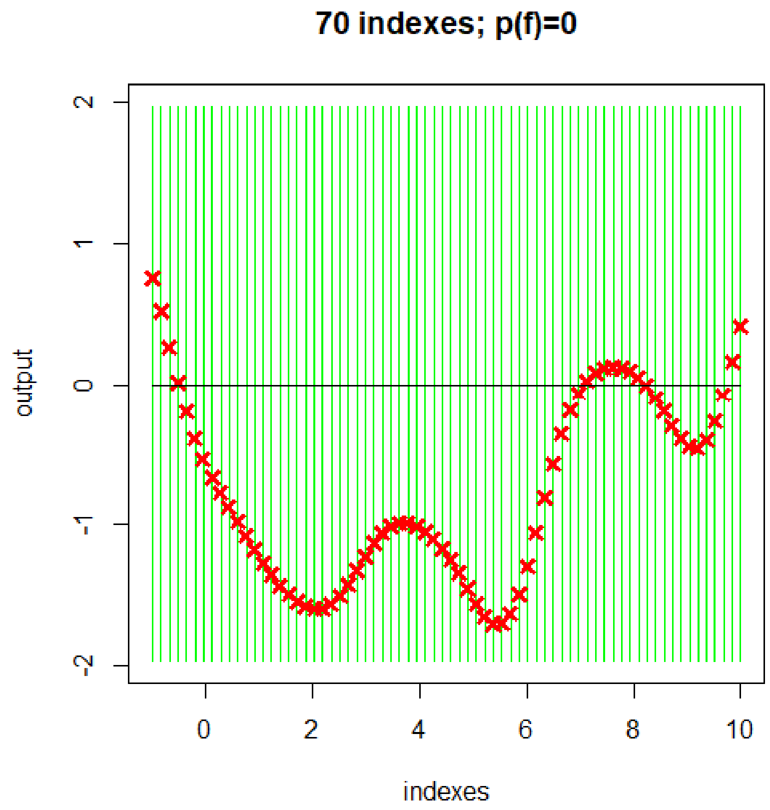
Gaussian process



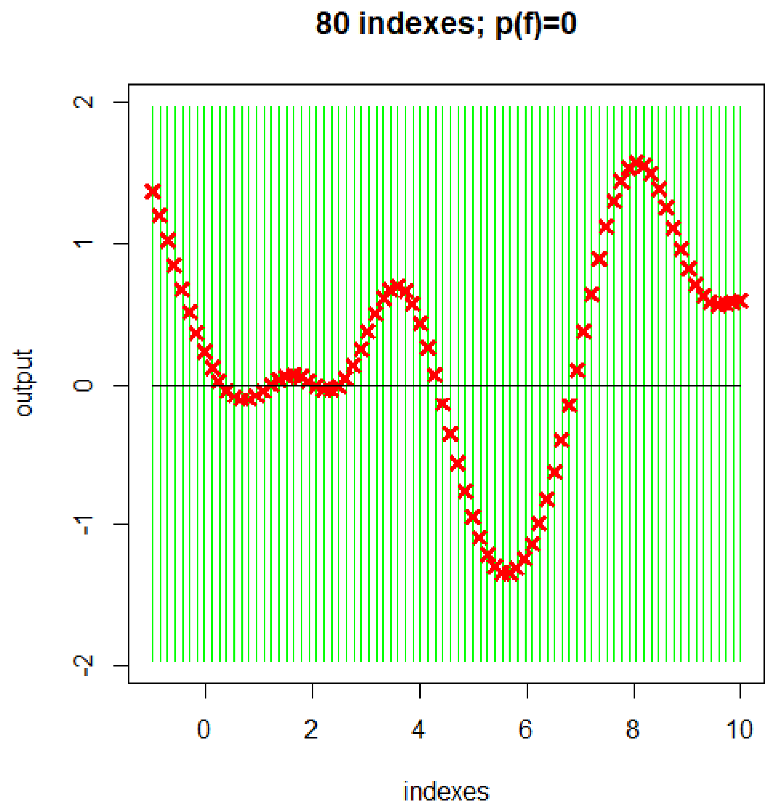
Gaussian process



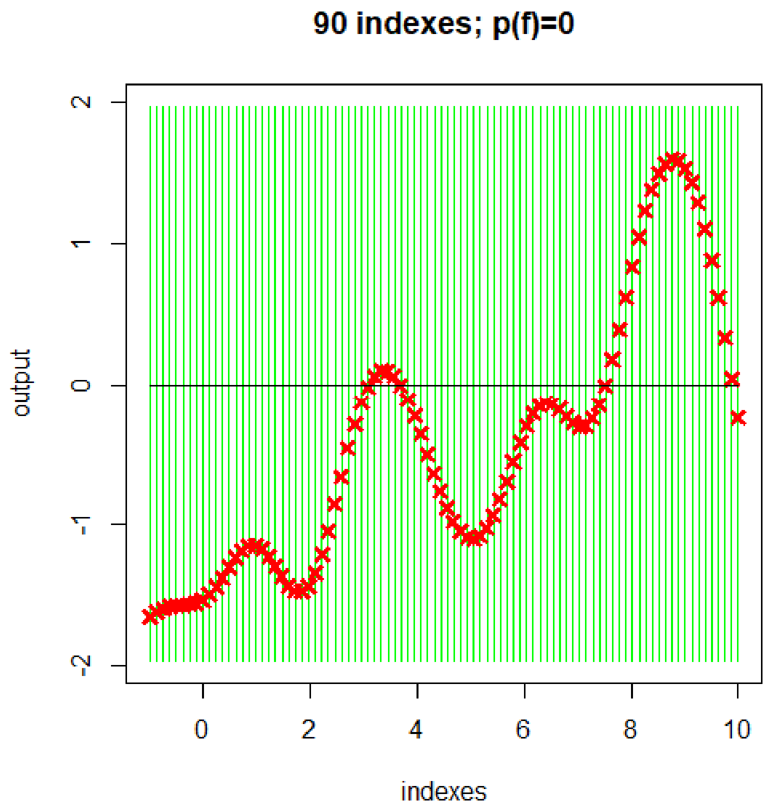
Gaussian process



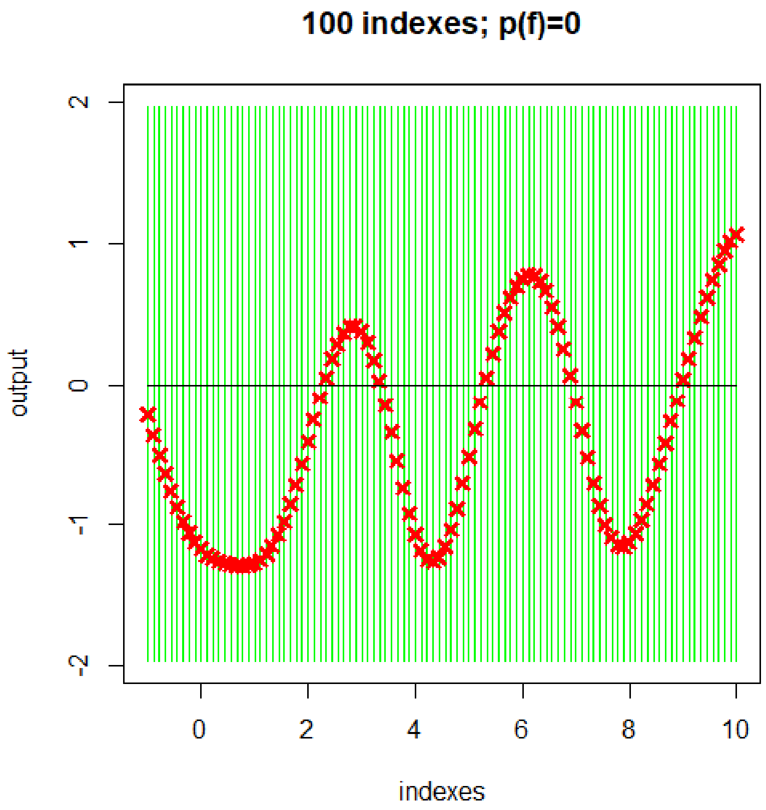
Gaussian process



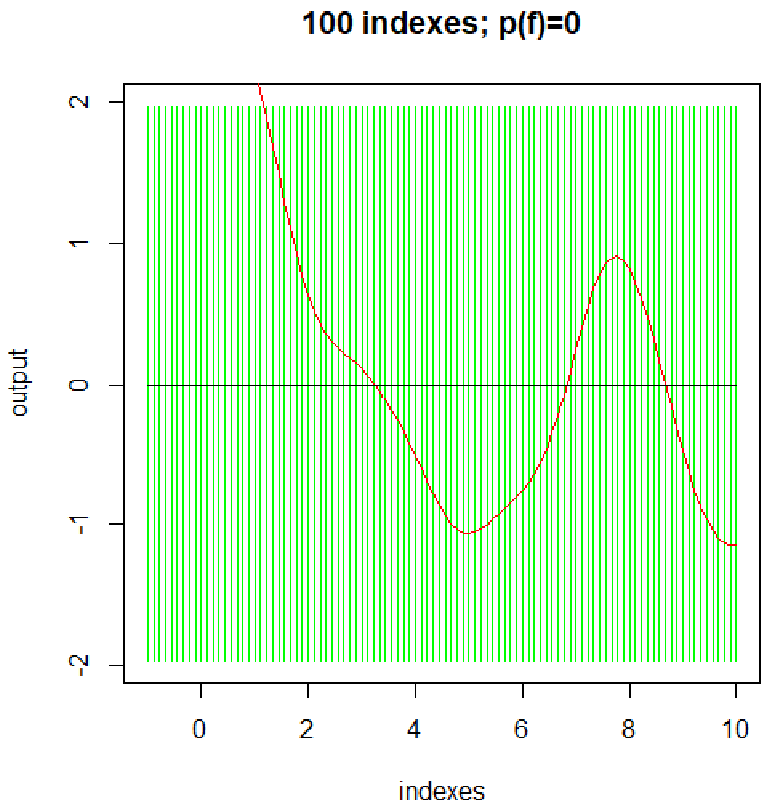
Gaussian process



Gaussian process



Gaussian process



Gaussian process

$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot))$$

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

$$x_1, \ldots, x_m \in \mathcal{X}$$

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_m) \end{bmatrix}, \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix} \right)$$

- m – function
- k – kernel function

Gaussian Process Regression

$$y^{(i)} = f(x^{(i)}) + \varepsilon^{(i)}, \qquad i = 1, \ldots, m$$

$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$$

Gaussian Process Regression

Training

– At the same time with testing!!!

Gaussian Process Regression

Testing

$$\begin{bmatrix} \vec{f} \\ \vec{f}_* \end{bmatrix} \Bigg| X, X_* \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right)$$

$$\begin{bmatrix} \vec{\varepsilon} \\ \vec{\varepsilon}_* \end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix} \sigma^2 I & \vec{0} \\ \vec{0}^T & \sigma^2 I \end{bmatrix}\right)$$

Gaussian Process Regression

Testing

$$\begin{bmatrix}\vec{y}\\ \vec{y}_*\end{bmatrix}\Bigg| X, X_* = \begin{bmatrix}\vec{f}\\ \vec{f}_*\end{bmatrix} + \begin{bmatrix}\vec{\varepsilon}\\ \vec{\varepsilon}_*\end{bmatrix} \sim \mathcal{N}\left(\vec{0}, \begin{bmatrix}K(X,X)+\sigma^2 I & K(X,X_*)\\ K(X_*,X) & K(X_*,X_*)+\sigma^2 I\end{bmatrix}\right)$$

