Inferring in Ordinary Differential Equations with Latent Functions through Gaussian Processes

Investigators: **Neil D. Lawrence** and Magnus Rattray
Researchers: Pei Gao, Jennifer Withers, Michalis Titsias, Antti Honkela

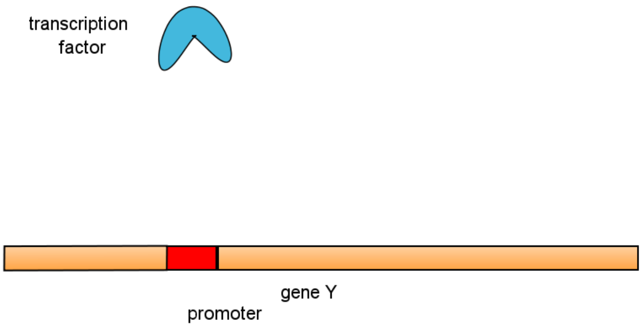
RSS Manchester Local Group

8th October 2008

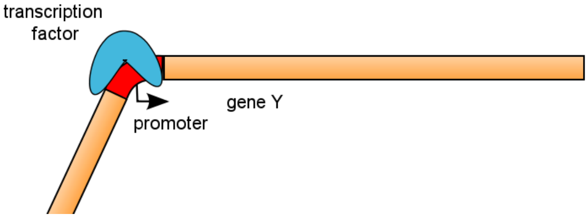
Outline

Outline

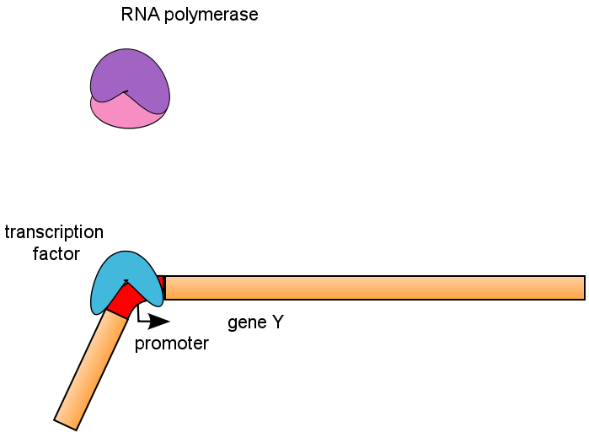
Transcriptional regulation of gene expression



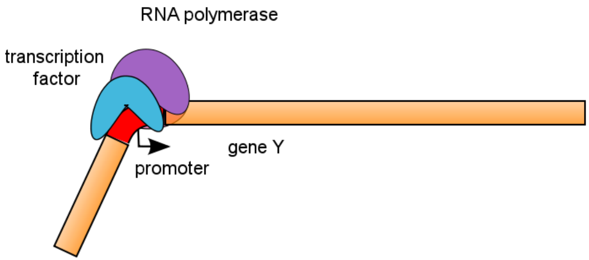
Transcriptional regulation of gene expression



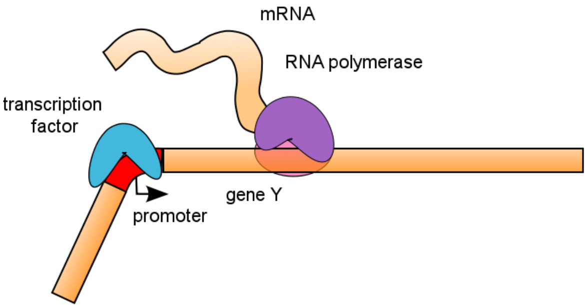
Transcriptional regulation of gene expression



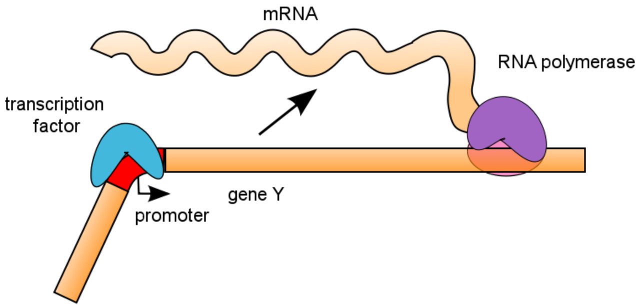
Transcriptional regulation of gene expression



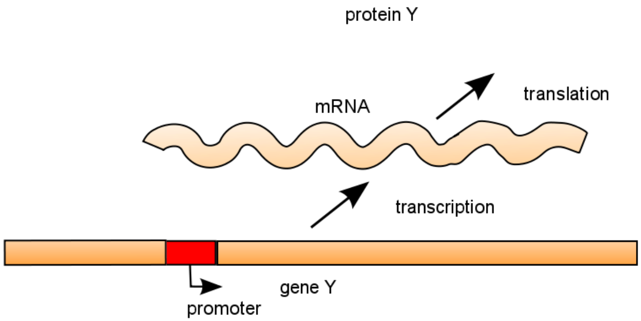
Transcriptional regulation of gene expression



Transcriptional regulation of gene expression



Transcriptional regulation of gene expression



RNA Polymerase

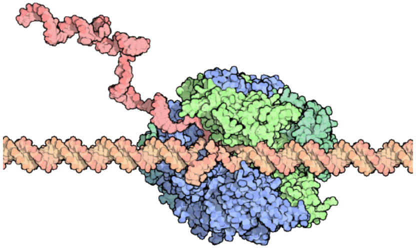


Figure: RNA Polymerase transcribing RNA from DNA. Image from "Molecule of the Month" at the protein data bank: http://mgl.scripps.edu/people/goodsell/pdb/pdb98/pdb98_1.html

Outline

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

ODE Model of Activation

- Linear Activation Model (Barenco et al., 2006, Genome Biology)

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

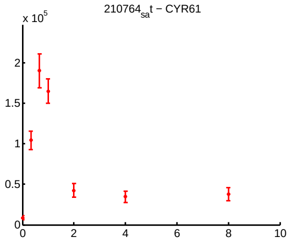
- $x_j(t)$ – concentration of gene j's mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline B_j, sensitivity S_j and decay D_j
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

Why use a model-based approach?

- Co-regulated genes can differ greatly in their expression profiles

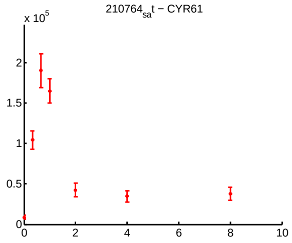
Why use a model-based approach?

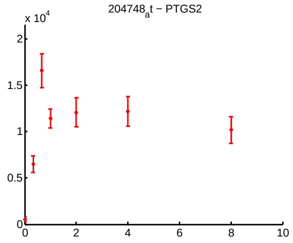
- Co-regulated genes can differ greatly in their expression profiles



Why use a model-based approach?

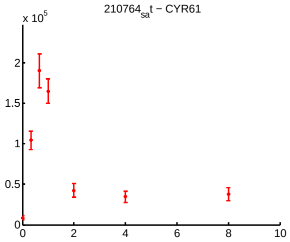
- Co-regulated genes can differ greatly in their expression profiles

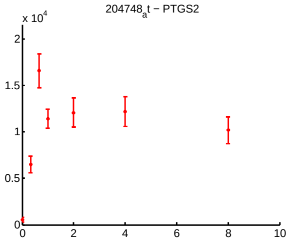




Why use a model-based approach?

- Co-regulated genes can differ greatly in their expression profiles

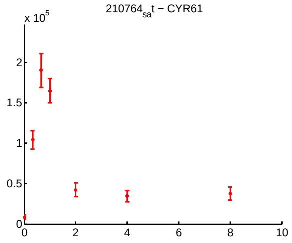


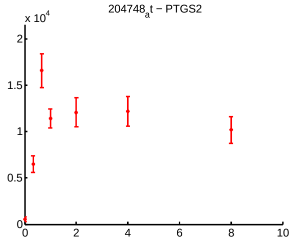


- Clustering cannot be relied on to identify co-regulated genes

Why use a model-based approach?

- Co-regulated genes can differ greatly in their expression profiles





- Clustering cannot be relied on to identify co-regulated genes
- A model-based approach is required

Models of non-linear regulation

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j f(t)}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Khanin et al., 2006, PNAS 103

Models of non-linear regulation

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j f(t)}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Khanin et al., 2006, PNAS 103

Models of non-linear regulation

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j f(t)}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + \frac{S_j}{\gamma_j + f(t)} - D_j x_j(t)$$

  used by Khanin et al., 2006, PNAS 103

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Standard inference approach

- Previous approaches all use similar inference methodology:
  - Represent $f(t)$ as coarse-grained piecewise continuous function $[f_1, f_2, \ldots, f_d]$
  - Often discretize where data are collected
  - Treat f_i as additional model parameters
  - Use maximum likelihood or Bayesian MCMC to estimate $\{f_i\}$ along with other model parameters of interest
- Limitations:
  - Arbitrary choice of discretization points
  - Coarse-grain gives crude approximation to $f(t)$
  - Fine-grain leads to harder inference problem

Outline

Gaussian Distribution

Zero mean Gaussian distribution

- A multi-variate Gaussian distribution is defined by a mean and a covariance matrix.

$$N\left(\mathbf{f}|\mu,\mathbf{K}\right) = \frac{1}{\left(2\pi\right)^{\frac{N}{2}}\left|\mathbf{K}\right|^{\frac{1}{2}}}\exp\left(-\frac{\left(\mathbf{f}-\mu\right)^{\mathrm{T}}\mathbf{K}^{-1}\left(\mathbf{f}-\mu\right)}{2}\right).$$

- We will consider the special case where the mean is zero,

$$N\left(\mathbf{f}|\mathbf{0},\mathbf{K}\right) = \frac{1}{\left(2\pi\right)^{\frac{N}{2}}\left|\mathbf{K}\right|^{\frac{1}{2}}}\exp\left(-\frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}\right).$$

Do Quick Review Later

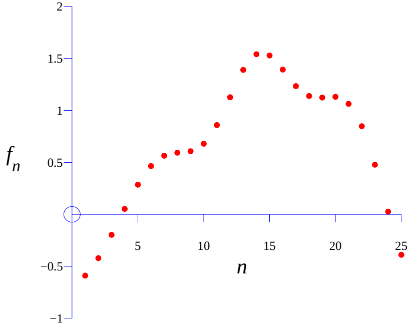
Sampling a Function

Multi-variate Gaussians

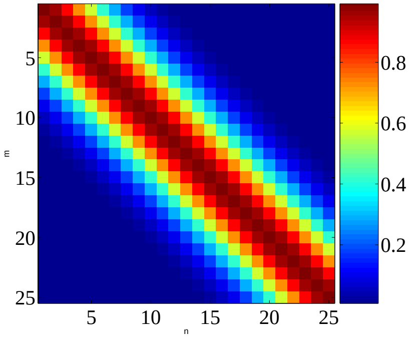
- We will consider a Gaussian with a particular structure of covariance matrix.
- Generate a single sample from this 25 dimensional Gaussian distribution, $\mathbf{f} = [f_1, f_2 \dots f_{25}]$.
- We will plot these points against their index.

Gaussian Distribution Sample

`demGPSample`



(a)



(b)

Figure: (a) 25 instantiations of a function, f_n, (b) colormap of covariance matrix.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Prediction of f_2 from f_1

```
demGPCov2D([1 2])
```

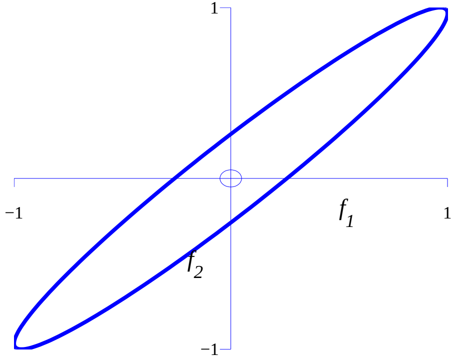


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

```
demGPCov2D([1 2])
```

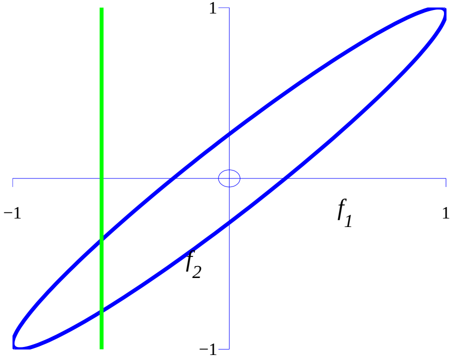


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

```
demGPCov2D([1 2])
```

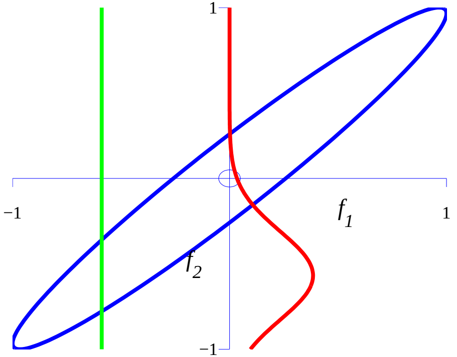


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGPCov2D([1 5])
```

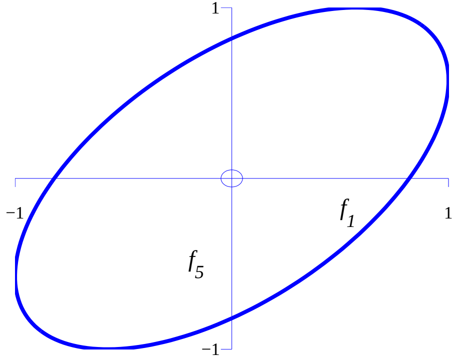


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGPCov2D([1 5])
```

