Modelling Transcriptional Regulation with Gaussian Processes

Neil Lawrence
School of Computer Science
University of Manchester
Joint work with **Magnus Rattray** and **Guido Sanguinetti**

28th March 2007

Outline

1. Application
   - Methodology Overview
2. Linear Response Model
   - Covariance functions
   - Regression with Gaussian Processes
   - Toy Problem
   - Biological Problem
3. Non-linear Response Model
   - Linear Response with MLP Kernel
   - Non-linear Responses

Online Resources

All source code and slides are available online

- This talk available from home page (see talks link on side).
- Scripts available in the 'gpsim' toolbox
  - `http://www.cs.man.ac.uk/~neill/gpsim/`.
- MATLAB commands used for examples given in `typewriter font`.

Framework

Latent functions

- Many interaction networks have *latent functions*.
- Assume a Gaussian process (GP) prior distribution for the latent function.
  - Gaussian processes (GPs) are probabilistic models for functions. O'Hagan [1978, 1992], Rasmussen and Williams [2006]
- Our Approach
  1. Take a differential equation model for the system.
  2. Derive GP covariance jointly for observed and latent functions.
  3. Maximise likelihood with respect to parameters (mostly physically meaningful).

This Talk

Transcription Network

- Introduce Gaussian Processes for dealing with *latent functions* in transcription networks.
- Show how in a linear response model the latent function can be dealt with *analytically*.
- Discuss extensions to systems with non-linear responses.

Linear Response Model

p53 Inference [Barenco et al., 2006]

- Data consists of T measurements of mRNA expression level for N different genes.
- We relate gene expression, $x_j(t)$, to TFC, $f(t)$, by

$$\frac{dx_j(t)}{dt} = B_j + S_j f(t) - D_j x_j(t). \quad (1)$$

B_j basal transcription rate of gene j,
S_j is sensitivity of gene j
D_j is the decay rate of the mRNA.

- Dependence of mRNA transcription rate on TF is linear.

Linear Response Solution

Solve for TFC

- The equation given in (1) can be solved to recover

$$x_j(t) = \frac{B_j}{D_j} + S_j \exp(-D_j t) \int_0^t f(u) \exp(D_j u)\, du. \quad (2)$$

- If we model $f(t)$ as a GP then as (2) only involves *linear operations* $x_j(t)$ is also a GP.

Gaussian Processes

GP Advantages

- GPs allow for inference of continuous profies, accounting naturally for temporal structure.
  - GPs allow joint estimation of a mRNA concentration and production rates (derivative observations).
  - GPs deal consistently with the uncertainty inherent in the measurements.
  - GPs outstrip MCMC for computational efficiency.

Note: GPs have previously been proposed for solving differential equations [Graepel, 2003] and dynamical systems [Murray-Smith and Pearlmutter].

Gaussian Processes

- Gaussian process is governed by a mean *function* and a covariance *function*

$$p(\mathbf{f}) = N(\mathbf{f}|\bar{\mathbf{f}}, \mathbf{K})$$

$$k_{ij} = k(t_i, t_j), \quad \bar{f}_i = \bar{f}(t_i)$$

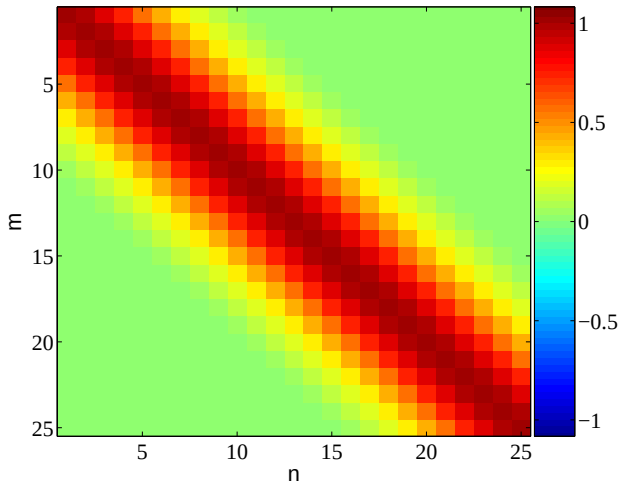
- Mean function often taken to be zero.
- Covariance function is any positive definite function, *e.g. RBF Covariance.*

Covariance Functions
Visualisation of RBF Covariance

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 time *inputs* to the function, t.
- For the example above it was based on Euclidean distance.
- The covariance function is also known as a kernel.