$$\vec{y_*} \mid \vec{y}, X, X_* \sim \mathcal{N}(\mu^*, \Sigma^*)$$

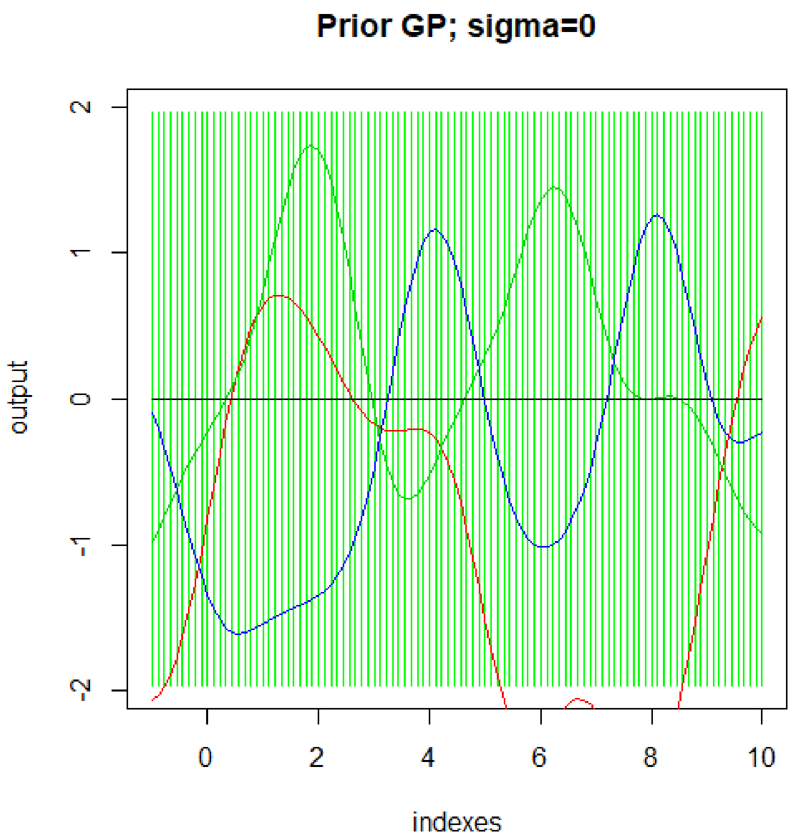
$$\mu^* = K(X_*,X)\left(K(X,X)+\sigma^2 I\right)^{-1}\vec{y}$$

$$\Sigma^* = K(X_*,X_*)+\sigma^2 I - K(X_*,X)\left(K(X,X)+\sigma^2 I\right)^{-1}K(X,X_*)$$

Gaussian Process Regression

Visualization **= application of the definition on multiple points (indexes)**

Before regression/training/testing



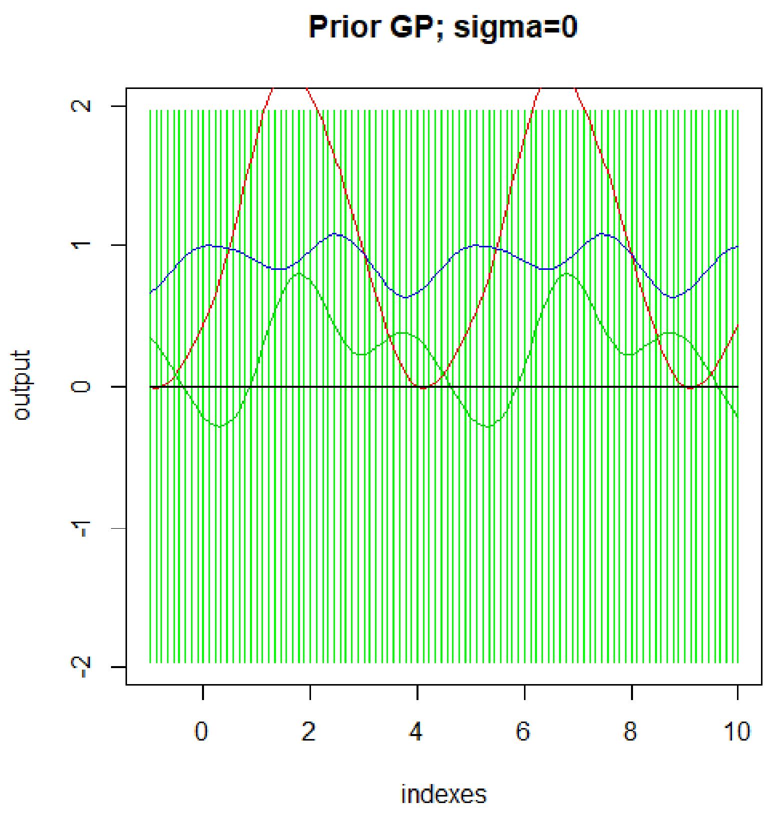
$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot,\cdot))$$

$$k(x,y) = 1^2 \cdot e^{-\frac{\|x-y\|^2}{2\cdot 1^2}}$$

Gaussian Process Regression

Visualization **= application of the definition on multiple points (indexes)**

Before regression/training/testing



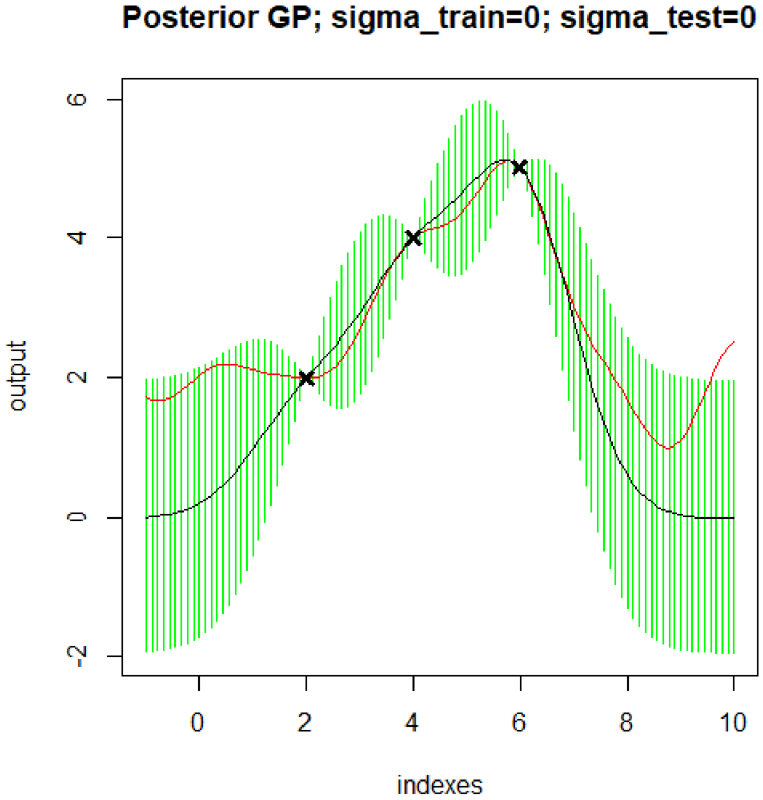
$$f(\cdot) \sim \mathcal{GP}(0, k(\cdot, \cdot))$$

$$k(x, y) = 1^2 \cdot e^{-\frac{2}{2^2} \sin^2(\pi \frac{(x-y)}{5})}$$

Gaussian Process Regression

Visualization = testing on multiple points (indexes)