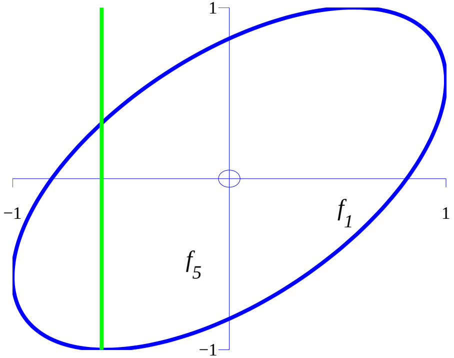


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGPCov2D([1 5])
```

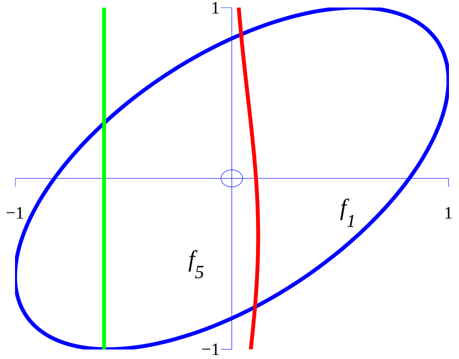


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

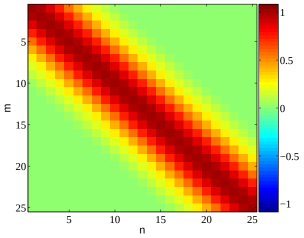
Covariance Functions

Where did this covariance matrix come from?

RBF Kernel Function

$$k\left(t, t'\right) = \alpha \exp\left(-\frac{\left\|t - t'\right\|^2}{2l^2}\right)$$

- Covariance matrix is built using the *inputs* to the function t.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.



Covariance Samples

`demCovFuncSample`

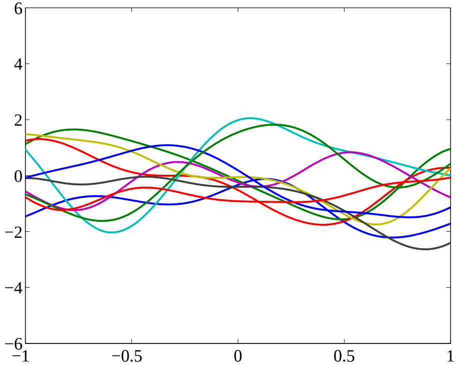


Figure: RBF kernel with $l = 10^{-\frac{1}{2}}$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

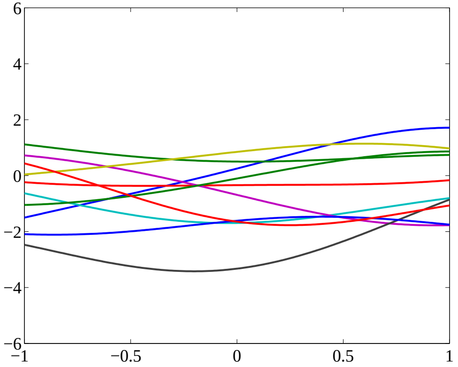


Figure: RBF kernel with $l = 1$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

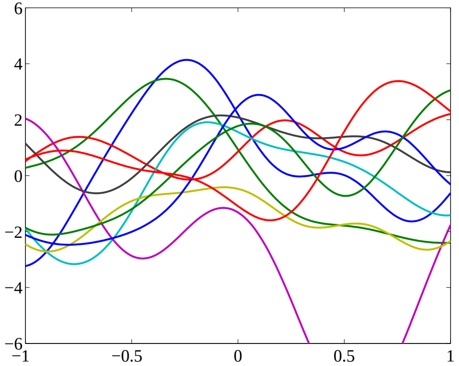


Figure: RBF kernel with $l = 0.3$, $\alpha = 4$

Gaussian Process Regression

`demRegression`

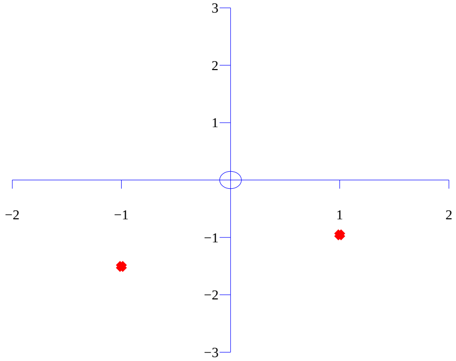


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

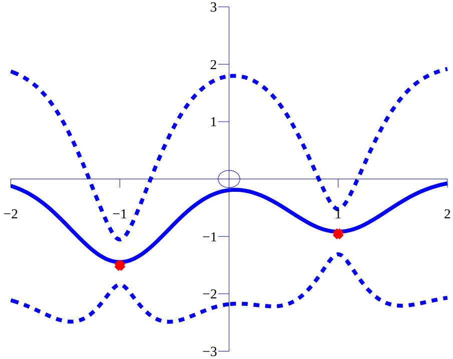


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

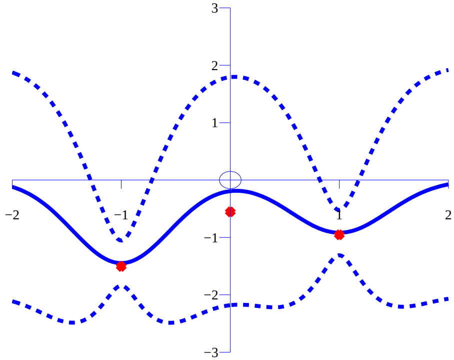


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

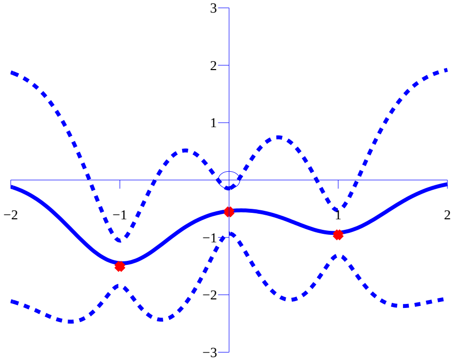


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

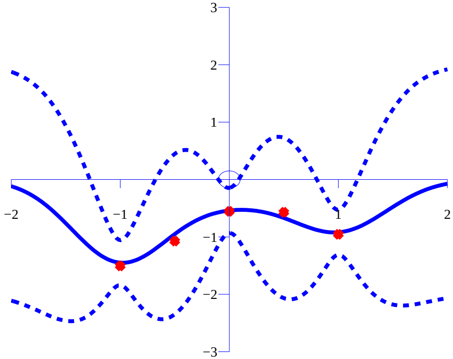


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

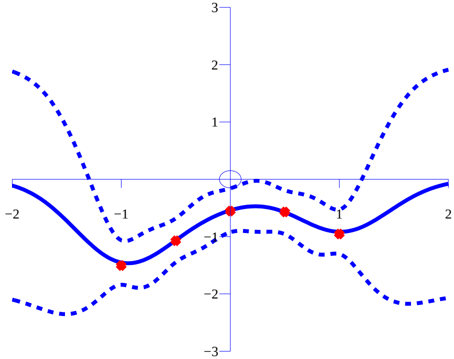


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

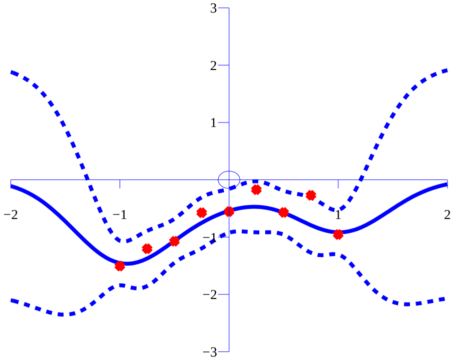


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

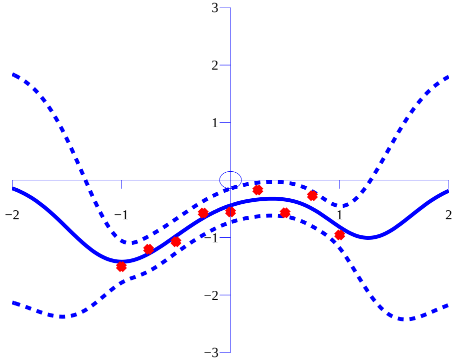
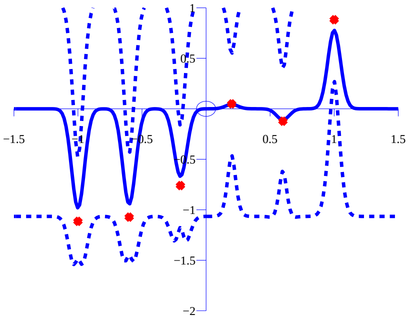


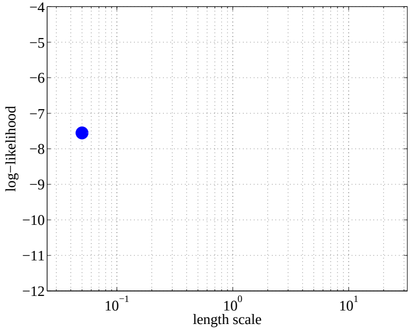
Figure: Examples include WiFi localization, C14 callibration curve.

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



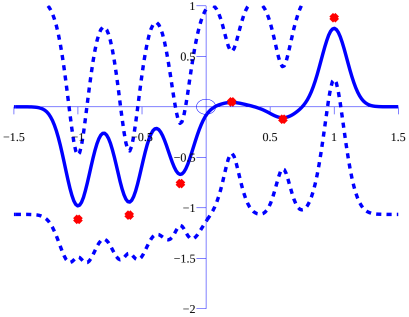


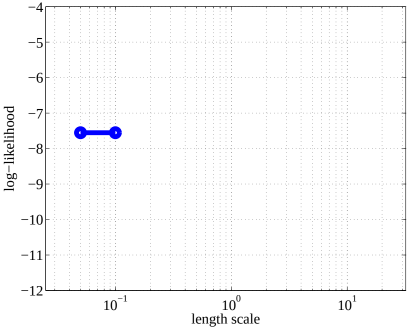
$$\log N\left(\mathbf{f}|\mathbf{0},\mathbf{K}\right) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



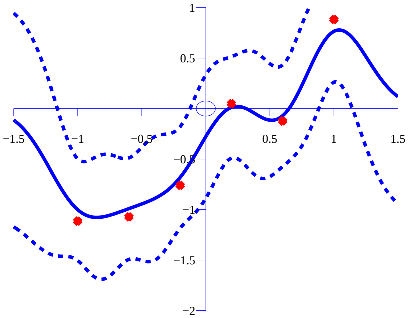


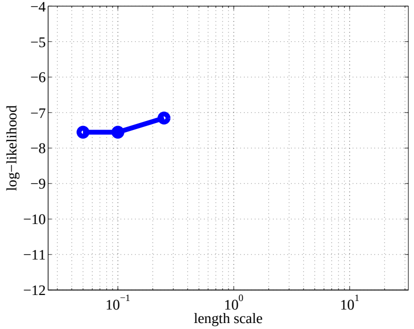
$$\log N(\mathbf{f}|\mathbf{0},\mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



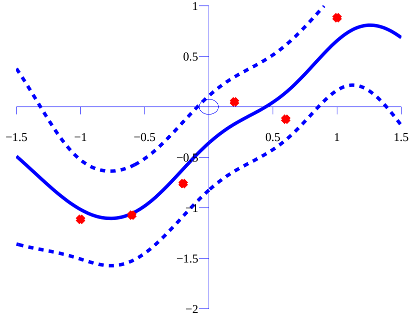


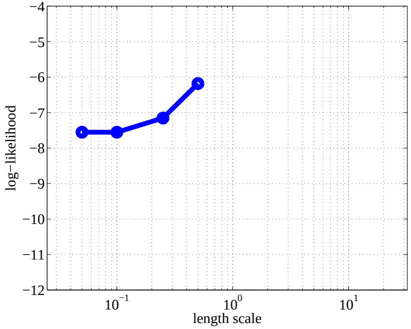
$$\log N\left(\mathbf{f}|\mathbf{0},\mathbf{K}\right) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log\left|\mathbf{K}\right| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



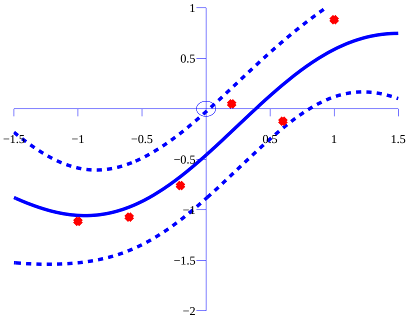


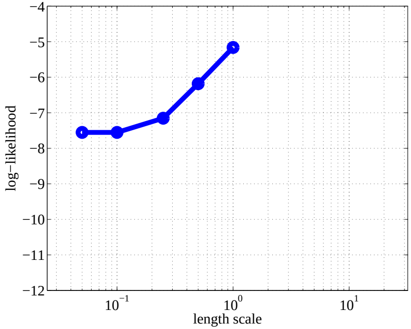
$$\log N(\mathbf{f}|\mathbf{0},\mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