Covariance Samples

`demCovFuncSample` – sample from the prior

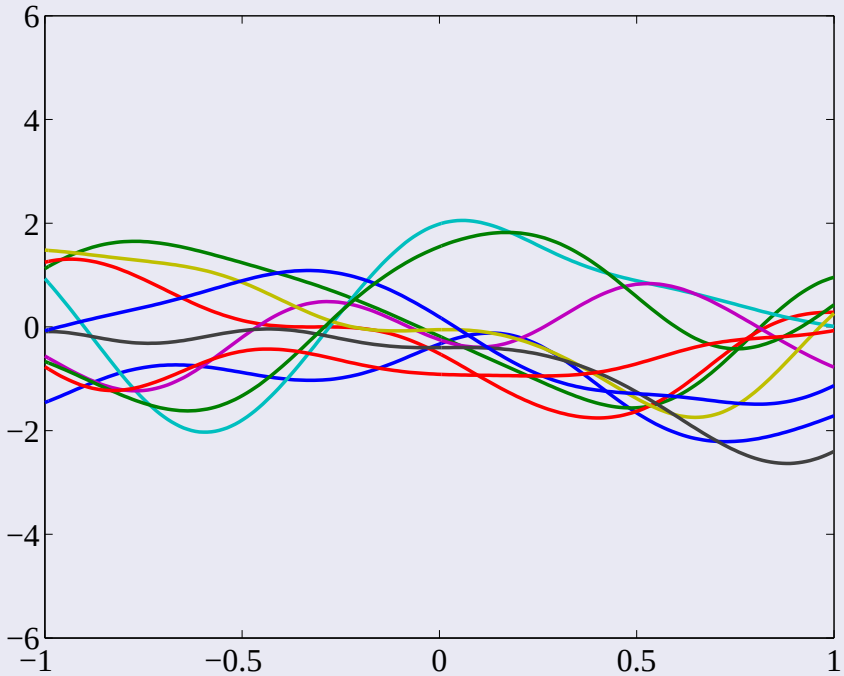


Figure: RBF kernel with $\gamma = 10$, $\alpha = 1$

Covariance Samples

demCovFuncSample – sample from the prior

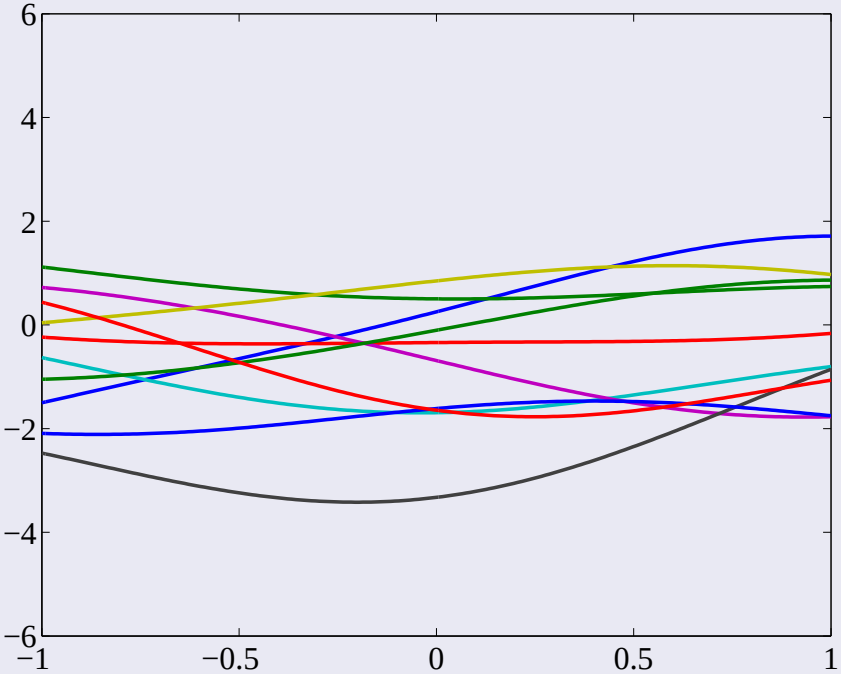


Figure: RBF kernel with $l = 1$, $\alpha = 1$

Covariance Samples

`demCovFuncSample` – sample from the prior

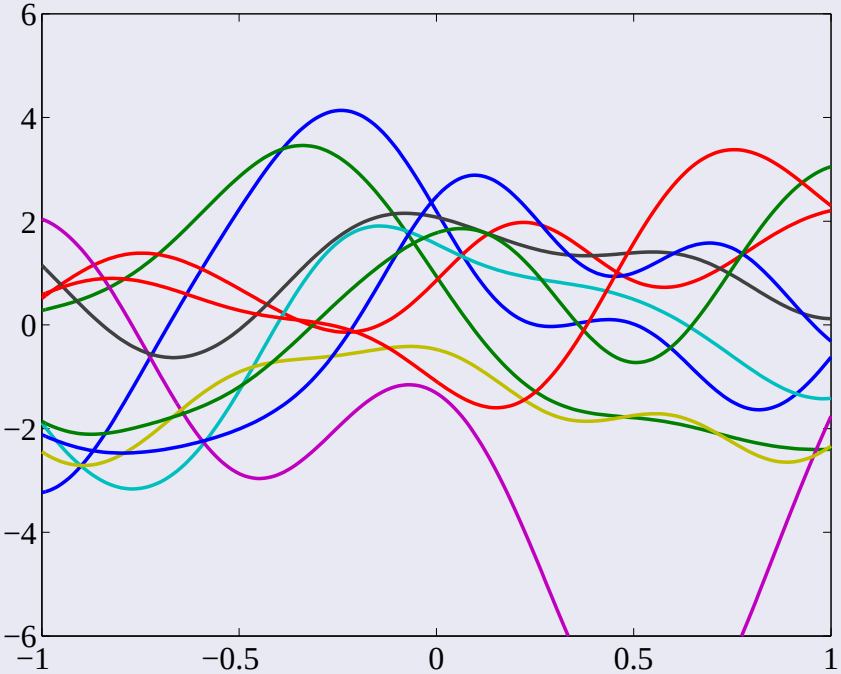


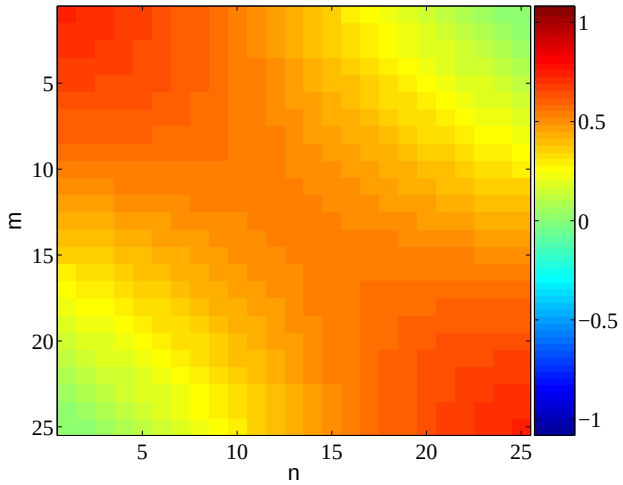
Figure: RBF kernel with $l = 0.3$, $\alpha = 4$

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 — samples from the prior

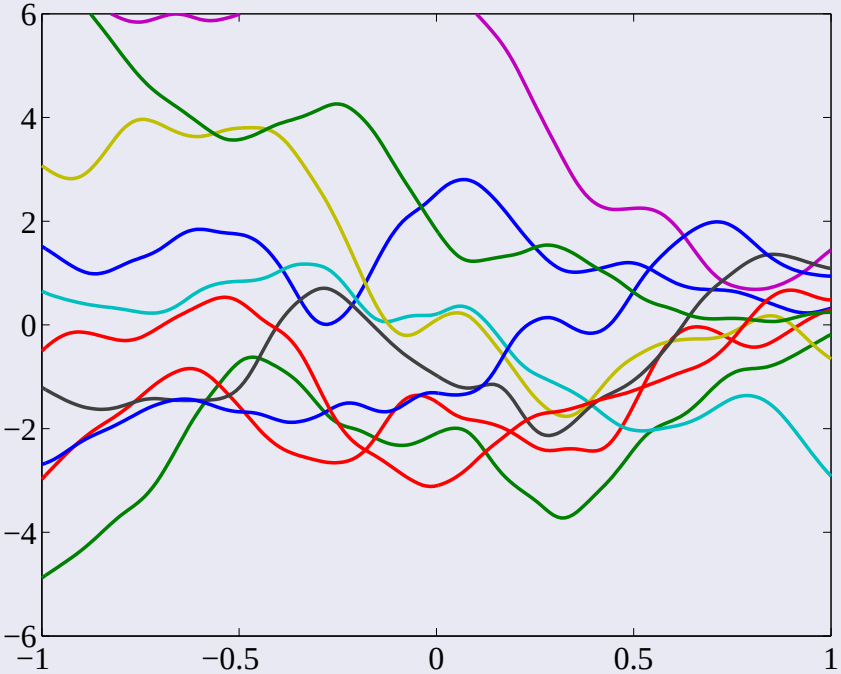


Figure: MLP kernel with $\alpha = 8$, $w = 100$ and $b = 100$

Covariance Samples

demCovFuncSample — samples from the prior

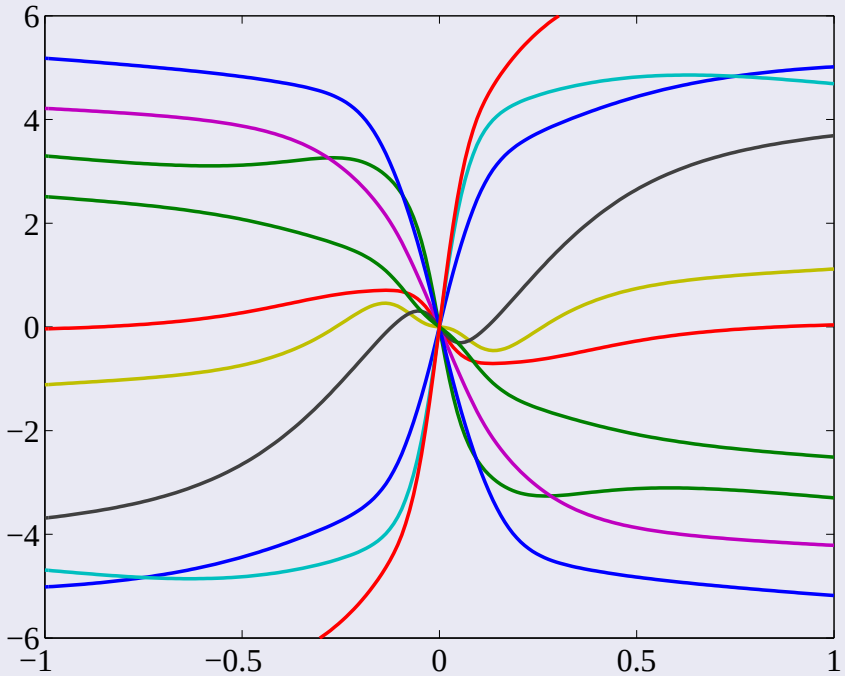


Figure: MLP kernel with $\alpha = 8$, $b = 0$ and $w = 100$

Prior to Posterior

Prediction with GPs

- GPs provide a probabilistic prior over functions.
- By combining with data we get a *posterior* over functions.
- This is obtained through combining a covariance function with data.
- Toy Example: regression with GPs.