At regression/training/testing time



$$\vec{y_*} \mid \vec{y}, X, X_* \sim \mathcal{N}(\mu^*, \Sigma^*)$$

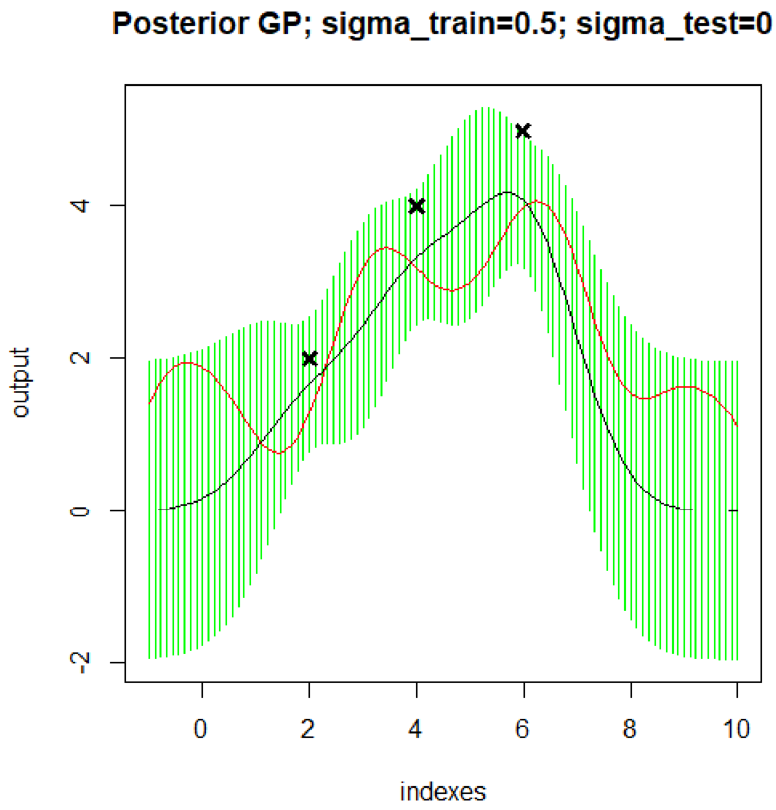
$$\mu^* = K(X_*, X)\left(K(X,X) + \sigma^2 I\right)^{-1} \vec{y}$$
$$\Sigma^* = K(X_*, X_*) + \sigma^2 I - K(X_*, X)\left(K(X,X) + \sigma^2 I\right)^{-1} K(X, X_*)$$

$$k(x,y) = 1^2 \cdot e^{-\frac{\|x-y\|^2}{2\cdot 1^2}}$$

Gaussian Process Regression

Visualization = testing on multiple points (indexes)

At regression/training/testing time



$$\vec{y_*} \mid \vec{y}, X, X_* \sim \mathcal{N}(\mu^*, \Sigma^*)$$

$$\mu^* = K(X_*, X)\left(K(X,X) + \sigma^2 I\right)^{-1}\vec{y}$$
$$\Sigma^* = K(X_*, X_*) + \sigma^2 I - K(X_*, X)\left(K(X,X) + \sigma^2 I\right)^{-1} K(X, X_*)$$

$$k(x,y) = 1^2 \cdot e^{-\frac{\|x-y\|^2}{2\cdot 1^2}}$$

Source:

MIT, 6867 ML, Fall 2006, Tommi Jaakkola, HW5, pr. 2

Regression and clustering. Gene expression

The EM algorithm for the Gaussian Process Mixture Model

EM/GPMM

Gene expression represents the process by which the information contained within a gene (our DNA) becomes a useful product, such as a protein.

The **expression level** of a gene indicates the amount of gene-product in the cell.

Observed and latent data

Expression level at t_1	Expression level at t_2	…	Expression level at t_{30}	Generated by Gaussian Process #…
-0.09900893	0.2818237	…	0.1033819	1
-0.00857957	0.1534053	…	0.08532405	2
-0.03709729	-0.00954268	…	0.01441636	1
…	…	…	…	…

t_1	t_2	…	t_{30}
0.0000000	0.2166616	…	6.2831853

Particular EM algorithm

EM/GPMM

$$k(x,x') = \sigma_f^2 \exp\left(-\frac{(x-x')^2}{2\rho^2}\right) + \sigma_n^2 \delta(x,x')$$

$$K_j^{(t)} \overset{\text{not.}}{=} K(h_j^{(t)})$$

E:

$$\gamma_{ij}^{(t+1)} \overset{\text{not.}}{=} \frac{\pi_j^{(t)} \mathcal{N}(x_i; 0, K_j^{(t)}))}{\sum_{l=1}^{k} \pi_l^{(t)} \mathcal{N}(x_i; 0, K_l^{(t)}))}$$

M:

$$\pi_j^{(t+1)} = \frac{\sum_{i=1}^{n} \gamma_{ij}^{(t+1)}}{n}$$

Particular EM algorithm

$$\theta_l \in \{(\sigma_f)_l, (\sigma_n)_l, \rho_l\}$$

$$\frac{\partial Q}{\partial \theta_l} = \frac{1}{2}\sum_{i=1}^{n}\left(\gamma_{il}^{(t+1)} x_i^T (K_l^{(t)})^{-1}\left(\frac{\partial K_l}{\partial \theta_l}\right)^{(t)}(K_l^{(t)})^{-1} x_i\right) - \frac{\sum_{i=1}^{n}\gamma_{il}^{(t+1)}}{2}\mathrm{Tr}\left((K_l^{(t)})^{-1}\left(\frac{\partial K_l}{\partial \theta_l}\right)^{(t)}\right)$$

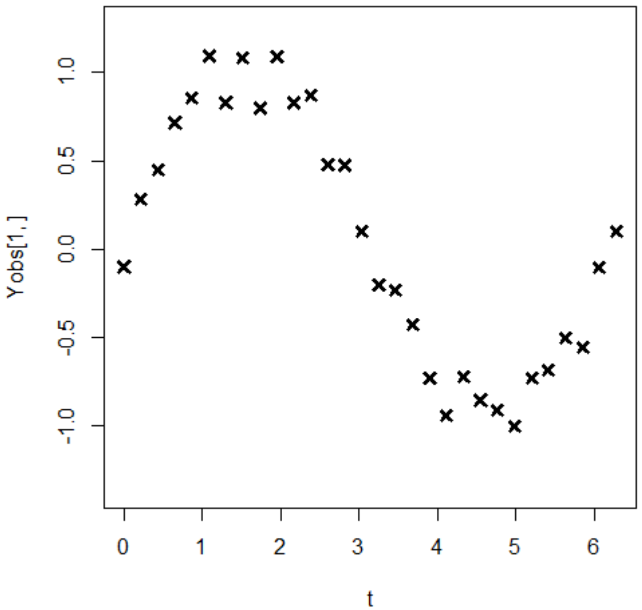
$$\frac{\partial k(x,y)}{\partial \sigma_f} = 2\sigma_f \exp\left(-\frac{(x-y)^2}{2\rho^2}\right)$$