```
demOptimiseKern
```



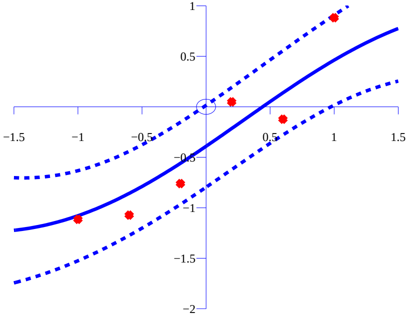


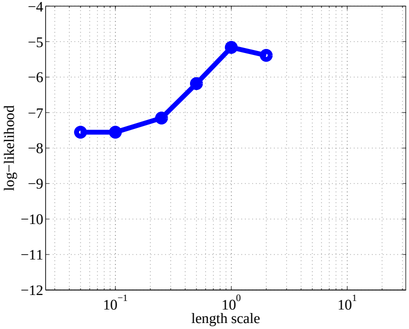
$$\log N(\mathbf{f}|\mathbf{0}, \mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



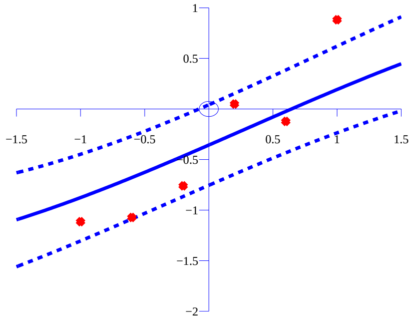


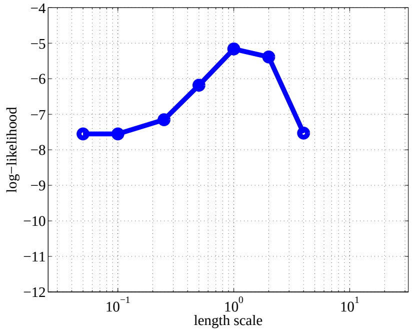
$$\log N\left(\mathbf{f}|\mathbf{0},\mathbf{K}\right) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log\left|\mathbf{K}\right| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



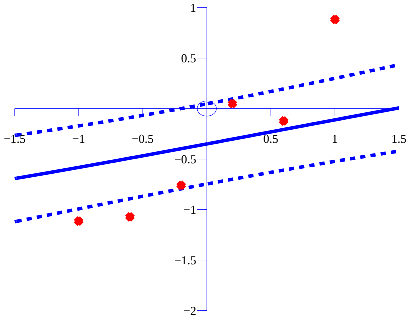


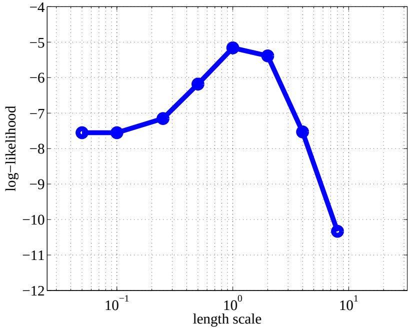
$$\log N(\mathbf{f}|\mathbf{0},\mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`



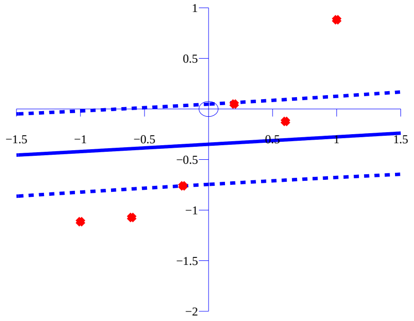


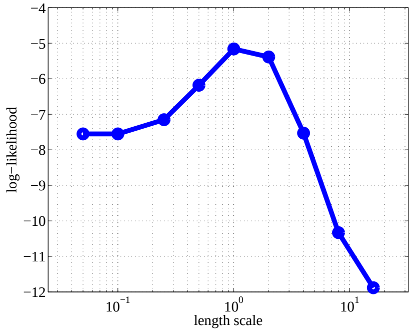
$$\log N(\mathbf{f}|\mathbf{0},\mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

`demOptimiseKern`





$$\log N(\mathbf{f}|\mathbf{0},\mathbf{K}) = -\frac{N}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{f}}{2}$$

Outline

Gaussian Processes

- Gaussian Process

$$f(t) \backsim \mathcal{GP}\left(m(t), k(t, t')\right)$$

where

$$\begin{aligned} m(t) &= \mathbb{E}[f(t)] = \langle f(t) \rangle \\ k(t, t') &= \mathbb{E}\left[(f(t) - m(t))(f(t') - m(t'))\right] \end{aligned}$$

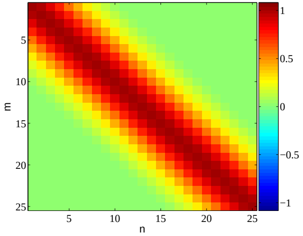
▸ We've done long review

Covariance Functions

RBF Kernel Function

$$k\left(t, t'\right) = \alpha \exp\left(-\frac{\left(t - t'\right)^2}{2l^2}\right)$$

- Covariance matrix is built using the *inputs* to the function t.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.

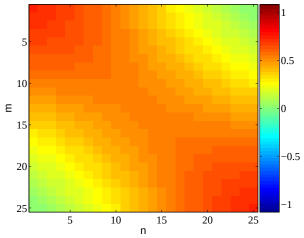


Different Covariance Functions

MLP Kernel Function

$$k\left(t, t'\right) = \alpha \sin^{-1}\left(\frac{wtt' + b}{\sqrt{wt^2 + b + 1}\sqrt{wt'^2 + b + 1}}\right)$$

- A non-stationary covariance matrix (Williams, 1997).
- Derived from a multi-layer perceptron (MLP).



Covariance Samples

`demCovFuncSample`

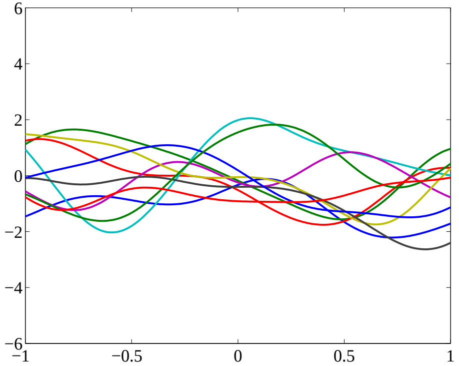


Figure: RBF kernel with $\gamma = 10^{-\frac{1}{2}}$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

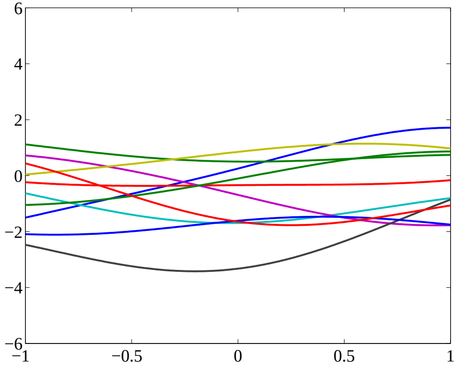


Figure: RBF kernel with $l = 1$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

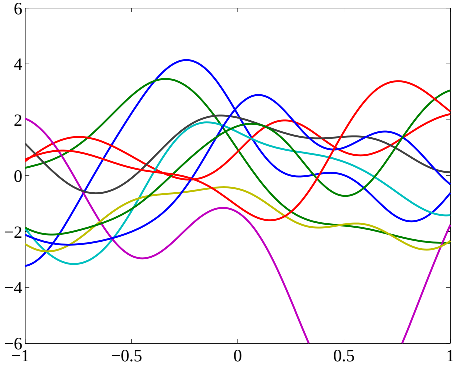


Figure: RBF kernel with $l = 0.3$, $\alpha = 4$

Covariance Samples

`demCovFuncSample`

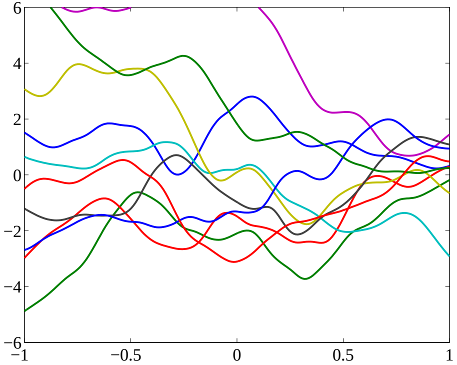


Figure: MLP kernel with $\alpha = 8$, $w = 100$ and $b = 100$

Covariance Samples

`demCovFuncSample`

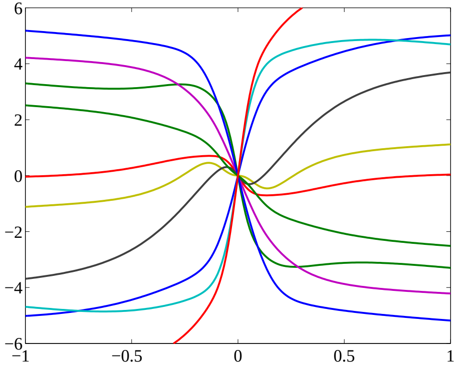


Figure: MLP kernel with $\alpha = 8$, $b = 0$ and $w = 100$

Linear Activation Model

Recall the linear model

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)\ .$$

This differential equation can be solved for $x_j(t)$ as

$$x_j(t) = \frac{B_j}{D_j} + S_j \int_0^t e^{-D_j(t-u)} f(u)\,\mathrm{d}u\ .$$

Note: This is a linear operation on $f(t)$.

If $f(t)$ is a zero mean Gaussian process then $x_i(t)$ is also a Gaussian process with mean $\frac{B_i}{D_i}$.

▸ Skip GP Properties

Two Properties of GPs

The integral of a GP is also a GP,

$$f(t) \sim N(\mathbf{0}, \mathbf{K}_{ff})$$

and

$$g(t) = \int_0^t f(u)\, du$$

then

$$g(t) \sim N(\mathbf{0}, \mathbf{K}_{gg}),$$

where

$$k_{gg}(t, t') = \int_0^t \int_0^{t'} k_{ff}(u, u')\, du du'$$

Two Properties of GPs

Product with deterministic function
Product with a deterministic funciton leads to another GP,

$$f\left(t\right) \sim N\left(\mathbf{0}, \mathbf{K}_{ff}\right),$$

and

$$g\left(t\right) = f\left(t\right) h\left(t\right)$$

where $h\left(t\right)$ is a deterministic function then,

$$g\left(t\right) \sim N\left(\mathbf{0}, \mathbf{K}_{gg}\right),$$

where

$$k_{gg}\left(t, t'\right) = h\left(t\right) k_{ff}\left(t, t'\right) h\left(t'\right)$$

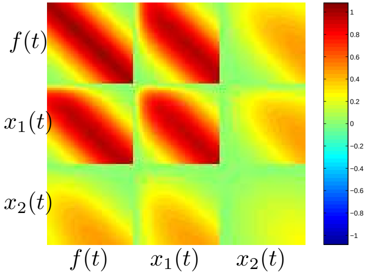
Covariance for Transcription Model

RBF covariance function for $f(t)$

$$x_i(t) = \frac{B_i}{D_i} + S_i \exp(-D_i t) \int_0^t f(u) \exp(D_i u) \,\mathrm{d}u.$$

- Joint distribution for $x_1(t)$, $x_2(t)$ and $f(t)$.
  - Here:

D_1	S_1	D_2	S_2
5	5	0.5	0.5



Skip SIM Samples

Joint Sampling of $x(t)$ and $f(t)$ from Covariance

`gpsimTest`

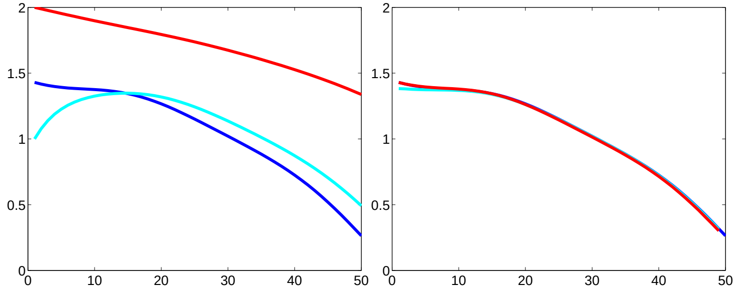
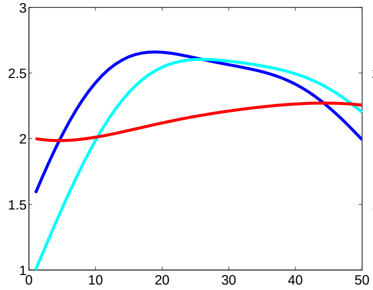


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

Joint Sampling of $x(t)$ and $f(t)$ from Covariance

`gpsimTest`



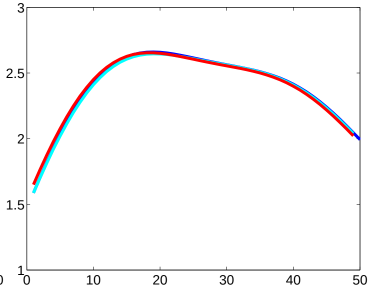


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

Joint Sampling of $x(t)$ and $f(t)$ from Covariance

`gpsimTest`

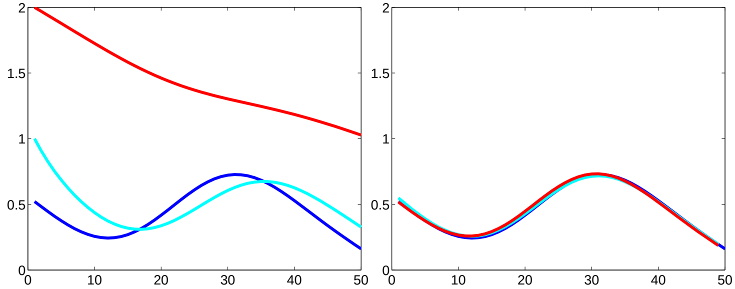


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

Covariance Function

Any linear opearation of a GP $\Longrightarrow$ Related GP