Gaussian Process Regression

demRegression

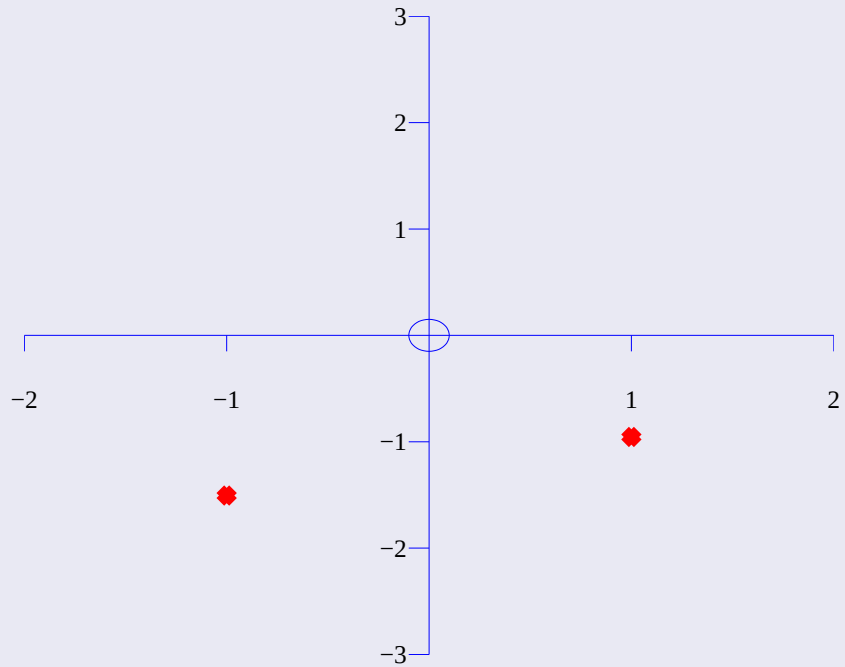


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

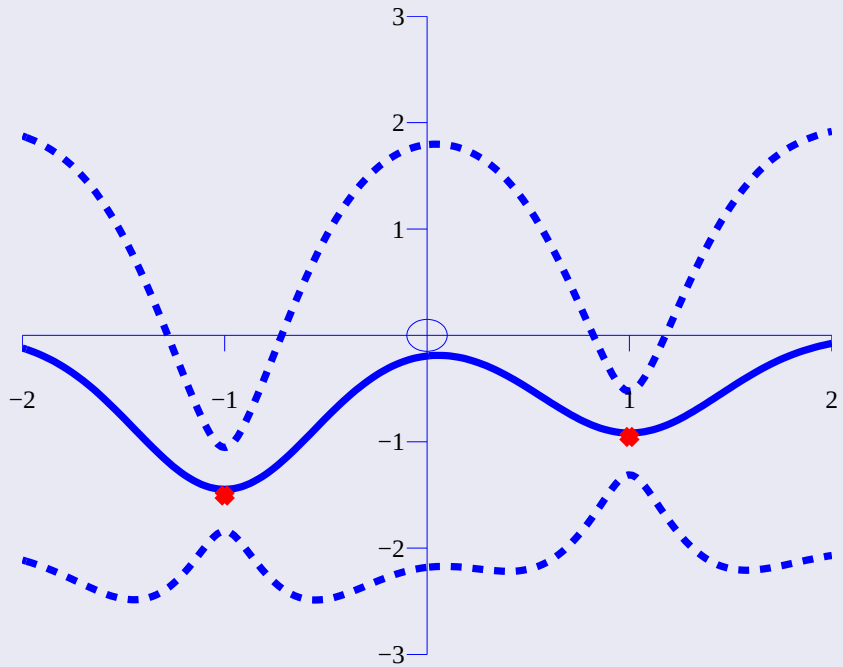


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

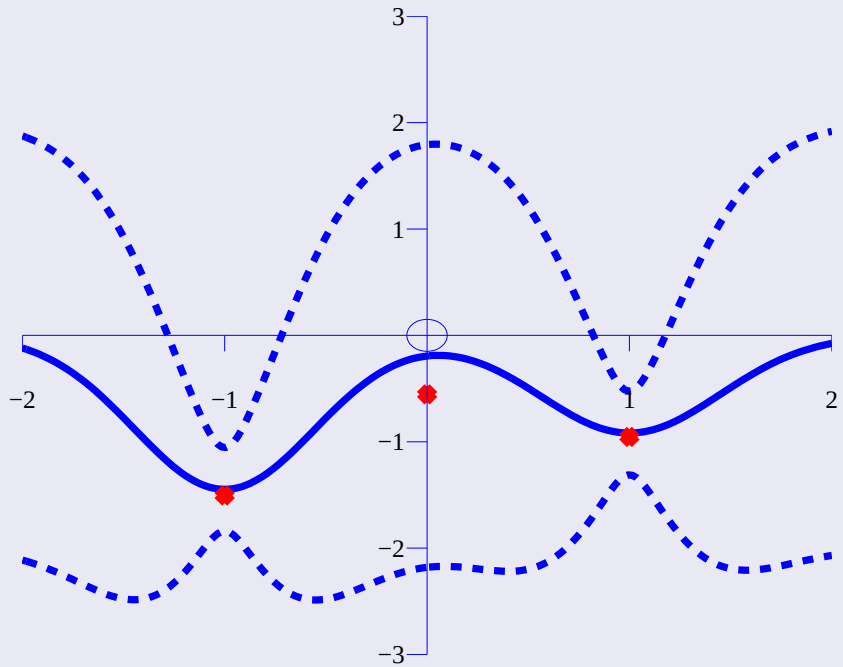


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

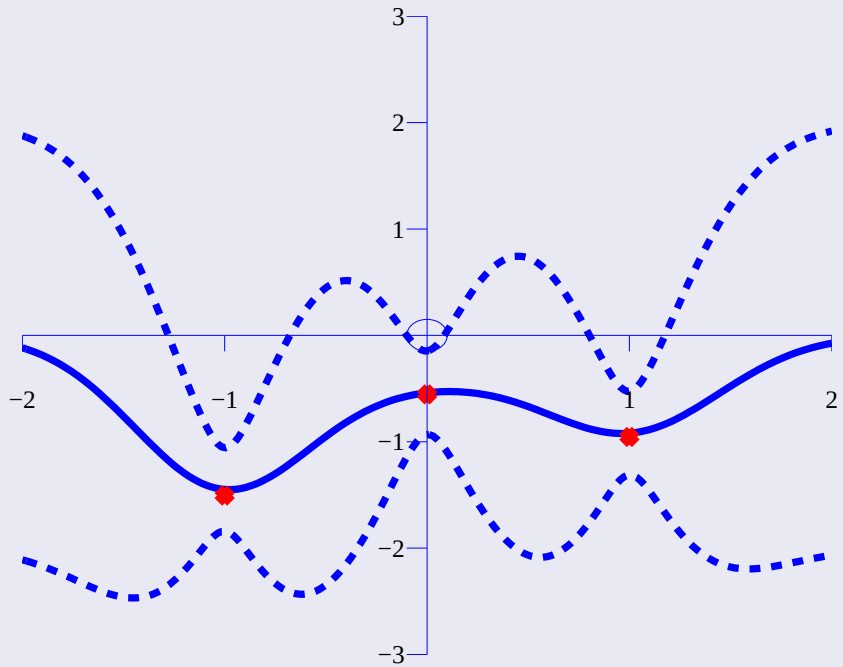


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

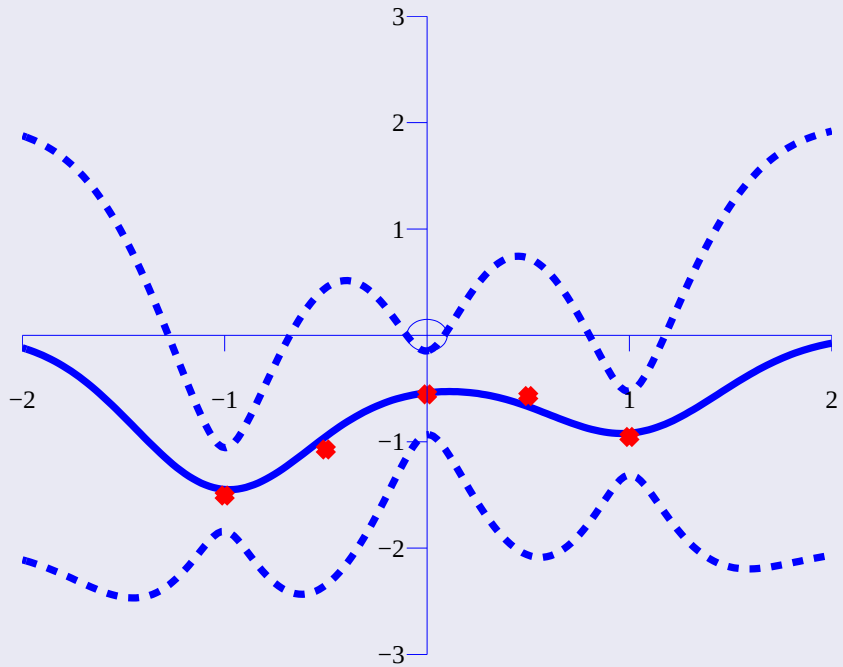


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

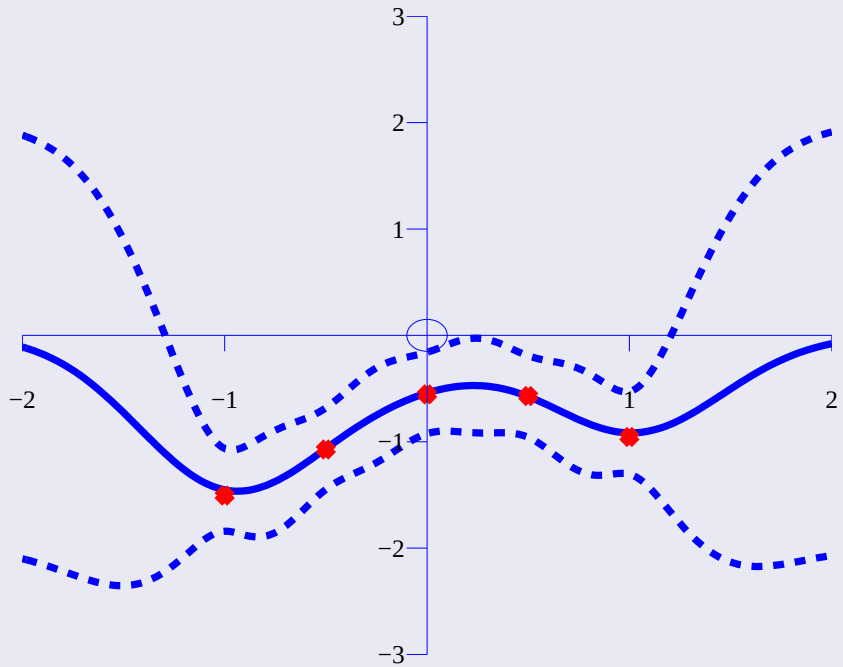


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

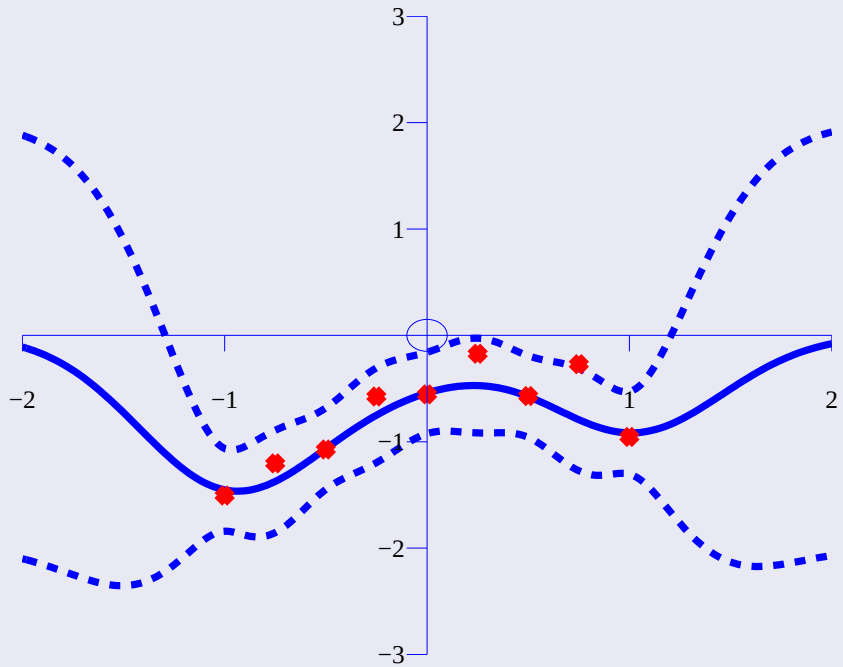


Figure: Going from prior to posterior with data.

Gaussian Process Regression

demRegression

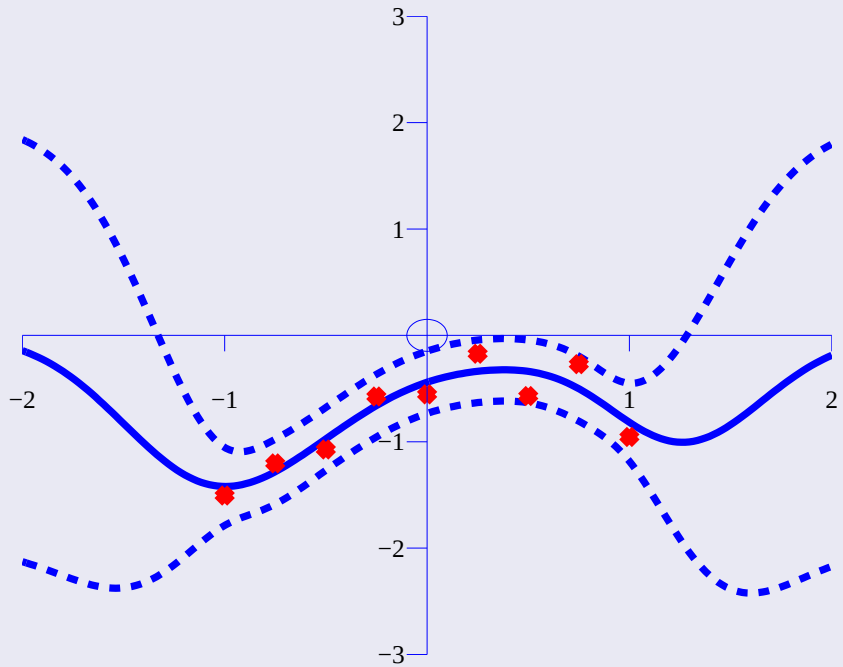


Figure: Going from prior to posterior with data.

Covariance of Latent Function

Prior Distribution for TFC

- We assume that the TF concentration is a Gaussian Process.
- We will assume an RBF covariance function

$$p(\mathbf{f}) = N(\mathbf{f}|\mathbf{0}, \mathbf{K}) \quad k(t, t') = \exp\left(-\frac{(t-t')}{2l^2}\right).$$

Computation of Joint Covariance

Covariance Function Computation

- We rewrite solution of differential equation as

$$x_j(t) = \frac{B_j}{D_j} + L_j[f](t)$$

where

$$L_j[f](t) = S_j \exp(-D_j t) \int_0^t f(u) \exp(D_j u)\, du \qquad (3)$$

is a linear operator.

Induced Covariance

Gene's Covariance

- The new covariance function is then given by

$$\text{cov}\left(L_j\left[f\right](t), L_k\left[f\right](t')\right) = L_j \otimes L_k\left[k_{ff}\right](t, t').$$

more explicitly

$$k_{x_j x_k}(t, t') = S_j S_k \exp\left(-D_j t - D_k t'\right) \int_0^t \exp\left(D_j u\right) \times \int_0^{t'} \exp\left(D_k u'\right) k_{ff}(u, u')\, du'\, du.$$

- With RBF covariance these integrals are tractable.