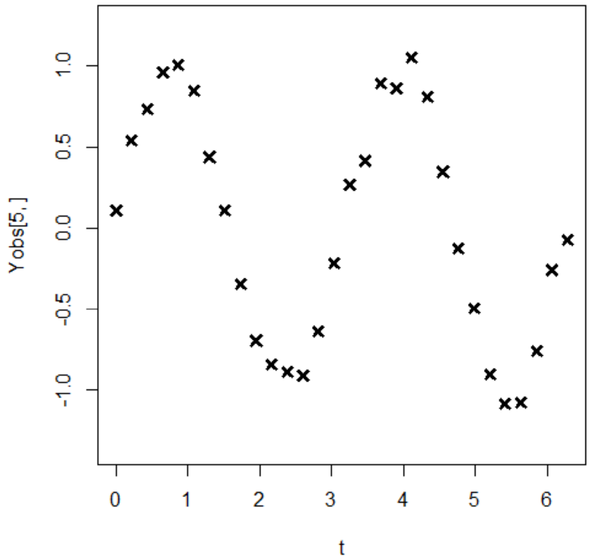
$$\frac{\partial k(x,y)}{\partial \sigma_n} = 2\sigma_n \delta(x,y)$$

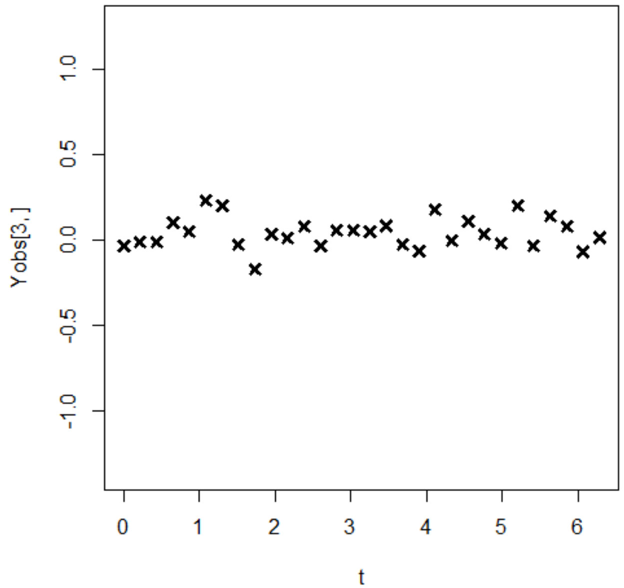
$$\frac{\partial k(x,y)}{\partial \rho} = \sigma_f^2 \exp\left(-\frac{(x-y)^2}{2\rho^2}\right)\frac{(x-y)^2}{\rho^3}$$

To find $\theta_l^{(t+1)}$ we used the gradient ascent method.

What we knew about the data...