$$f(t) \backsim \mathcal{GP}\left(0, k_{ff}\left(t, t'\right)\right) \Longrightarrow x_j(t) \backsim \mathcal{GP}\left(\frac{B_j}{D_j}, k_{xx}\left(t, t'\right)\right)$$

Hence, the cross-covariances between the genes is

$$k_{x_i,x_j}\left(t, t'\right) = S_i S_j \int_0^t \int_0^{t'} e^{-D_i(t-u)-D_j(t'-u')} k_{f,f}\left(t, t'\right) \mathrm{d}u \mathrm{d}u' \ .$$

Cross-covariances between $x_j(t)$ and $f(t)$ is

$$k_{x_j,f}\left(t, t'\right) = \int_0^t e^{-D_i(t-u)} k_{f,f}\left(t, t'\right) \mathrm{d}u \ .$$

Prediction of the transcription factor concentration $f(t)$

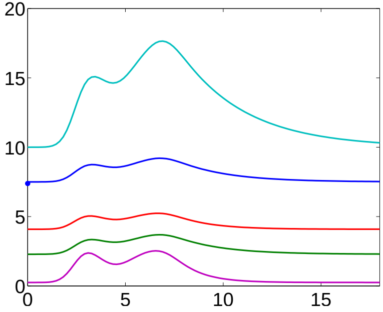
Under the linear model, we have

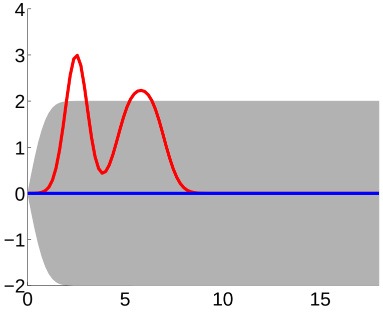
$$\begin{bmatrix} f \\ \mathbf{x} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ \frac{\mathbf{B}}{\mathbf{D}} \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{f\mathbf{x}} \\ K_{\mathbf{x}f} & K_{\mathbf{xx}} \end{bmatrix}\right)$$

Standard GP Regression yields the mean and covariance function of the predicted process as

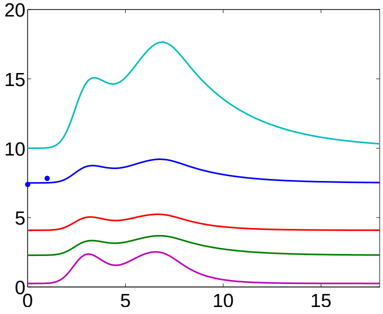
$$\begin{aligned} \langle f \rangle_{post} &= K_{f\mathbf{x}} K_{\mathbf{xx}}^{-1} \left(\mathbf{x} - \frac{\mathbf{B}}{\mathbf{D}} \right) \\ K_{ff}^{post} &= K_{ff} - K_{f\mathbf{x}} K_{\mathbf{xx}}^{-1} K_{\mathbf{x}f} \end{aligned}$$

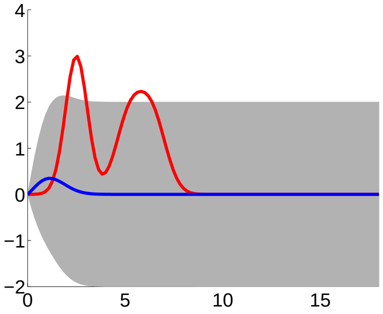
Artificial Example: Inferring $f(t)$



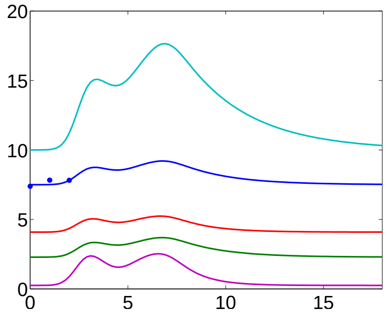


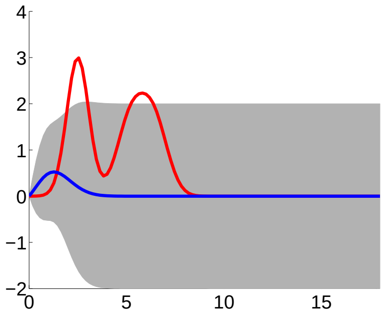
Artificial Example: Inferring $f(t)$



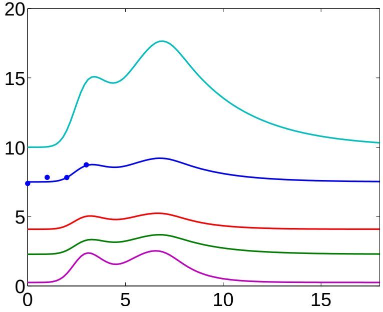


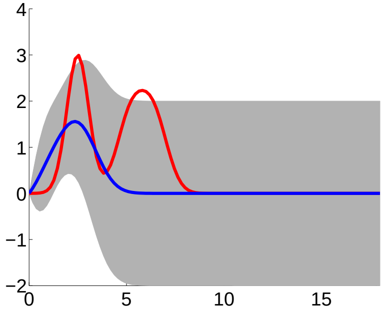
Artificial Example: Inferring $f(t)$



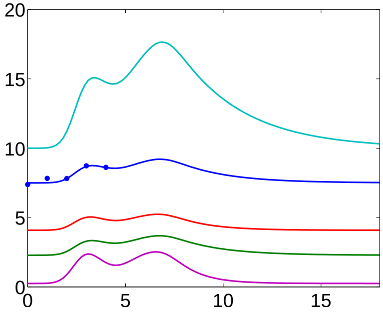


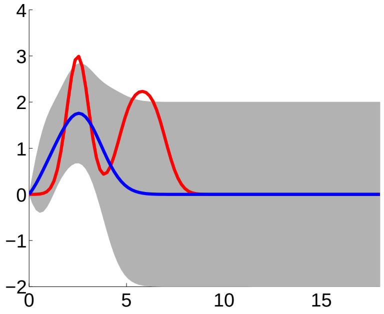
Artificial Example: Inferring $f(t)$



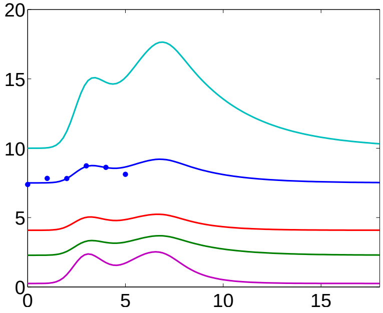


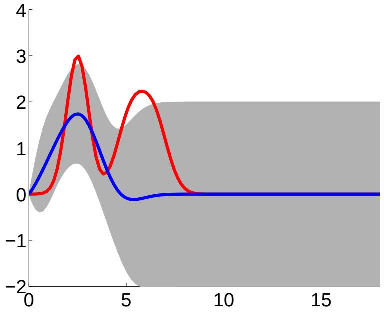
Artificial Example: Inferring $f(t)$



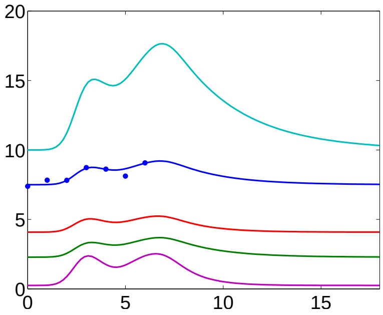


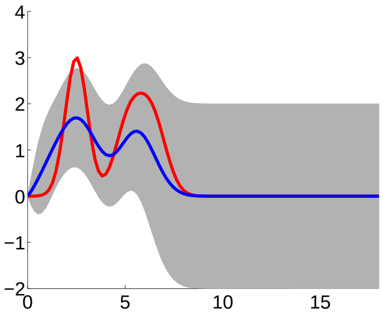
Artificial Example: Inferring $f(t)$



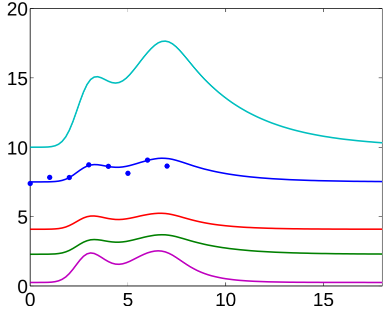


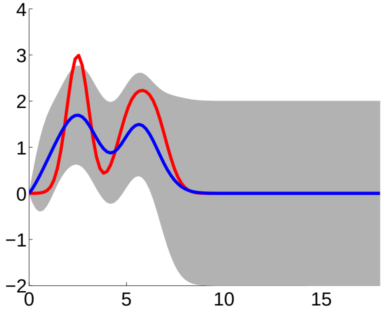
Artificial Example: Inferring $f(t)$



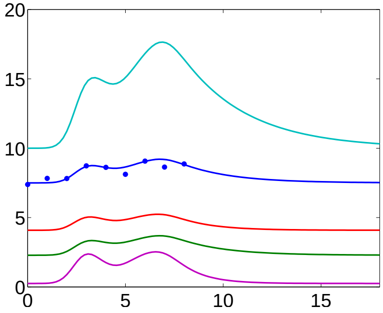


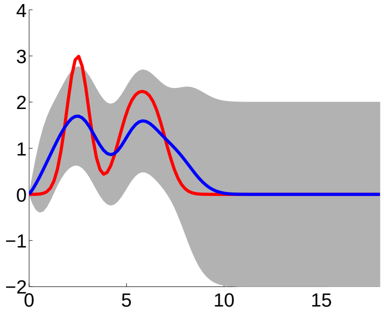
Artificial Example: Inferring $f(t)$



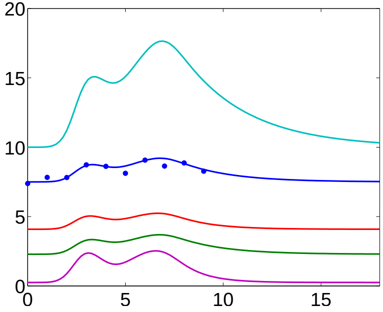


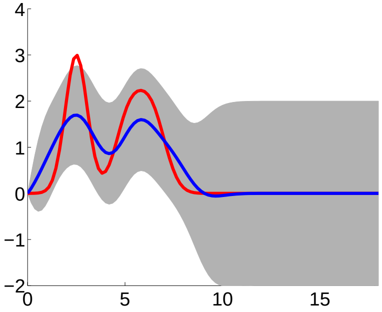
Artificial Example: Inferring $f(t)$



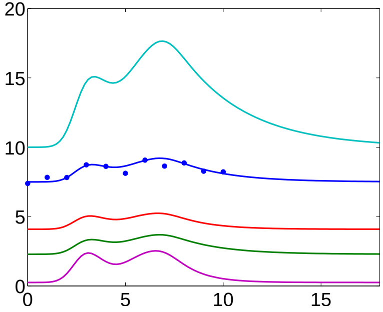


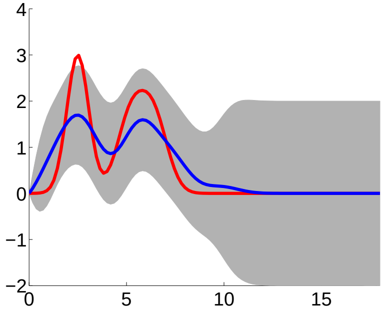
Artificial Example: Inferring $f(t)$



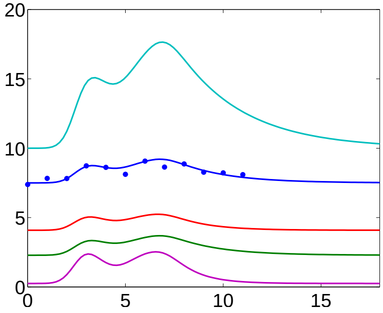


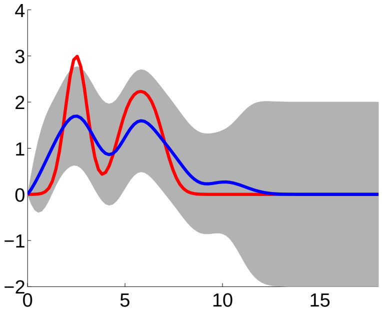
Artificial Example: Inferring $f(t)$



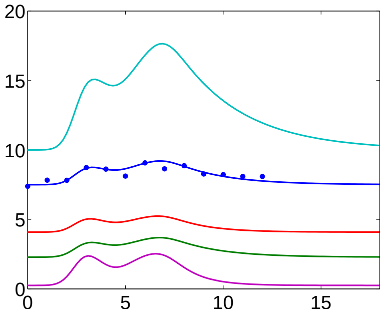


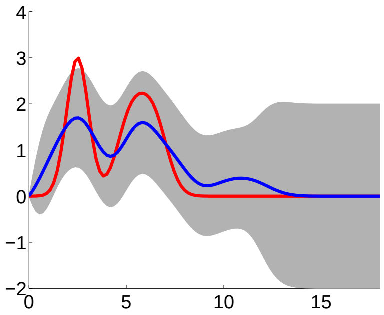
Artificial Example: Inferring $f(t)$



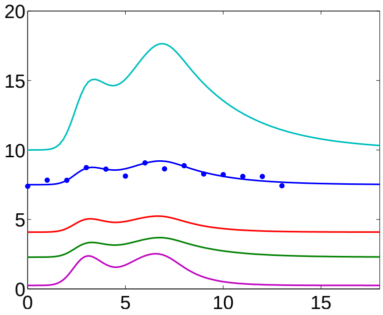


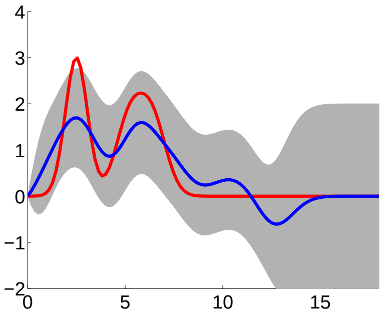
Artificial Example: Inferring $f(t)$



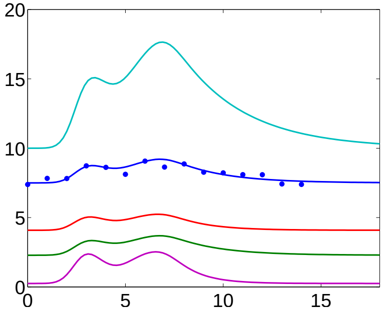


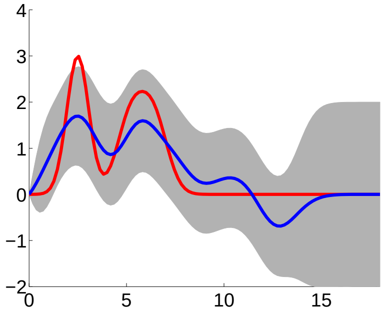
Artificial Example: Inferring $f(t)$



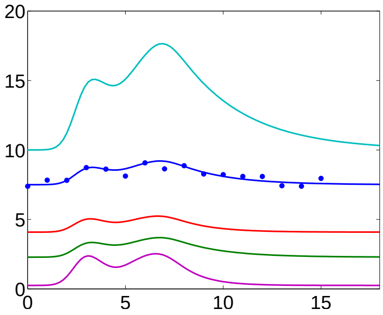


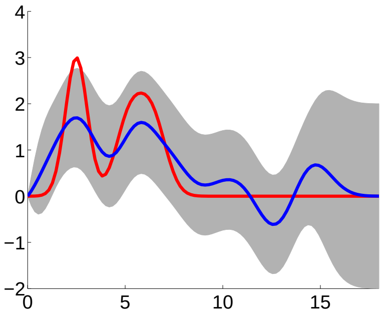
Artificial Example: Inferring $f(t)$



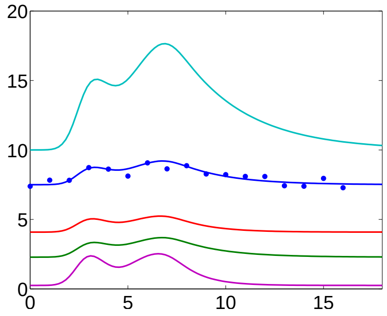


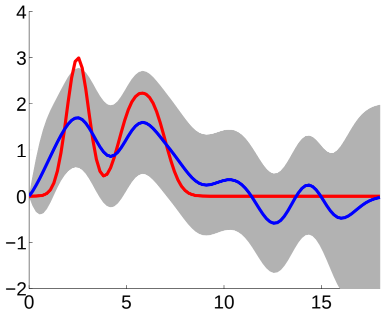
Artificial Example: Inferring $f(t)$



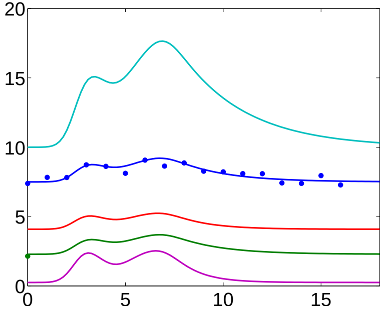


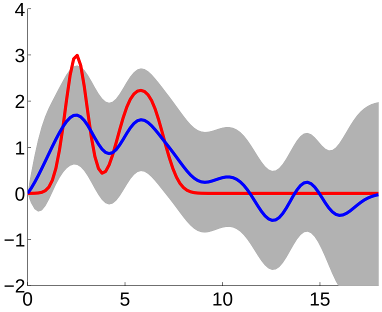
Artificial Example: Inferring $f(t)$



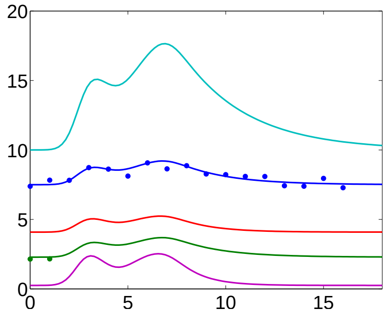


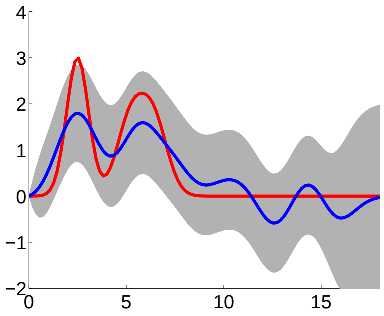
Artificial Example: Inferring $f(t)$



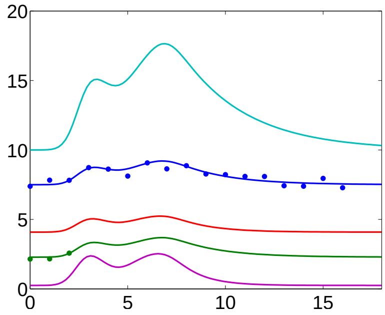


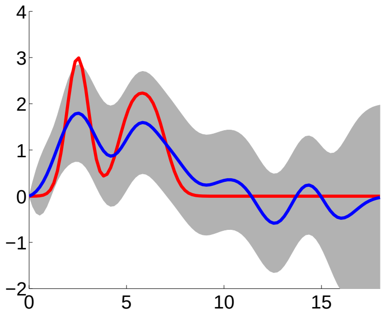
Artificial Example: Inferring $f(t)$



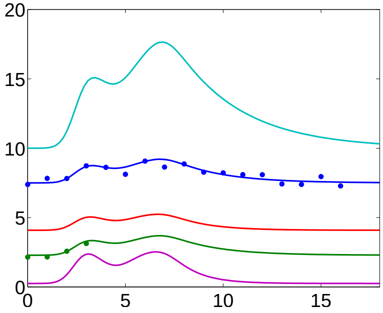


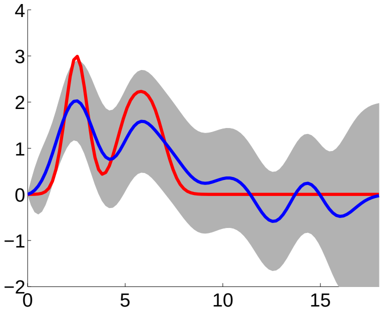
Artificial Example: Inferring $f(t)$



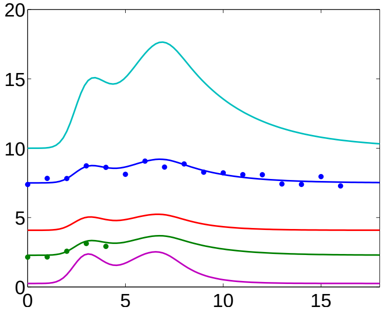


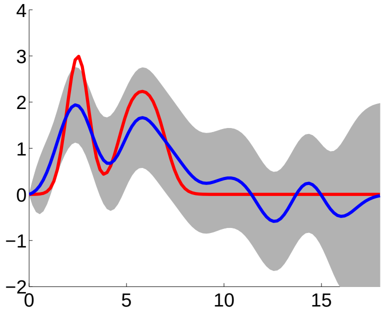
Artificial Example: Inferring $f(t)$



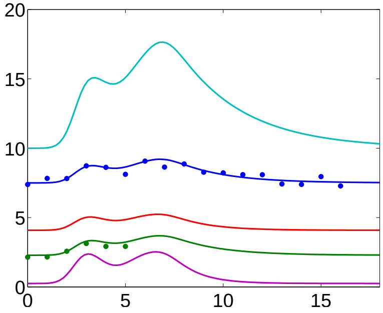


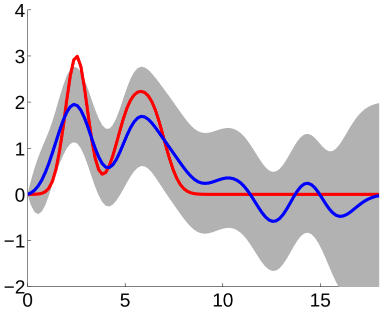
Artificial Example: Inferring $f(t)$



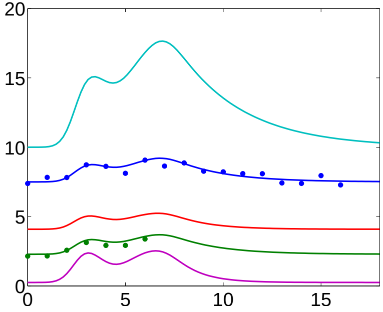


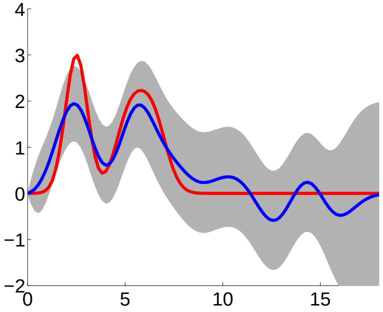
Artificial Example: Inferring $f(t)$



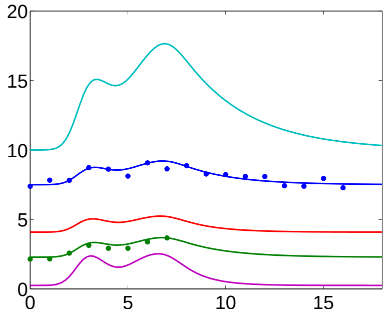


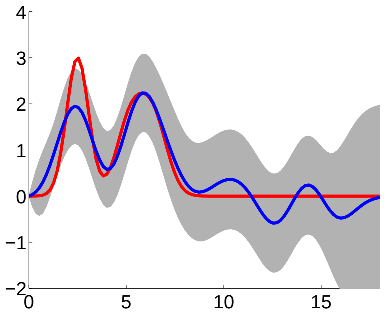
Artificial Example: Inferring $f(t)$



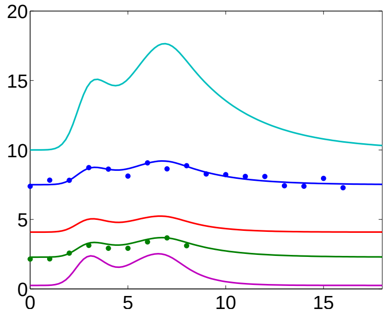


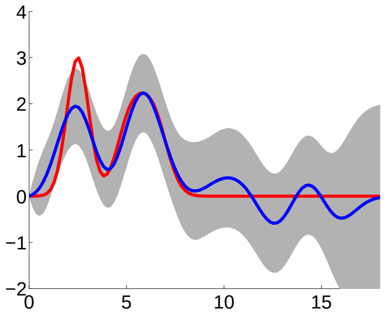
Artificial Example: Inferring $f(t)$



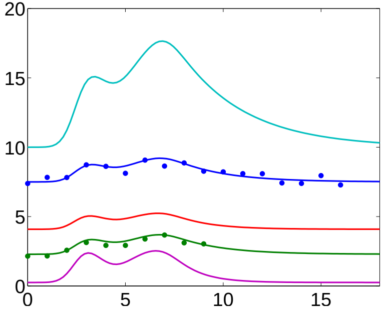


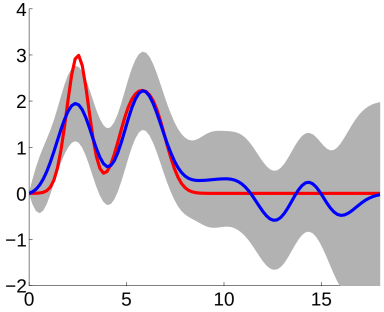
Artificial Example: Inferring $f(t)$



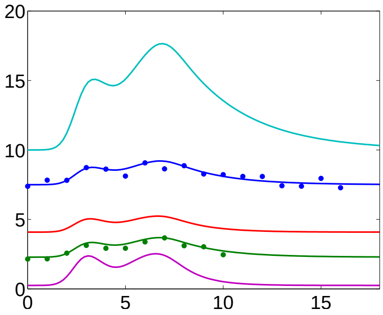


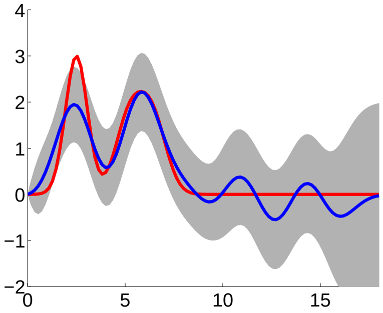
Artificial Example: Inferring $f(t)$



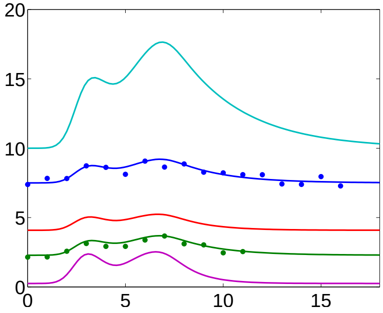


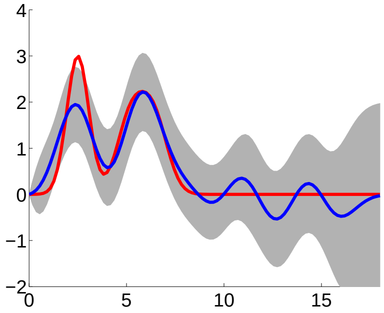