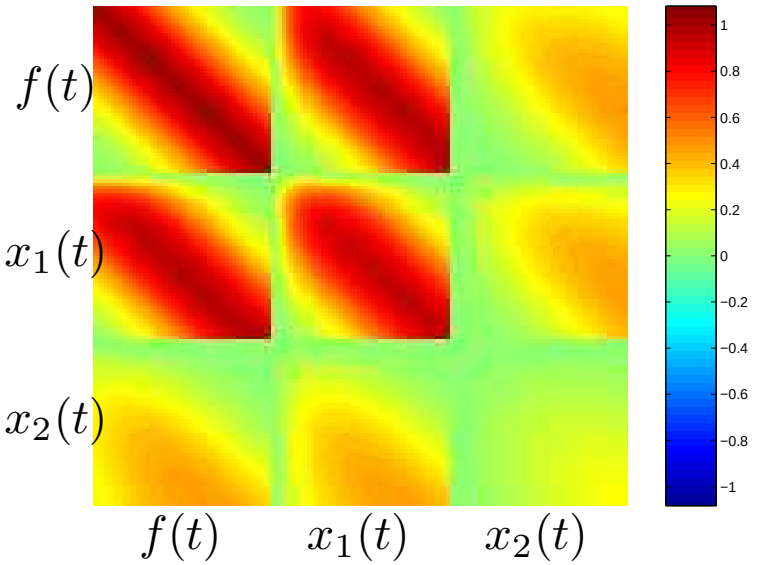
Covariance for Transcription Model

RBF Kernel function for $f(t)$

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

- 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



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

gpsimTest

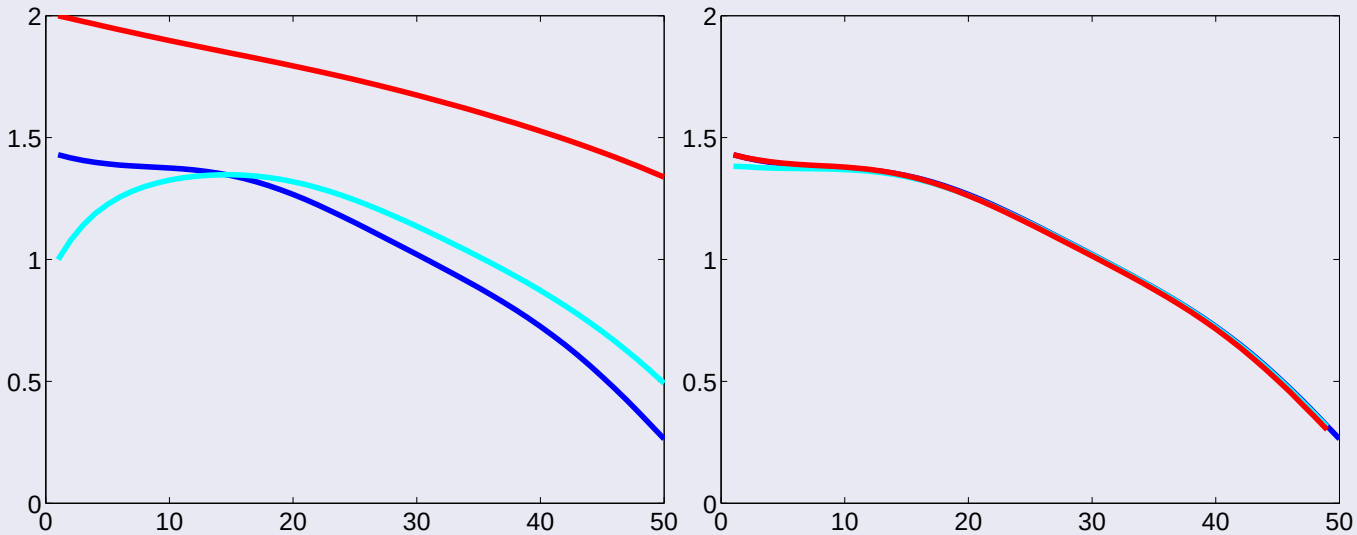


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

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

gpsimTest

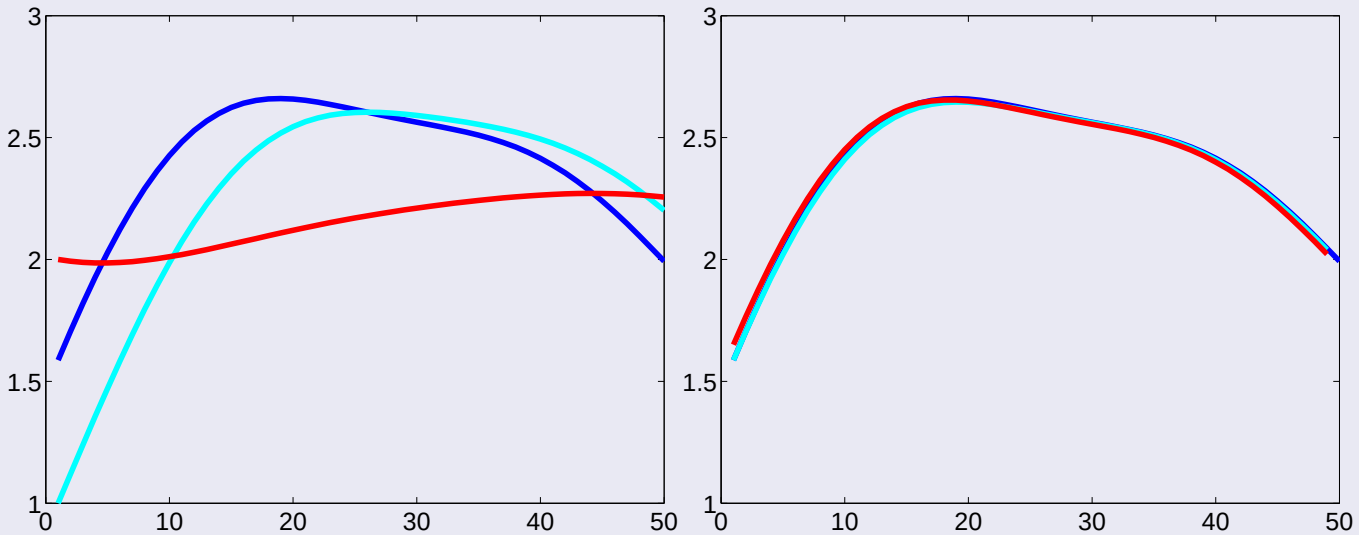


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

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

gpsimTest

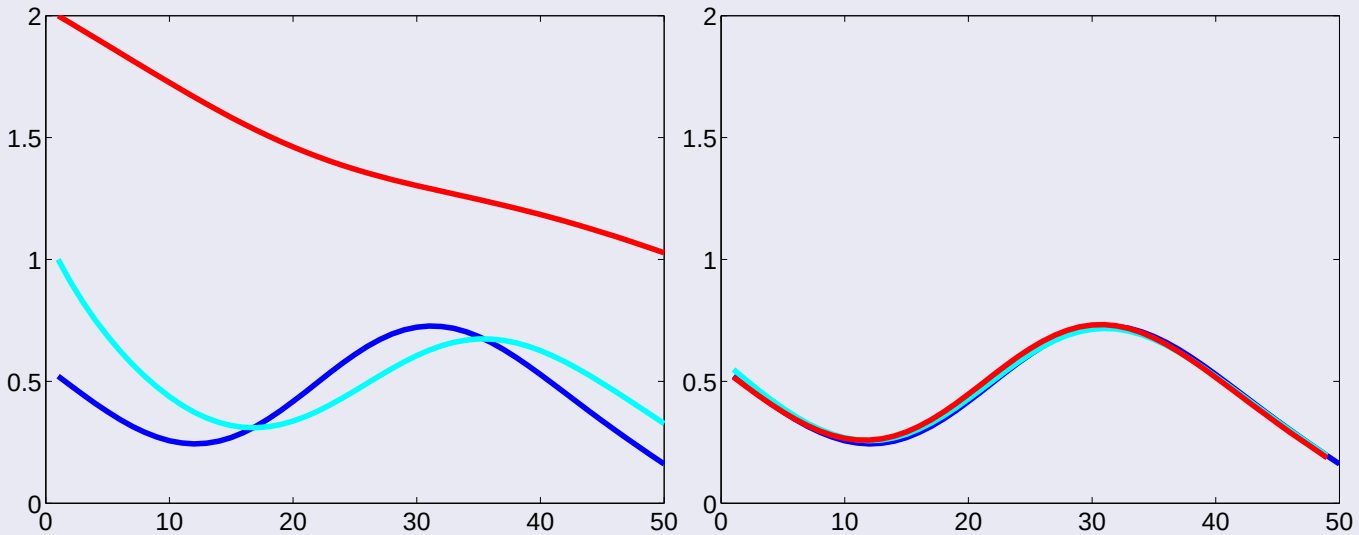


Figure: *Left*: joint samples from the transcription covariance, *blue*: $f(t)$, *cyan*: $x_1(t)$ and *red*: $x_2(t)$. *Right*: numerical solution for $f(t)$ of the differential equation from $x_1(t)$ and $x_2(t)$ (blue and cyan). True $f(t)$ included for comparison.