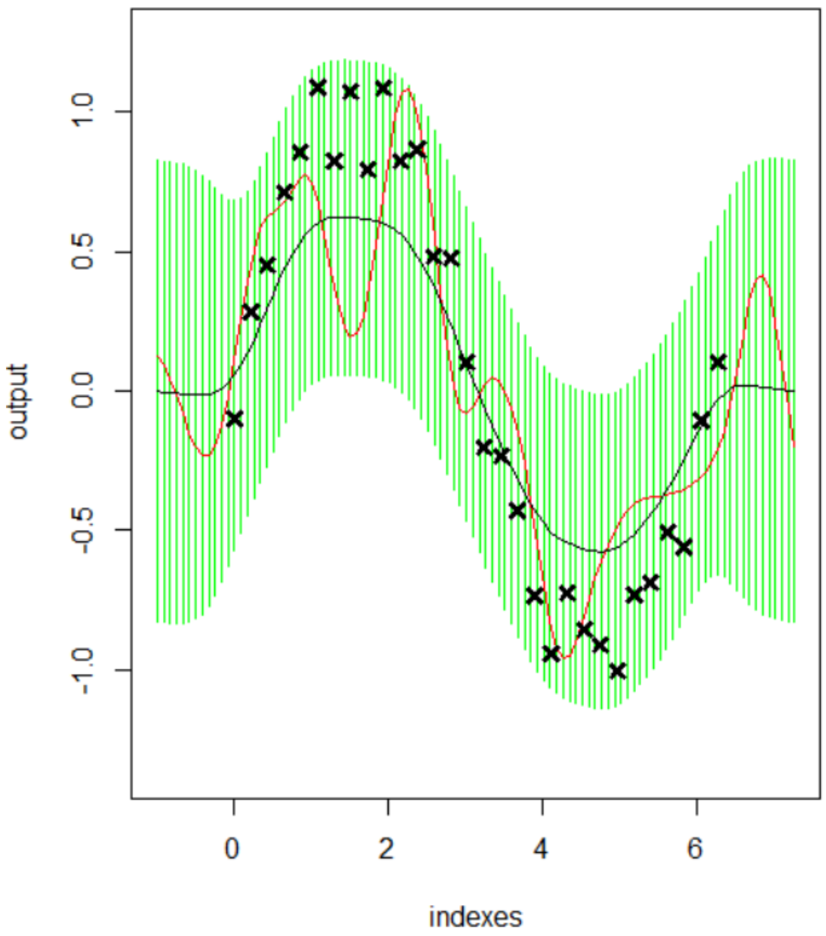




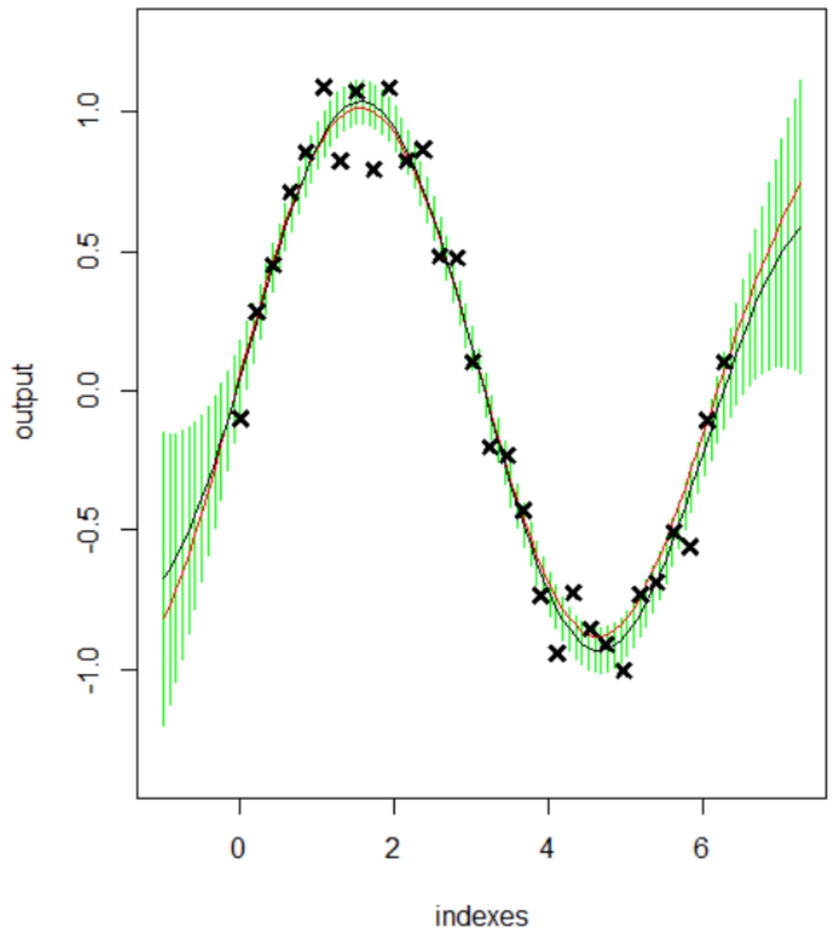
- => 3 types of curves (functions)