Artificial Example: Inferring $f(t)$



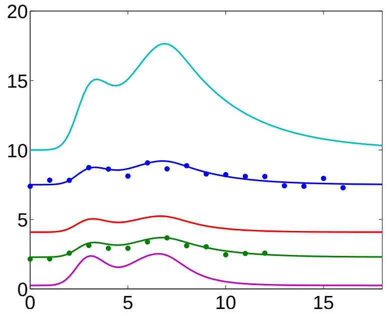


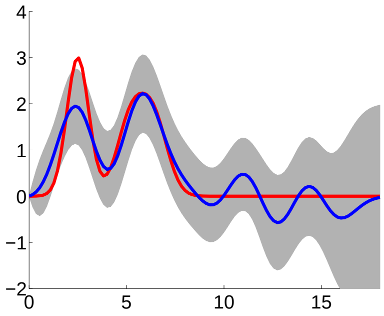
Artificial Example: Inferring $f(t)$



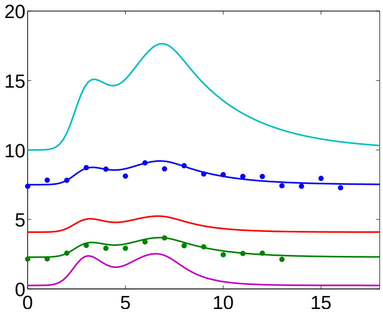


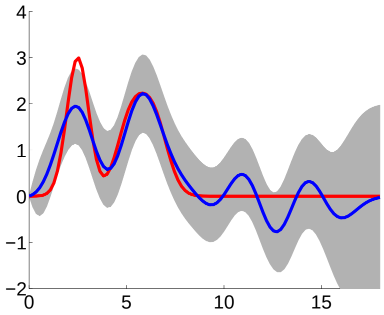
Artificial Example: Inferring $f(t)$



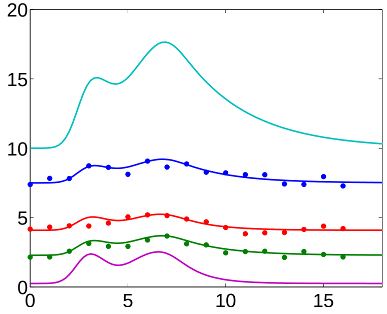


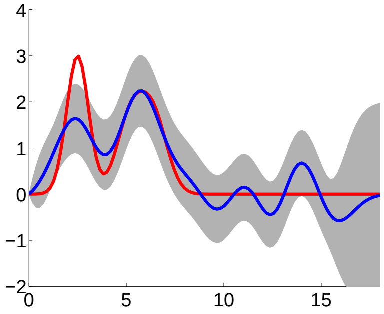
Artificial Example: Inferring $f(t)$



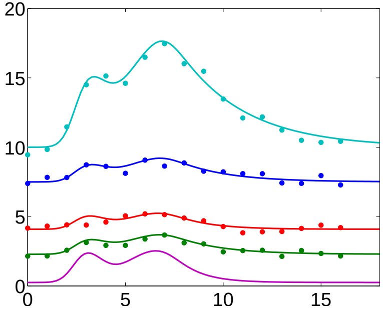


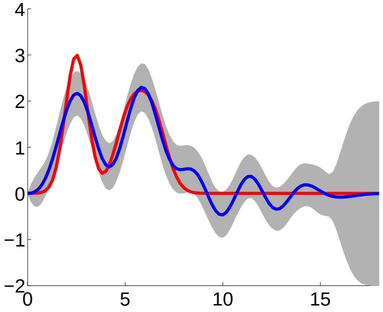
Artificial Example: Inferring $f(t)$



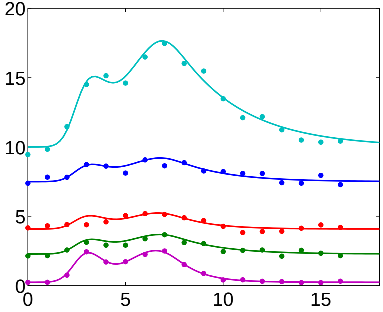


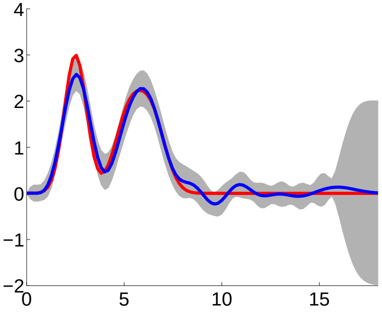
Artificial Example: Inferring $f(t)$





Artificial Example: Inferring $f(t)$





Parameter Estimation for the Linear Model

A likelihood function for the model parameters $\theta = \{B_j, S_j, D_j\}_{j=1}^N$ and GP length scale l is obtained by *integrating out* the latent function $f(t)$

$$L(\theta, l) = \int \left(\prod_j p(x_j|\theta, f(t)) \right) p(f(t)|l) \, \mathrm{d}f(t)$$

Under the GP model, the log marginal likelihood is then given by

$$\log L(\theta, l) = -\frac{1}{2} x^T \left(K + \sigma_n^2 \mathtt{I} \right)^{-1} x - \frac{1}{2} \log \left| K + \sigma_n^2 \mathtt{I} \right| - \frac{n}{2} \log 2\pi$$

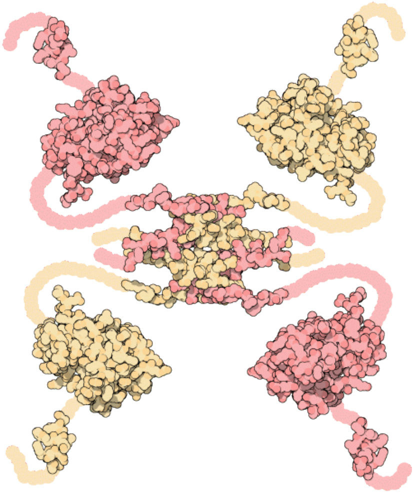
Cell Damage

- Radiation damages molecules in the cell.
- Most of this damage is quickly repaired — single strand breaks, backbone break.
- Double strand breaks are more serious — a complete disconnect along the chromosome.
- Cell cycle stages:
  - G_1: Cell is not dividing.
  - G_2: Cell is preparing for meitosis, chromosomes have divided.
  - S: Cell is undergoing meitosis (DNA synthesis).
- Main problem is in G1. In G2 there are two copies of the chromosome. In G1 only one copy.

p53 “Guardian of the Cell”

- Responsible for Repairing DNA damage
- Activates DNA Repair proteins
- Pauses the Cell Cycle (prevents replication of damage DNA)
- Initiates *apoptosis* (cell death) in the case where damage can't be repaired.
- Large scale feeback loop with NF-κB.

p53 DNA Damage Repair



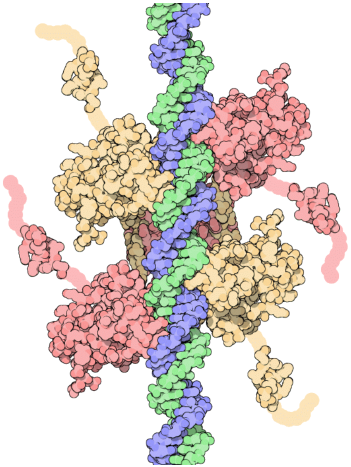
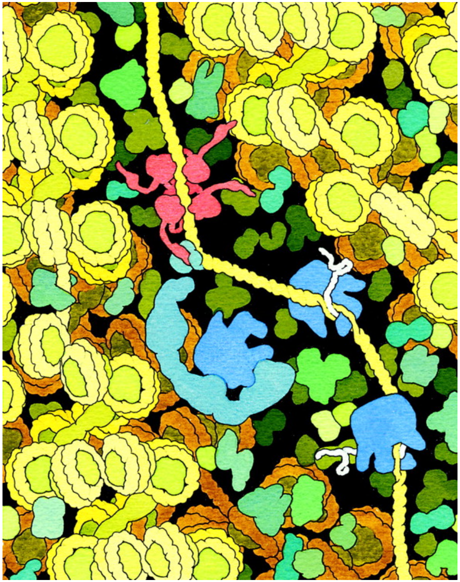


Figure: p53. *Left* unbound, *Right* bound to DNA. Images by David S. Goodsell from `http://www.rcsb.org/` (see the “Molecule of the Month” feature).

p53



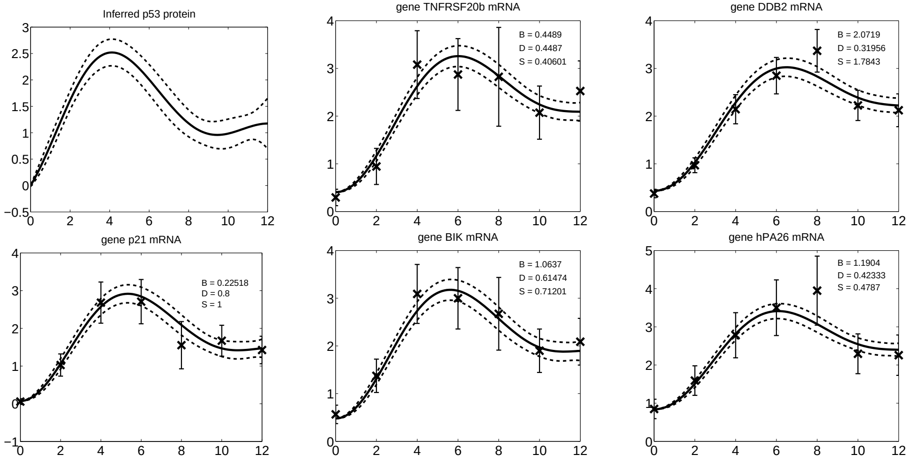
Some p53 Targets

DDB2	DNA Damage Specific DNA Binding Protein 2. (also governed by C/ EBP-beta, E2F1, E2F3,...).
p21	Cycline-dependent kinase inhibitor 1A (CDKN1A). A regulator of cell cycle progression. (also goverened by SREBP-1a, Sp1, Sp3,...).
hPA26/SESN1	sestrin 1 Cell Cycle arrest.
BIK:	BCL2-interacting killer. Induces cell death (apoptosis)
TNFRSF10b:	tumor necrosis factor receptor superfamily, member 10b. A transducer of apoptosis signals.

Data from Barenco et al. (2006). Microarray time course measuring gene expression after applying a dose of radiation to the system.

p53 (RBF covariance)

Pei Gao

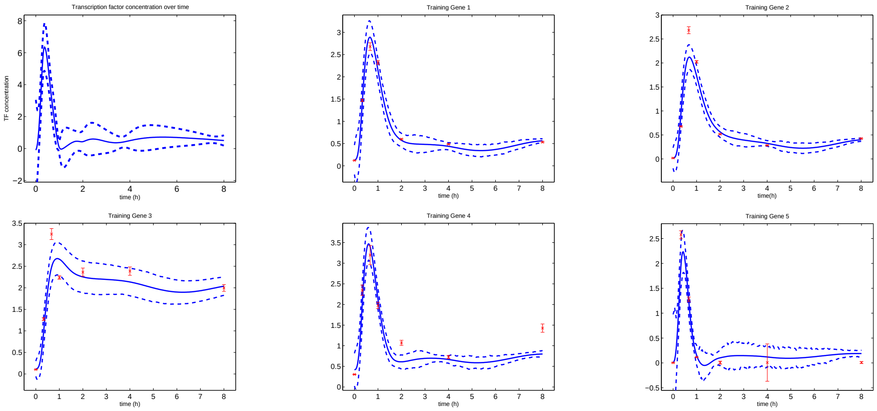


Ranking with ERK Signalling

- Target Ranking for Elk-1.

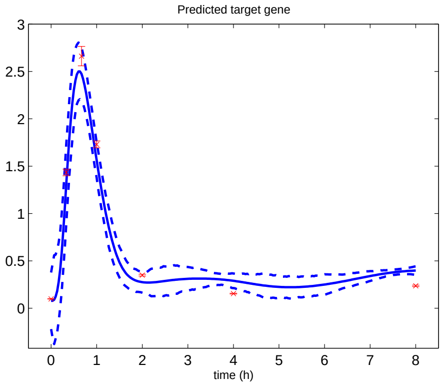
Elk-1 (MLP covariance)

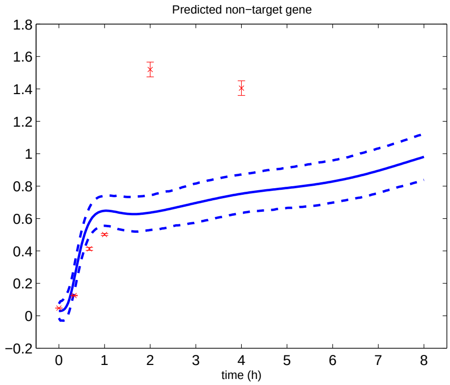
Jennifer Withers



Elk-1 target selection

Fitted model used to rank potential targets of Elk-1





Outline

Nonlinear Response Models

Consider the following modification to the model,

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j g(f(t)) - D_j x_j(t),$$

where $g(\cdot)$ is a non-linear function. The differential equation can still be solved,

$$x_j(t) = \frac{B_j}{D_j} + S_j \int_0^t e^{-D_j(t-u)} g_j(f(u))\,\mathrm{d}u$$

MAP-Laplace Approximation

Based on Laplace's method,

$$p(f \mid x) = N(\hat{f}, A^{-1}) \propto \exp\left(-\frac{1}{2}\left(f - \hat{f}\right)^T A \left(f - \hat{f}\right)\right)$$

where $\hat{f} = \mathrm{argmax} p(f \mid x)$ and $A = -\nabla\nabla \log p(f \mid y) \mid_{f=\hat{f}}$ is the Hessian of the negative posterior at that point.

To obtain $\hat{f}$ and A, we define the following function $\psi(f)$ as:

$$\log p(f|x) \propto \psi(f) = \log p(x \mid f) + \log p(f)$$

MAP-Laplace Approximation

Assigning a GP prior distribution to $f(t)$, it then follows that

$$\log p(f) = -\frac{1}{2} f^T K^{-1} f - \frac{1}{2}\log|K| - \frac{n}{2}\log 2\pi$$

where K is the covariance matrix of $f(t)$. Hence,

$$\begin{aligned}\nabla\psi(f) &= \nabla \log p(x|f) - K^{-1} f\\ \nabla\nabla\psi(f) &= \nabla\nabla \log p(x|f) - K^{-1} = -W - K^{-1}\end{aligned}$$

Estimation of $\psi(f)$

Newton's method is applied to find the maximum of $\psi(f)$ as

$$\begin{aligned} f^{new} &= f - (\nabla\nabla\psi(f))^{-1}\nabla\psi(f) \\ &= (W + K^{-1})^{-1}\,(Wf - \nabla \log p(x|f)) \end{aligned}$$