Noise Corruption

Estimate Underlying Noise

- Allow the mRNA abundance of each gene at each time point to be corrupted by noise, for observations at t_i for $i = 1, \ldots, T$,

$$y_j(t_i) = x_j(t_i) + \epsilon_j(t_i) \tag{4}$$

with $\epsilon_j(t_i) \sim \mathcal{N}\left(0, \sigma_{ji}^2\right)$.

- Estimate noise level using probe-level processing techniques of Affymetrix microarrays (*e.g.* mmgMOS, [Liu et al., 2005]).
- The covariance of the noisy process is then $K_{\mathbf{yy}} = \Sigma + K_{\mathbf{xx}}$, with $\Sigma = \text{diag}\left(\sigma_{11}^2, \ldots, \sigma_{1T}^2, \ldots, \sigma_{N1}^2, \ldots, \sigma_{NT}^2\right)$.

Artificial Data

Toy Problem

- Results from an artificial data set.
- We used a 'known TFC' and derived six 'mRNA profiles'.
  - Known TFC composed of three Gaussian basis functions.
  - mRNA profiles derived analytically.
- Fourteen subsamples were taken and corrupted by noise.
- This 'data' was then used to:
  - Learn decays, sensitivities and basal transcription rates.
  - Infer a posterior distribution over the missing TFC.