Results

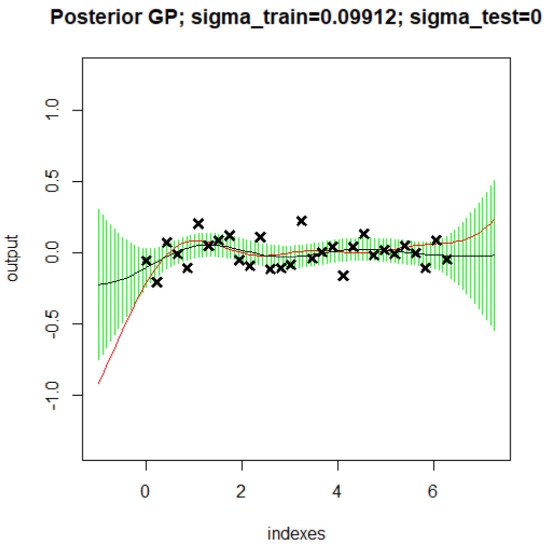
Random fit

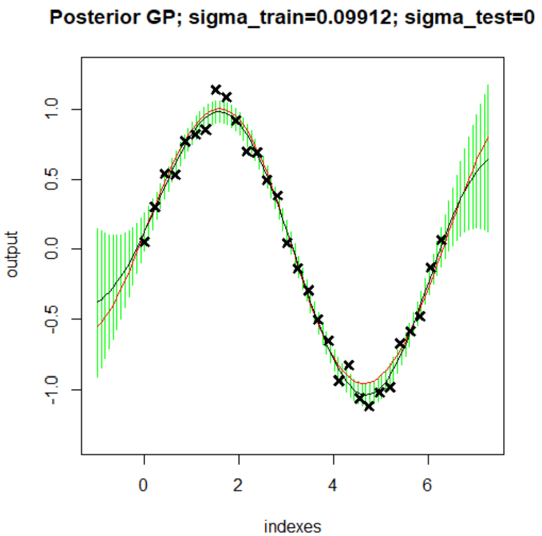


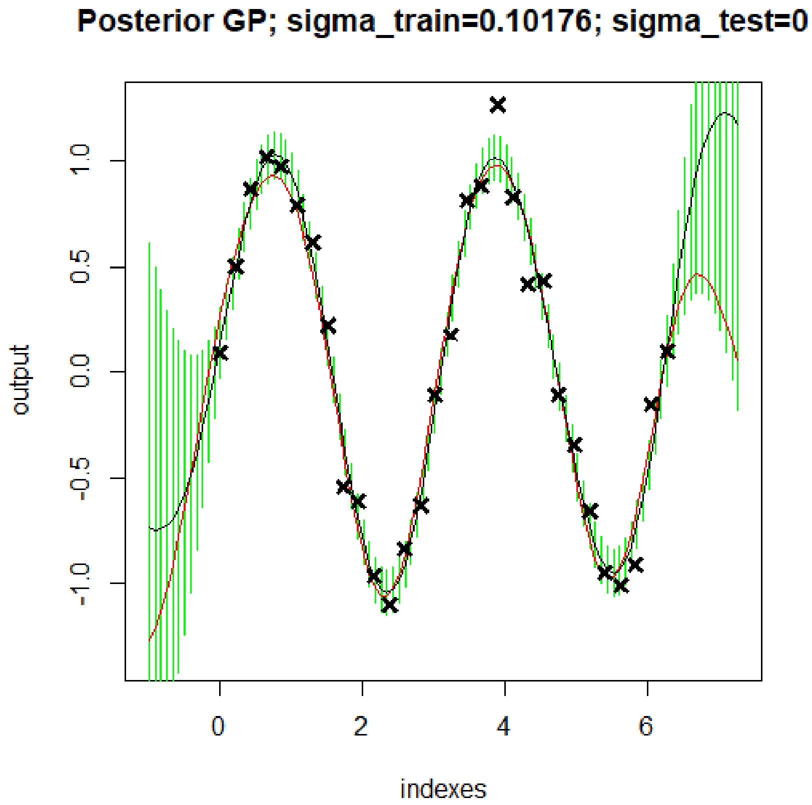
EM/GPMM fit



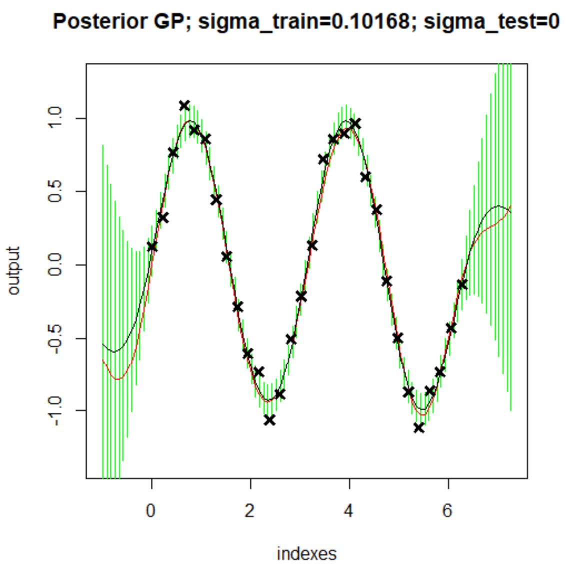
Results: k=2


Posterior GP; sigma_train=0.09912; sigma_test=0
output
indexes

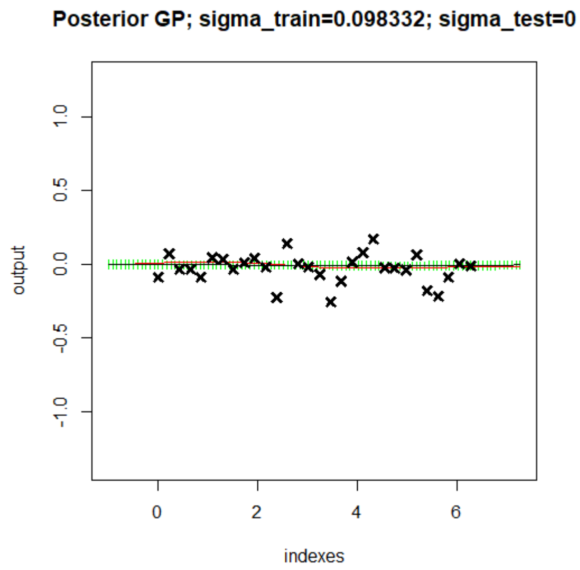

Posterior GP; sigma_train=0.09912; sigma_test=0
output
indexes


Posterior GP; sigma_train=0.10176; sigma_test=0
output
indexes

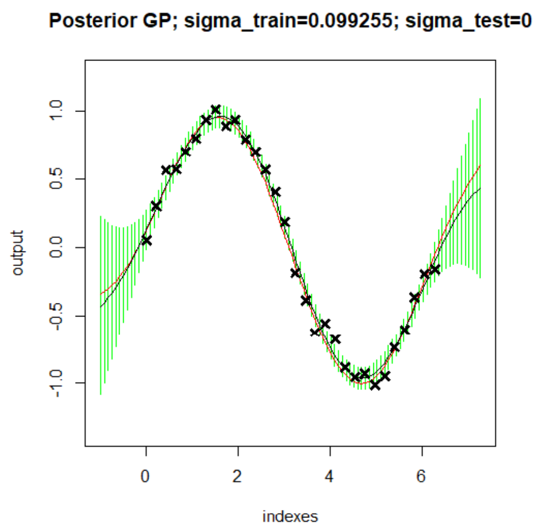
Results: k=3



Posterior GP; sigma_train=0.10168; sigma_test=0
output
indexes

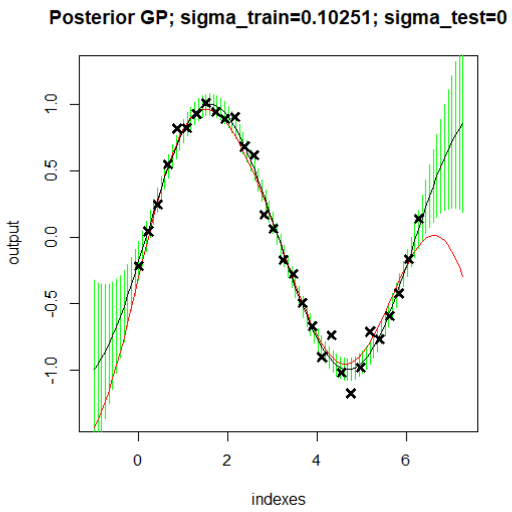


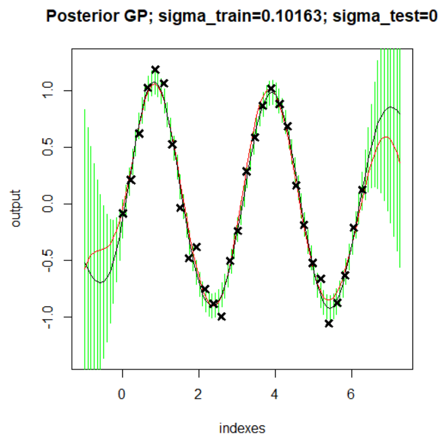
Posterior GP; sigma_train=0.098332; sigma_test=0
output
indexes

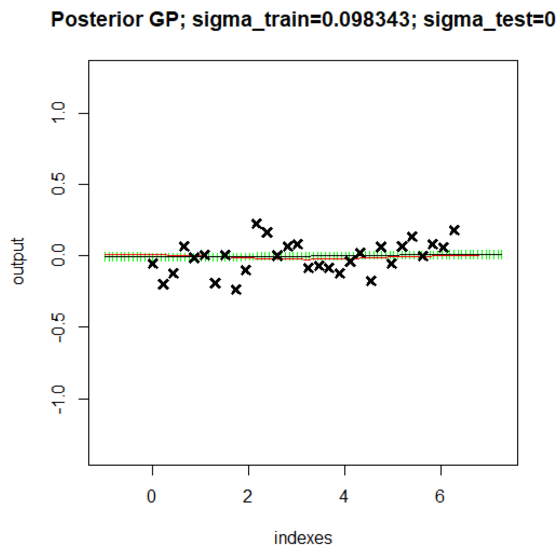


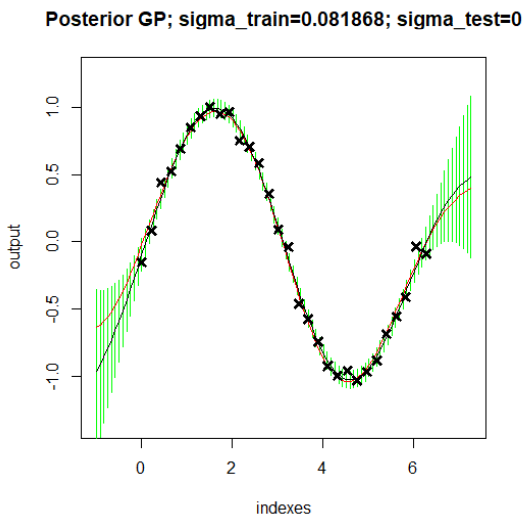
Posterior GP; sigma_train=0.099255; sigma_test=0
output
indexes