In addition, $A = -\nabla\nabla\psi(\hat{f}) = W + K^{-1}$ where W is the negative Hessian matrix. Hence, the Laplace approximation to the posterior is a Gaussian with mean $\hat{f}$ and covariance matrix A^{-1}as

$$p(f \mid x) \simeq N(\hat{f}, A^{-1}) = N(\hat{f}, (W + K^{-1})^{-1})$$

Model Parameter Estimation

The marginal likelihood is useful for estimating the model parameters θ and covariance parameters l

$$p(x|\theta, l) = \int p(x|f, \theta, l)p(f)df = \int \exp(\psi(f))df$$

Using Taylor expansion of $\psi(f)$,

$$\log p(x|\theta, l) = \log p(x|\hat{f}, \theta, l) - \frac{1}{2}f^T K^{-1} f - \frac{1}{2}\log|I + KW|$$

The parameters $\eta = \{\theta, l\}$ can be then estimated by using

$$\frac{\partial \log p(x|\eta)}{\partial \eta} = \frac{\partial \log p(x|\eta)}{\partial \eta}\ |_{\text{explicit}} + \frac{\partial \log p(x|\eta)}{\partial \hat{f}}\frac{\partial \hat{f}}{\partial \eta}$$

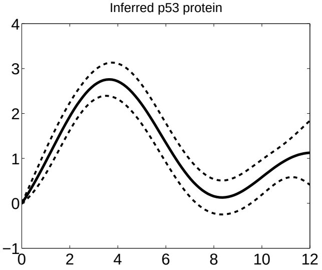
Michaelis-Menten Kinetics

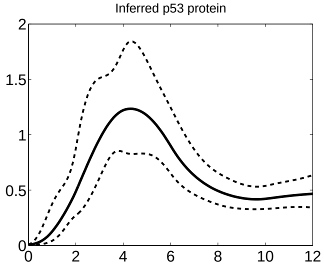
Pei Gao

- The Michaelis-Menten activation model uses the following non-linearity

$$g_j\left(f\left(t\right)\right) = \frac{e^{f(t)}}{\gamma_j + e^{f(t)}},$$

where we are using a GP $f\left(t\right)$ to model the log of the TF activity.





(a)

Repression Model

Pei Gao

- We can use an analogous model of repression,

$$g_j\left(f\left(t\right)\right) = \frac{1}{\gamma_j + e^{f\left(t\right)}}$$

In the case of repression we have to include the transient term,

$$x_j\left(t\right) = \alpha_j e^{-D_j t} + \frac{B_j}{D_j} + S_j \int_0^t e^{-D_j(t-u)} g_j(f\left(u\right))\mathrm{d}u$$

SOS Response

- Post replication DNA system: allows DNA replication to bypass errors in the DNA.
- DNA damage may occur as a result of activity of antibiotics.
- LexA is bound to the genome preventing transcription of the SOS genes.
- RecA protein is stimulated by single stranded DNA, inactivates the LexA repessor.
- This allows several of the LexA targets to transcribe.
- The SOS pathway may be essential in antibiotic resistance Cirz et al. (2005).
- Aim is to target these proteins to produce drugs to increase efficacy of antibiotics Lee et al. (2005).

LexA Experimental Description

- Data from Courcelle et al. (2001)
- UV irradiation of *E. coli.* in both wild-type cells and lexA1 mutants, which are unable to induce genes under LexA control.
- Response measured with two color hybridization to cDNA arrays.

Their Model

Given measurements of gene expression at N time points $(t_0, t_1, \ldots, t_{N-1})$, the temporal profile of a gene k, µk(t), that solves the ODE in Eq. 1 can be approximated by

$$\mu_k(t) = \mu_k^0 e^{-\delta_k t} + \frac{\alpha_k}{\delta_k} + \beta_k e^{-\delta_k t} \frac{1}{\delta_k} \sum_{j=0}^{N-2} \left(e^{\delta_k t_{j+1}} - e^{\delta_k t_j}\right) \frac{1}{\gamma_k + \bar{\eta}_j} \,, \qquad [2]$$

where $\bar{\eta}_j = \frac{(\eta(tj)+\eta(tj+1))}{2}$ on each subinterval $(tj, tj+1)$, $j = 0, \ldots, N-2$. This is under the simplifying assumption that $\eta(t)$ is a piece-wise constant function on each subinterval $(tj, tj+1)$. **One can come up with linear (or higher order) $\eta(t)$ approximations on each subinterval. This will introduce additional parameters, which will be impossible to infer with any certainty given limited amount of data.**

Khanin et al. (2006)

Their Results

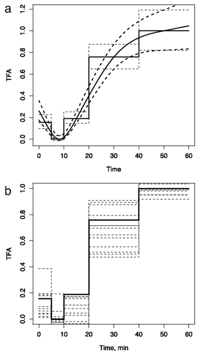


Figure: Fig. 2 from Khanin et al. (2006): Reconstructed activity level of master repressor LexA, following a UV dose of 40 J/m2.

Their Results

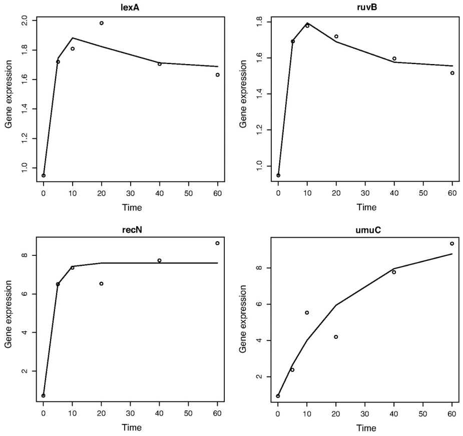


Figure: Fig. 3 from Khanin et al. (2006): Reconstructed profiles for four genes in the LexA SIM.

Results for the repressor LexA

Pei Gao

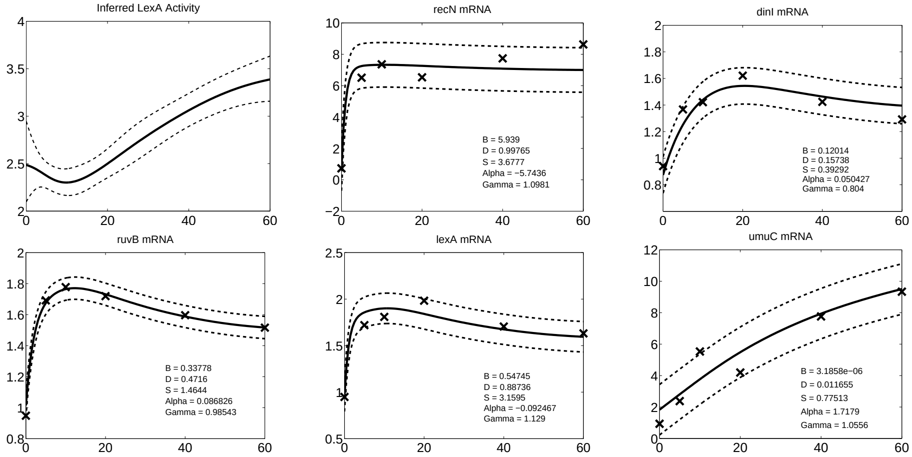


Figure: Our results using an MLP kernel. Presented at ECCB08 Gao et al. (2008).

Outline

Discussion and Future Work

- Integration of probabilistic inference with mechanistic models.
- These results are small simple systems.
- Ongoing work:
  - Scaling up to larger systems
  - Applications to other types of system, *e.g.* non-steady-state metabolomics, spatial systems etc.
  - Improved approximations.
  - Stochastic differential equations

Outline

Acknowledgements

- Investigators: Neil Lawrence and Magnus Rattray
- Researchers: Peo Gao, Antti Honkela, Michalis Titsias and Jennifer Withers
- Charles Girardot and Eileen Furlong of EMBL in Heidelberg (mesoderm development in *D. Melanogaster* see Appendix).
- Martino Barenco and Mike Hubank at the Institute of Child Health in UCL (p53 pathway).
- Raya Khanin and Ernst Wit of the University of Glasgow and the University of Lancaster (*E. coli* repressor system).