Artificial Data Results

demToyProblem1

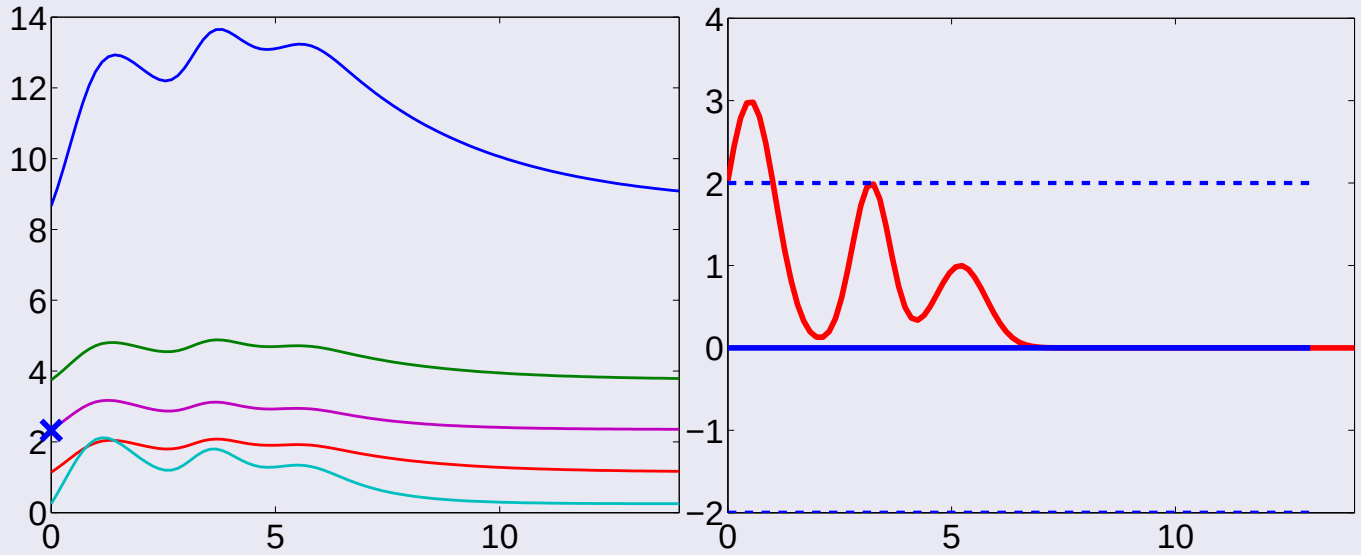


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

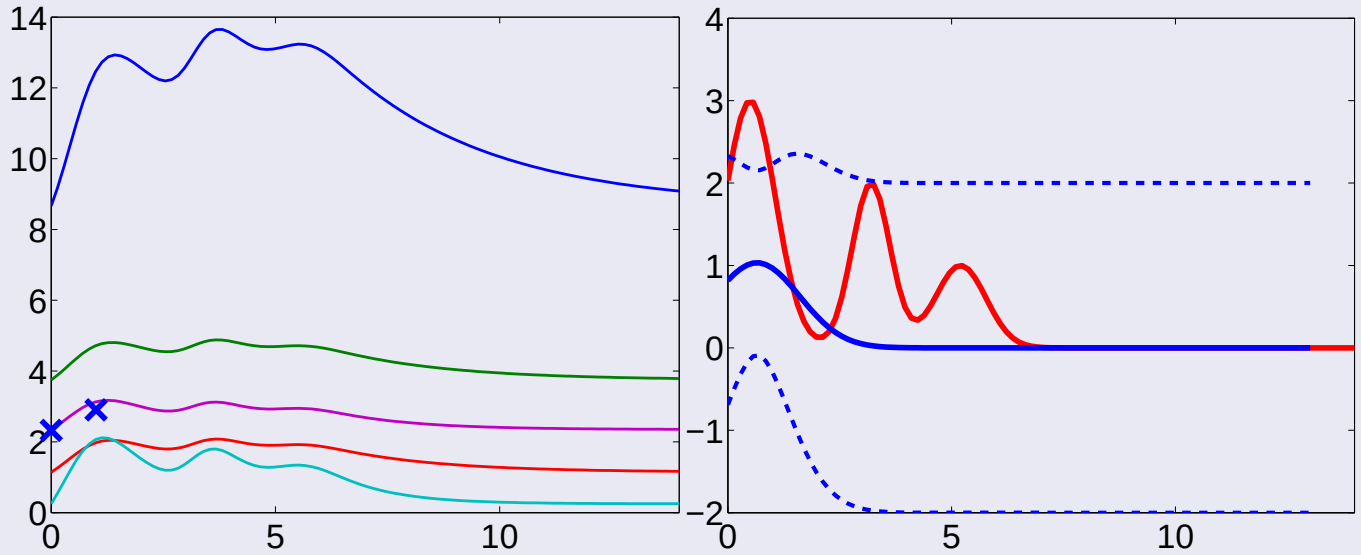


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

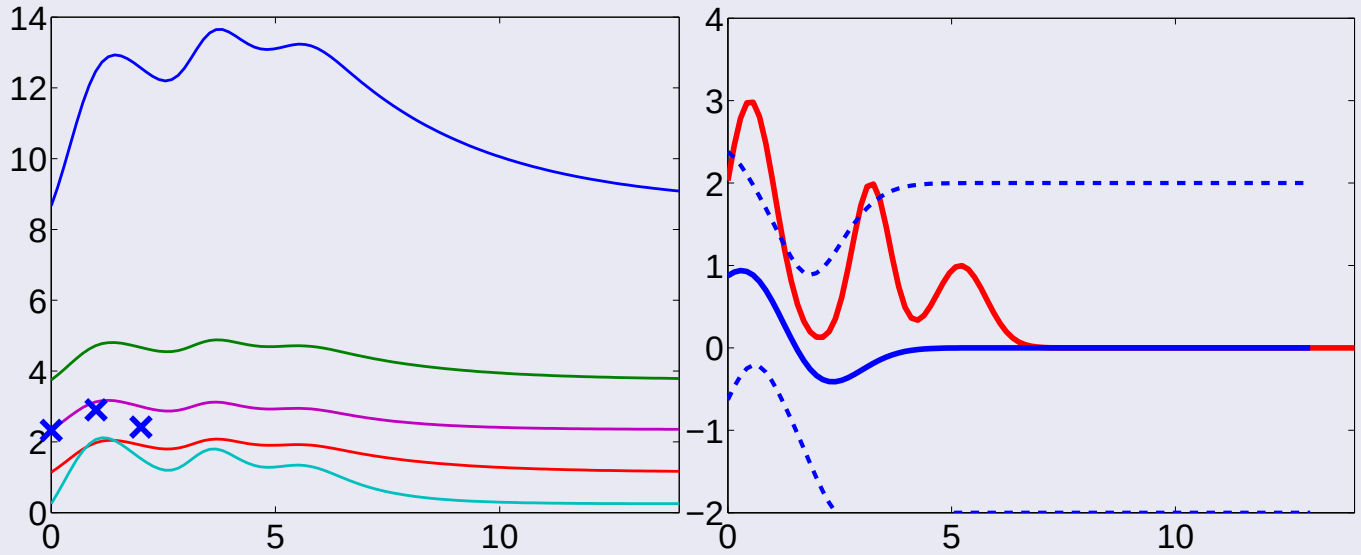


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

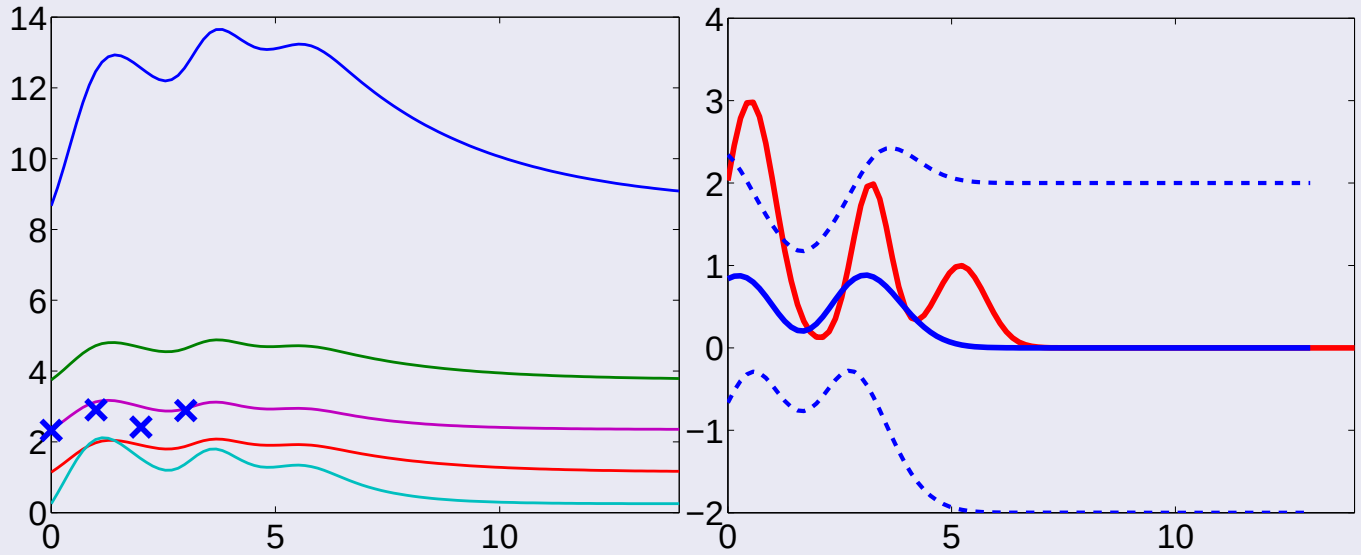


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

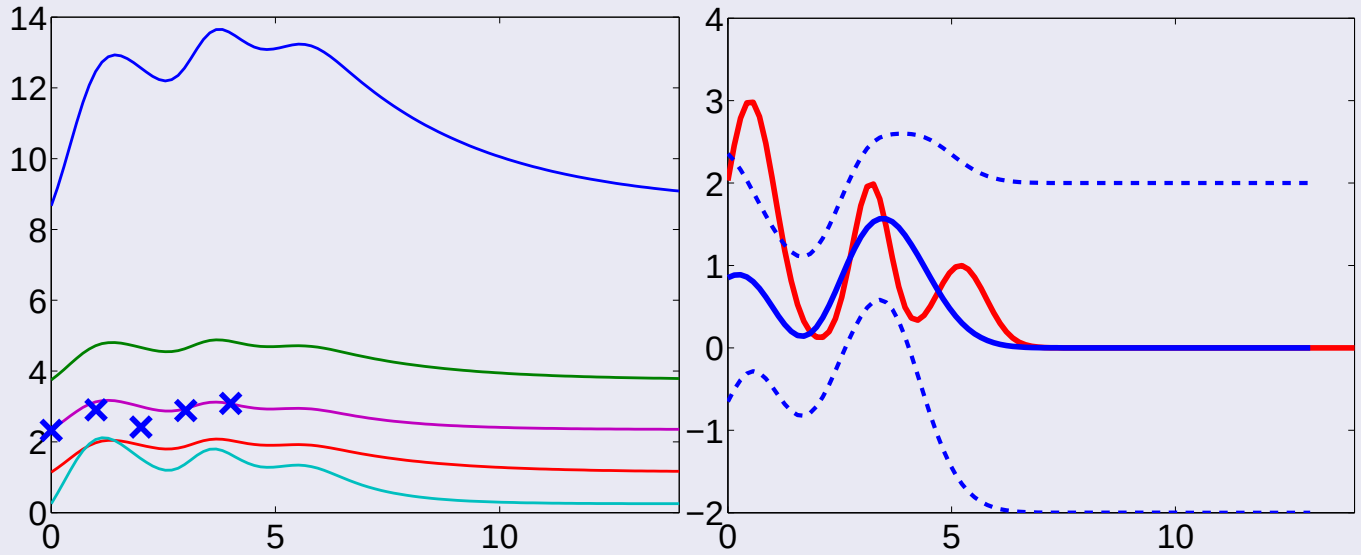