Results: k=4

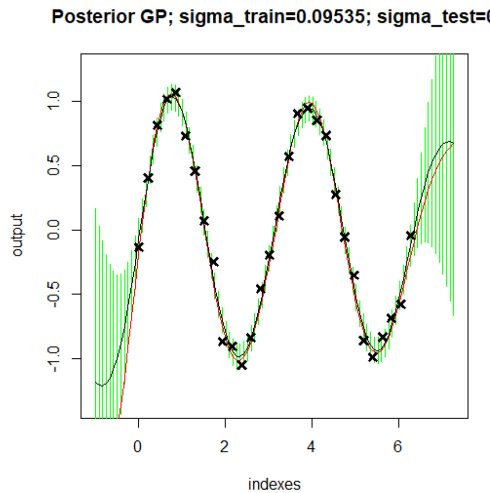

Posterior GP; sigma_train=0.10251; sigma_test=0
output
indexes

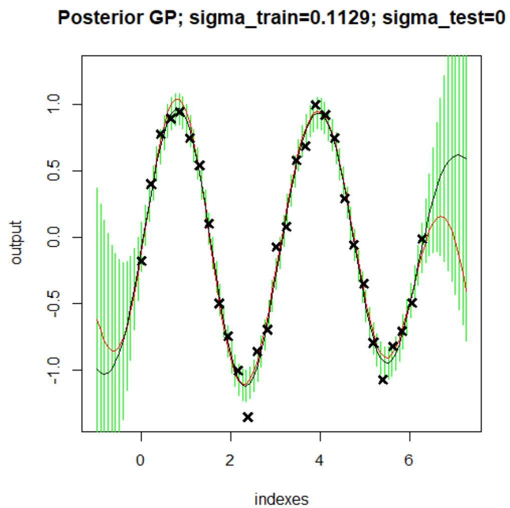

Posterior GP; sigma_train=0.10163; sigma_test=0
output
indexes

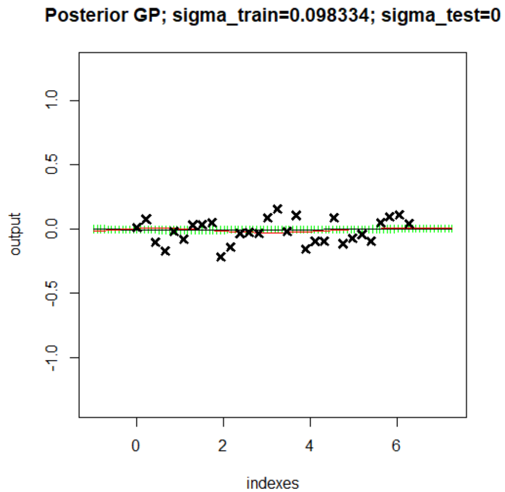

Posterior GP; sigma_train=0.098343; sigma_test=0
output
indexes

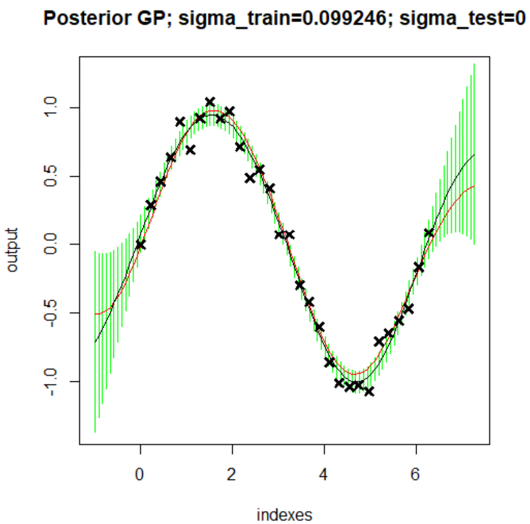

Posterior GP; sigma_train=0.081868; sigma_test=0
output
indexes

Results: k=5









NULL cluster

Future work

- Simple alternatives
  - Perform a hard-clustering
    - Using hierarchical clustering
    - Using k-means
  - Perform EM/GMM
  - Then, in each cluster, fit the parameters of a GP via MLE
- Dirichlet processes
- Other datasets
- Numerical analysis of the quality of the clusters we obtained

Bibliography

[1] Nello Cristianini & John Shawe-Taylor (2000): *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, 1 edition. Cambridge University Press.

[2] A. P. Dempster, N. M. Laird & D. B. Rubin (1977): *Maximum likelihood from incomplete data via the EM algorithm*. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B* 39(1), pp. 1–38.

[3] Guojun Gan, Chaoqun Ma & Jianhong Wu (2007): *Data clustering - theory, algorithms, and applications*. SIAM. Available at `http://bookstore.siam.org/sa20/`.

[4] James Hensman, Magnus Rattray & Neil D. Lawrence (2014): *Fast nonparametric clustering of structured time-series*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, doi:10.1109/TPAMI.2014.2318711.

[5] Tommi Jaakkola: *MIT, 6867 Machine Learning, Fall 2006, Problem Set 5, pr. 2*. Available at `https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/assignments/hw5.pdf`

[6] Tommi Jaakkola: *MIT, 6867 Machine Learning, Fall 2006, Problem Set 5, pr. 2, Dataset*. Available at `https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/assignments/prob2_data.zip`

[7] Thomas M. Mitchell (1997): *Machine Learning*, 1 edition. McGraw-Hill, Inc., New York, NY, USA.

[8] Andrew Ng: *Stanford University, Machine Learning course, Gaussian Processes*. Available at `http://cs229.stanford.edu/summer2019/gaussian_processes.pdf`.

[9] Carl Edward Rasmussen & Christopher K. I. Williams (2005): *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

[10] Claude Sammut & Geoffrey I. Webb, editors (2017): *Encyclopedia of Machine Learning and Data Mining*. Springer, doi:10.1007/978-1-4899-7687-1.

[11] Volker Tresp (2001): *Mixtures of Gaussian processes*. In: *Advances in Neural Information Processing Systems 13*, MIT Press, pp. 654–660.