Funded by the BBSRC award "Improved Processing of microarray data using probabilistic models" and EPSRC award "Gaussian Processes for Systems Identification with applications in Systems Biology"

References I

M. Barenco, D. Tomescu, D. Brewer, R. Callard, J. Stark, and M. Hubank. Ranked prediction of p53 targets using hidden variable dynamic modeling. *Genome Biology*, 7(3):R25, 2006. [PDF].

R. T. Cirz, J. K. Chin, D. R. Andes, V. de CrÃ©cy-Lagard, W. A. Craig, and F. E. Romesberg. Inhibition of mutation and combating the evolution of antibiotic resistance. *PLoS Biology*, 3(6), 2005.

J. Courcelle, A. Khodursky, B. Peter, P. O. Brown, , and P. C. Hanawalt. Comparative gene expression profiles following UV exposure in wild-type and SOS-deficient *Escherichia coli*. *Genetics*, 158:41–64, 2001.

P. Gao, A. Honkela, M. Rattray, and N. D. Lawrence. Gaussian process modelling of latent chemical species: Applications to inferring transcription factor activities. *Bioinformatics*, 24:i70–i75, 2008. [PDF]. [DOI].

D. S. Goodsell. The molecular perspective: p53 tumor suppressor. *The Oncologist, Vol. 4, No. 2, 138-139, April 1999*, 4(2): 138–139, 1999.

R. Khanin, V. Viciotti, and E. Wit. Reconstructing repressor protein levels from expression of gene targets in *E. Coli*. *Proc. Natl. Acad. Sci. USA*, 103(49):18592–18596, 2006. [PDF]. [DOI].

A. M. Lee, C. T. Ross, B.-B. Zeng, , and S. F. Singleton. A molecular target for suppression of the evolution of antibiotic resistance: Inhibition of the *Escherichia coli* RecA protein by N6-(1-Naphthyl)-ADP. *J. Med. Chem.*, 48(17), 2005.

S. Rogers and M. Girolami. Model based identification of transcription factor regulatory activity via Markov chain Monte Carlo. Presentation at MASAMB '06, 2006.

C. K. I. Williams. Computing with infinite networks. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, Cambridge, MA, 1997. MIT Press.

MCMC for Non Linear Response

Michalis Titsias

The Metropolis-Hastings algorithm

- Initialize $\mathbf{f}^{(0)}$
- Form a Markov chain. Use a proposal distribution $Q(\mathbf{f}^{(t+1)}|\mathbf{f}^{(t)})$ and accept with the M-H step

$$\min\left(1, \frac{p(\mathbf{y}|\mathbf{f}^{(t+1)})p(\mathbf{f}^{(t+1)})}{p(\mathbf{y}|\mathbf{f}^{(t)})p(\mathbf{f}^{(t)})}\frac{Q(\mathbf{f}^{(t)}|\mathbf{f}^{(t+1)})}{Q(\mathbf{f}^{(t+1)}|\mathbf{f}^{(t)})}\right)$$

- $\mathbf{f}$ can be very *high dimensional* (hundreds of points)
- How do we choose the proposal $Q(\mathbf{f}^{(t+1)}|\mathbf{f}^{(t)})$?
  - Can we use the GP prior $p(\mathbf{f})$ as the proposal?

Sampling using control points

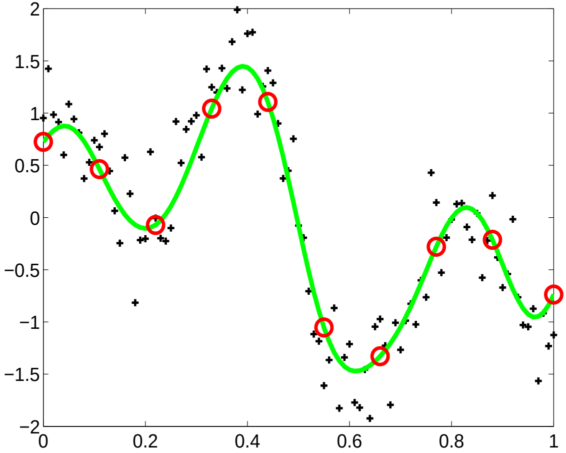
- Separate the points in $\mathbf{f}$ into two groups:
  - few control points $\mathbf{f}_c$
  - and the large majority of the remaining points $\mathbf{f}_\rho = \mathbf{f} \setminus \mathbf{f}_c$
- Sample the control points $\mathbf{f}_c$ using a proposal $q(\mathbf{f}_c^{(t+1)}|\mathbf{f}_c^{(t)})$
- Sample the remaining points $\mathbf{f}_\rho$ using the conditional GP prior $p(\mathbf{f}_\rho^{(t+1)}|\mathbf{f}_c^{(t+1)})$
- The whole proposal is

$$Q(\mathbf{f}^{(t+1)}|\mathbf{f}^{(t)}) = p(\mathbf{f}_\rho^{(t+1)}|\mathbf{f}_c^{(t+1)})q(\mathbf{f}_c^{(t+1)}|\mathbf{f}_c^{(t)})$$

- Its like sampling from the prior $p(\mathbf{f})$ but imposing random walk behaviour through the control points

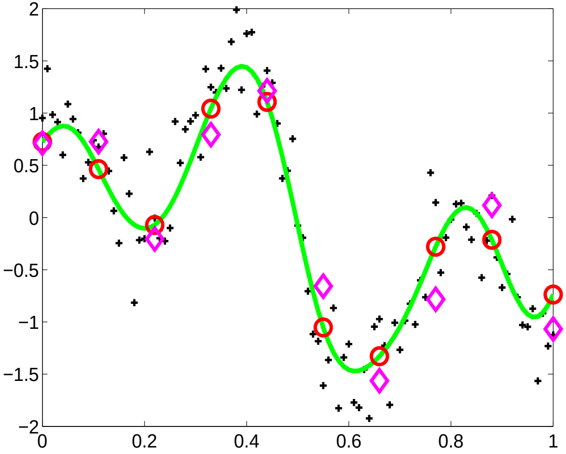
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



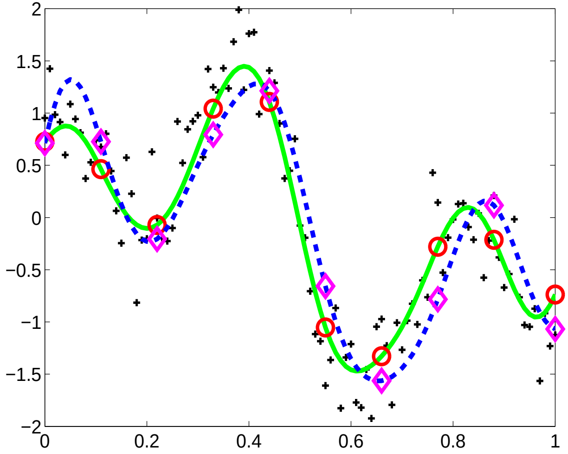
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



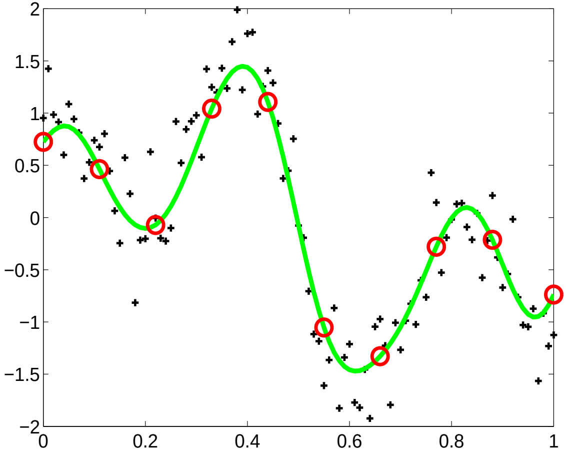
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



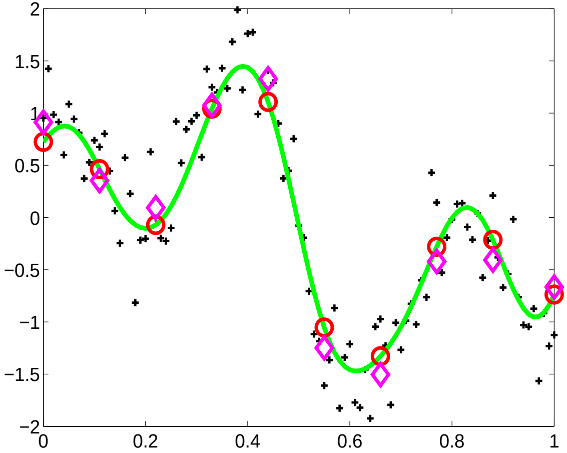
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



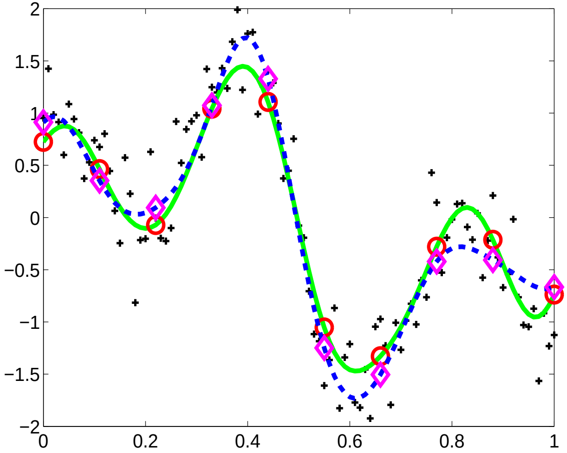
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



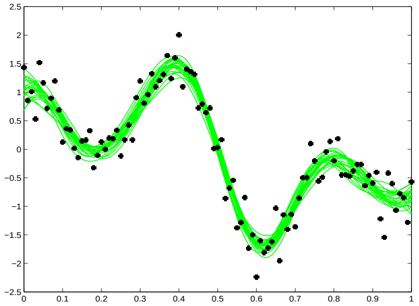
Sampling using control points: Regression-Examples

Sample 121 points using 10 control points



Sampling using control points

Few samples drawn during MCMC



Sampling using control points: Adaption of the proposal

Issues that need to be resolved during the burn in MCMC phase

- Number of control points
- Which points should be used as control points
- Improve the acceptance rate by
  - Adapting the variance of $q(\mathbf{f}_c^{(t+1)}|\mathbf{f}_c^{(t)})$ during the burn in period
  - Sampling the control points in a block-wise manner (keep some of them fixed when you sample others)

For the transcription factor modelling application there are natural choices for all the above issues. In the data we have considered so far we only need to adapt the variances of $q(\mathbf{f}_c^{(t+1)}|\mathbf{f}_c^{(t)})$

Transcriptional regulation using Gaussian processes

- Solve the equation

$$x_j(t) = \frac{B_j}{D_j} + A_j \exp(-D_j t) + S_j \exp(-D_j t) \int_0^t g(f(u)) \exp(D_j u) du$$

- Apply numerical integration using a very dense grid $(u_i)_{i=1}^P$ and $\mathbf{f} = (f_i(u_i))_{i=1}^P$

$$x_j(t) \simeq \frac{B_j}{D_j} + A_j \exp(-D_j t) + S_j \exp(-D_j t) \sum_{p=1}^{P_t} w_p g(f_p) \exp(D_j u_p)$$

  Assuming Gaussian noise for the observed gene expressions $\{x_{jt}\}$, the ODE defines the likelihood $p(\mathbf{x}|\mathbf{f})$

- Bayesian inference: Assume a GP prior for the transcription factor $\mathbf{f}$ and apply MCMC to infer $(\mathbf{f}, \{A_j, B_j, D_j, S_j\}_{j=1}^N)$
  - $\mathbf{f}$ is inferred in a continuous manner $(P \gg T)$

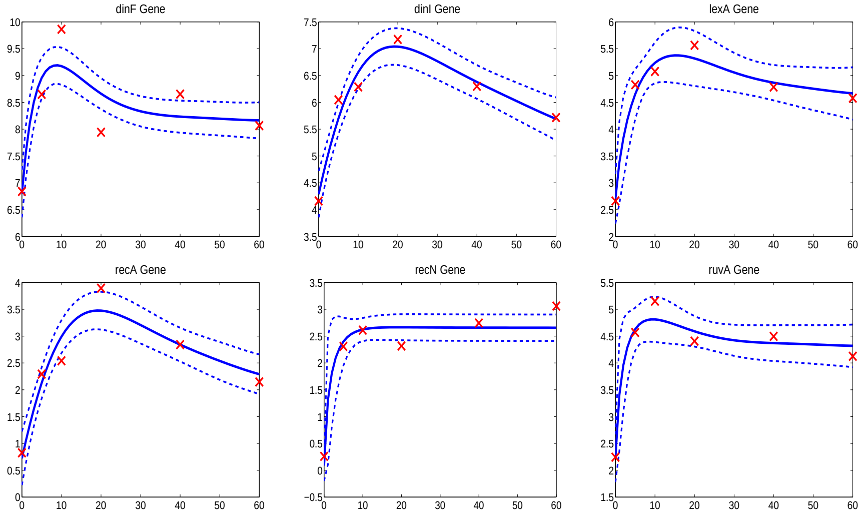
Results in E.coli data: Khanin et al. (2006)

- One transcription factor (lexA) that acts as a repressor. We consider the Michaelis-Menten kinetic equation

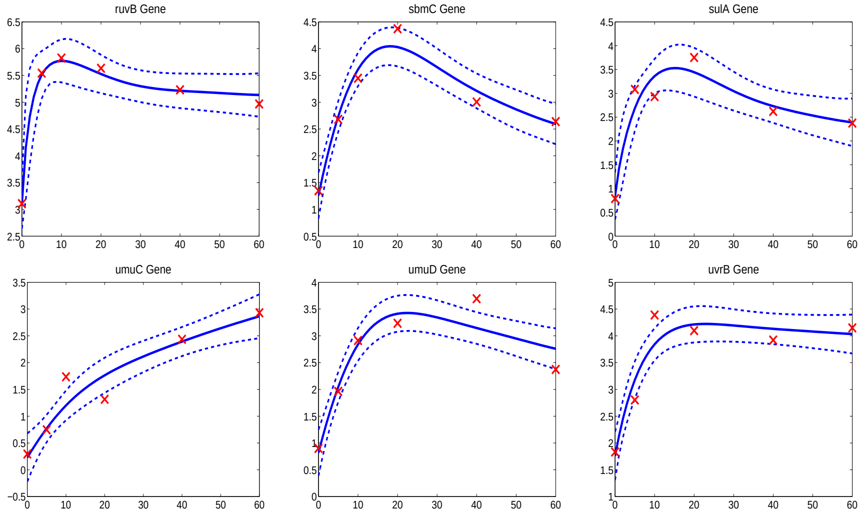
$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j \frac{1}{\exp(f(t)) + \gamma_j} - D_j x_j(t)$$

- We have 14 genes (5 kinetic parameters each)
- Gene expressions are available for $T = 6$ time slots
- TF ($\mathbf{f}$) is discretized using 121 points
- MCMC details:
  - 6 control points are used (placed in a equally spaced grid)
  - Running time was 5 hours for 2 million sampling iterations plus burn in
  - Acceptance rate for $\mathbf{f}$ after burn in was between $15\% - 25\%$

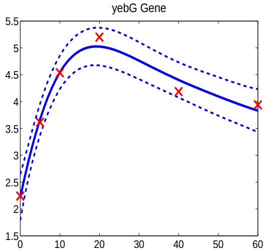
Results in E.coli data: Predicted gene expressions

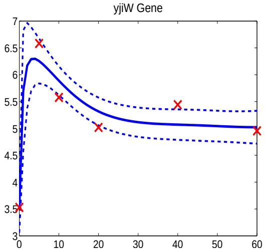


Results in E.coli data: Predicted gene expressions

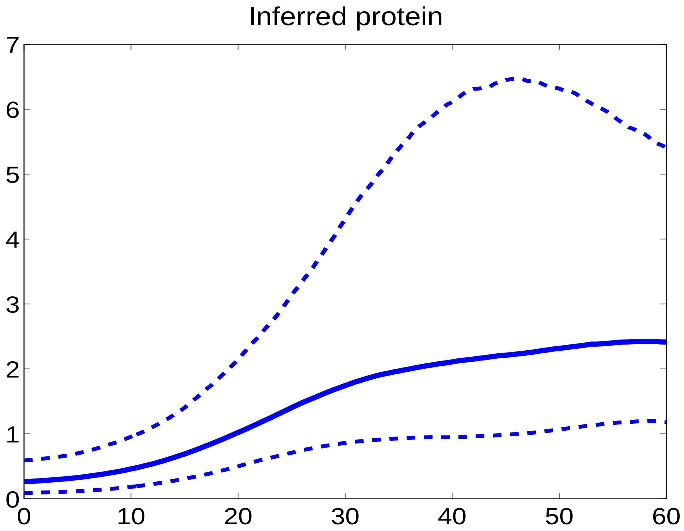


Results in E.coli data: Predicted gene expressions

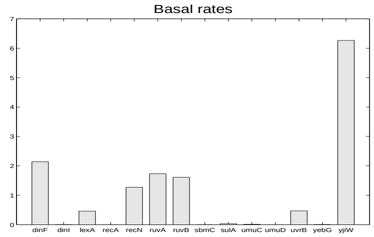


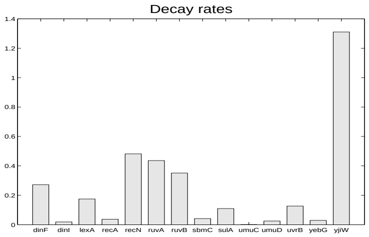


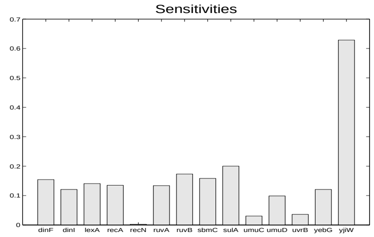
Results in E.coli data: Protein concentration

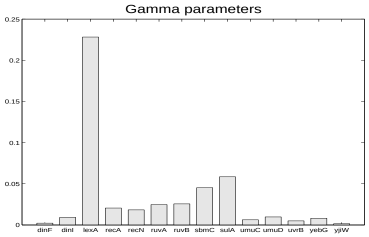


Results in E.coli data: Kinetic parameters

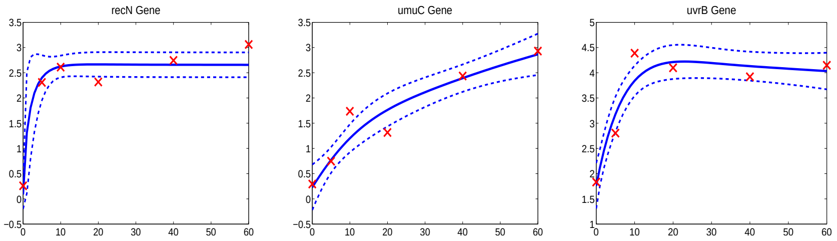


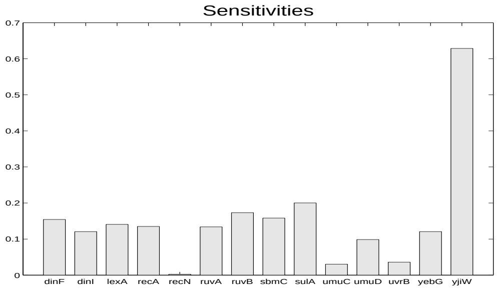




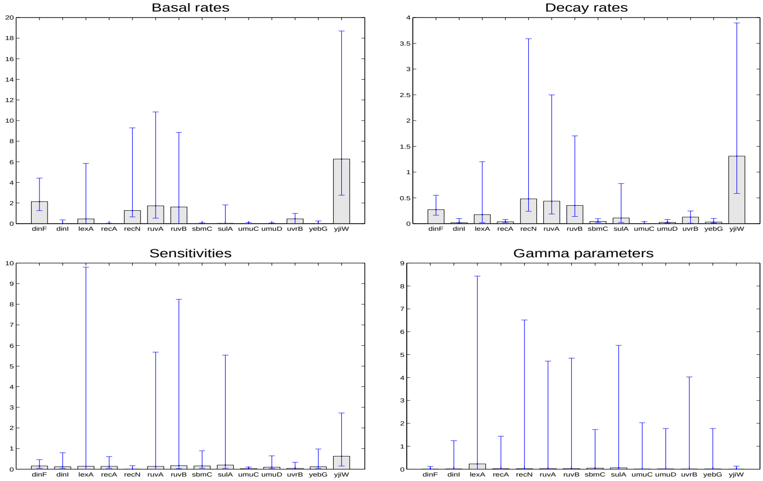


Results in E.coli data: Genes with low sensitivity value





Results in E.coli data: Confidence intervals for the kinetic parameters



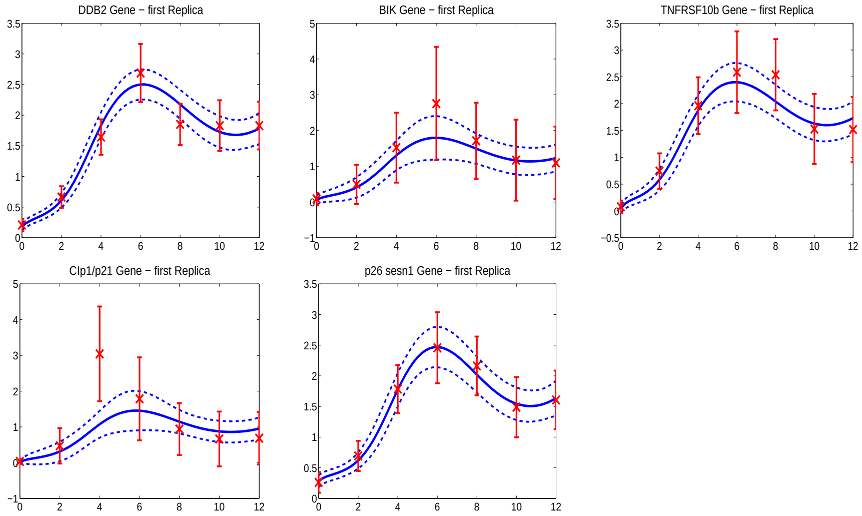
p53 System Again

- One transcription factor (p53) that acts as an activator. We consider the Michaelis-Menten kinetic equation

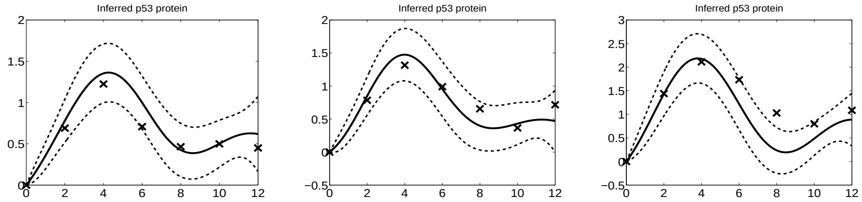
$$\frac{\mathrm{d}x_j(t)}{dt} = B_j + S_j \frac{\exp(f(t))}{\exp(f(t)) + \gamma_j} - D_j x_j(t)$$

- We have 5 genes
- Gene expressions are available for $T = 7$ times and there are 3 replicas of the time series data
- TF ($\mathbf{f}$) is discretized using 121 points
- MCMC details:
  - 7 control points are used (placed in a equally spaced grid)
  - Running time $4/5$ hours for 2 million sampling iterations plus burn in
  - Acceptance rate for $\mathbf{f}$ after burn in was between $15\% - 25\%$

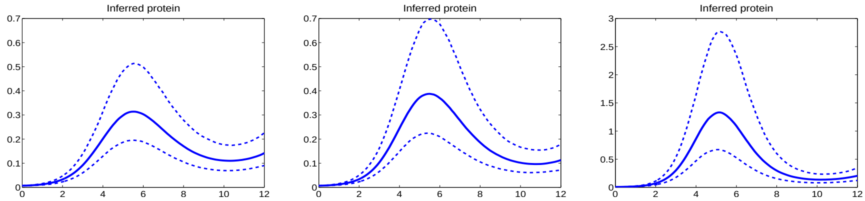
Data used by Barenco et al. (2006): Predicted gene expressions for the 1st replica



Data used by Barenco et al. (2006): Protein concentrations

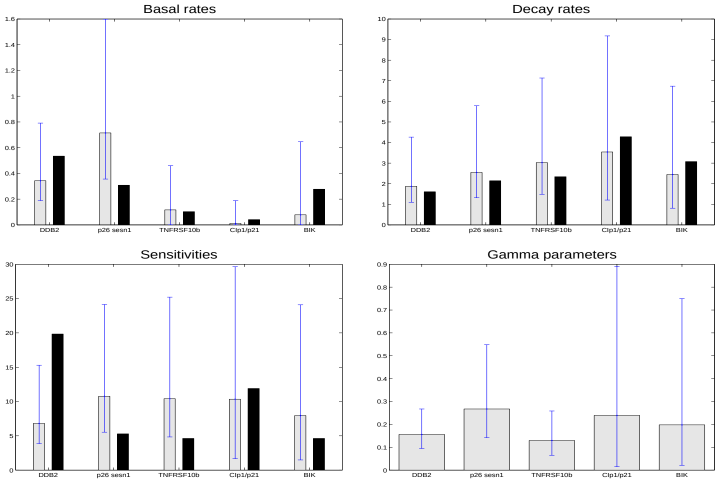


Linear model (Barenco et al. predictions are shown as crosses)



Nonlinear (Michaelis-Menten kinetic equation)

p53 Data Kinetic parameters



Our results (grey) compared with Barenco et al. (2006) (black). Note that Barenco et al. use a linear model

Outline

Outline

Cascaded Differential Equations

Antti Honkela

- Transcription factor protein also has governing mRNA.
- This mRNA can be measured.
- In signalling systems this measurement can be misleading because it is activated (phosphorylated) transcription factor that counts.
- In development phosphorylation plays less of a role.

Drosophila *Mesoderm* Development

Data from Furlong Lab in Heidelberg.

- Describe mesoderm development.

Cascaded Differential Equations

Antti Honkela

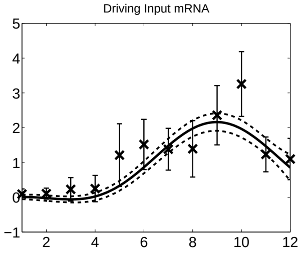
We take the production rate of active transcription factor to be given by

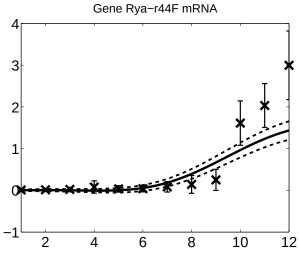
$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = \sigma y(t) - \delta f(t)$$
$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = B_j + S_j f(t) - D_j x_j(t)$$

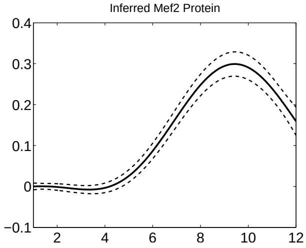
The solution for $f(t)$, setting transient terms to zero, is

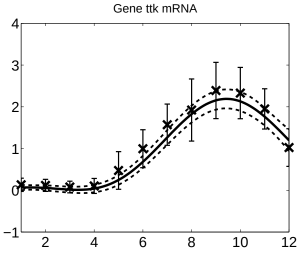
$$f(t) = \sigma \int_0^t y(v)\,\mathrm{e}^{\delta(v-t)}\mathrm{d}v\ .$$

Results for Mef2 using the Cascade model


Driving Input mRNA


Gene Rya−r44F mRNA


Inferred Mef2 Protein


Gene ttk mRNA