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

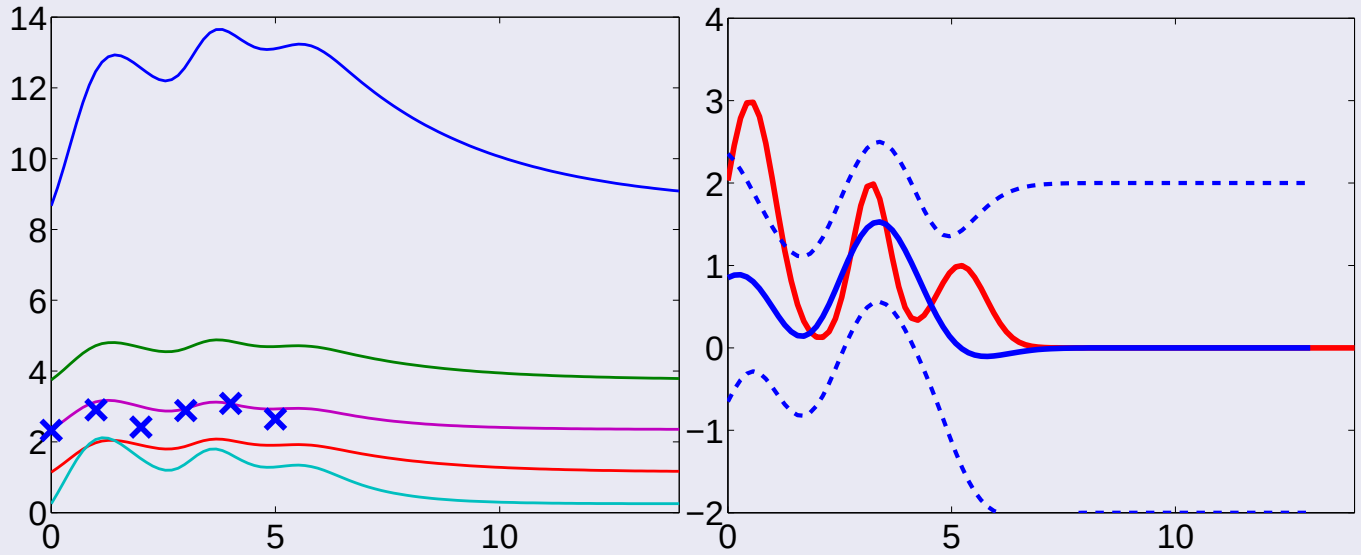


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

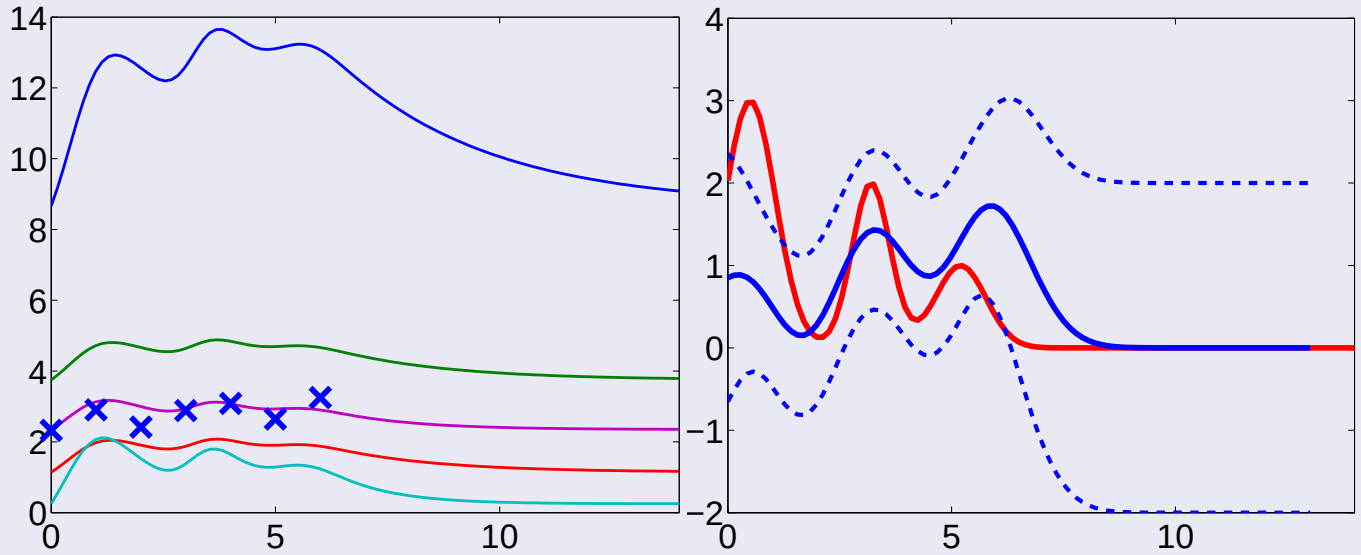


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

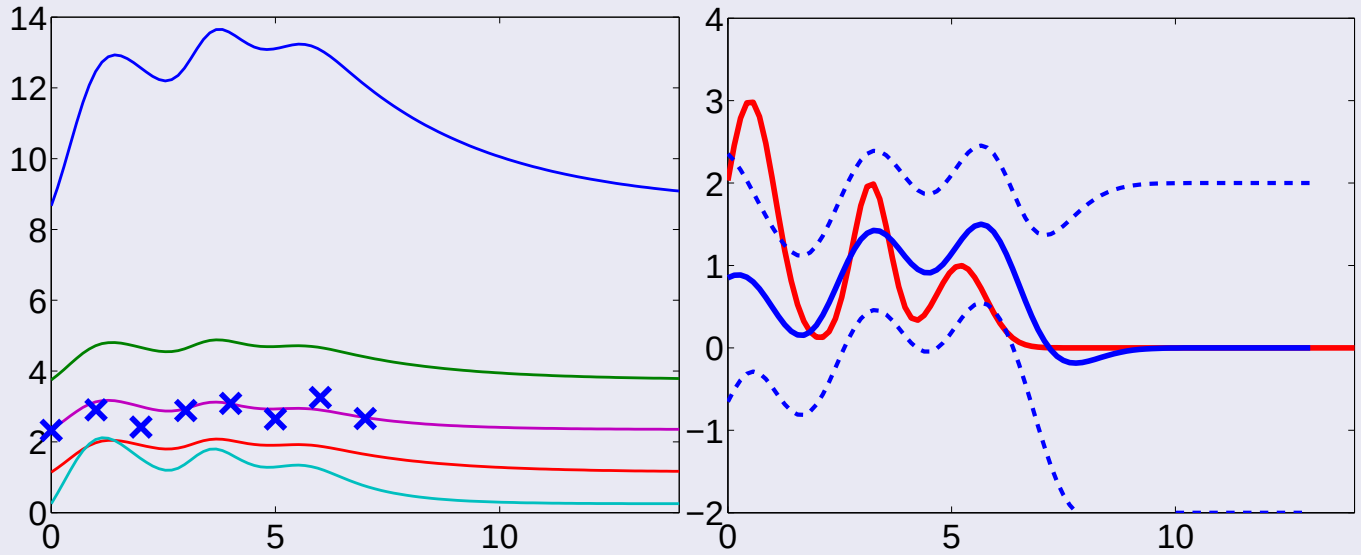


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

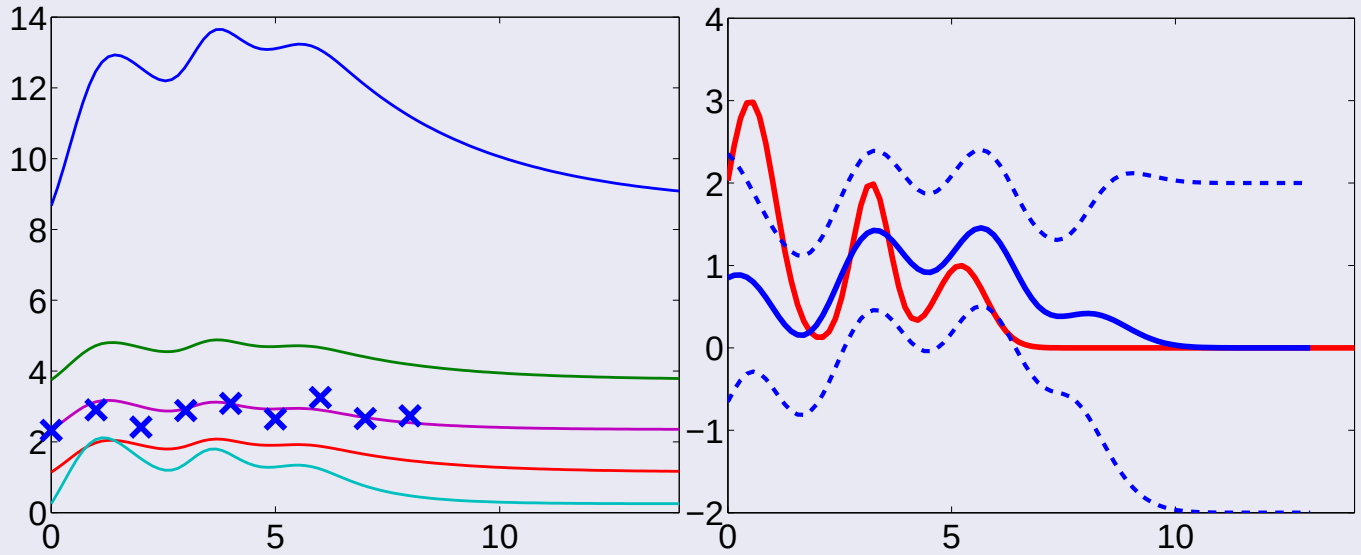


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

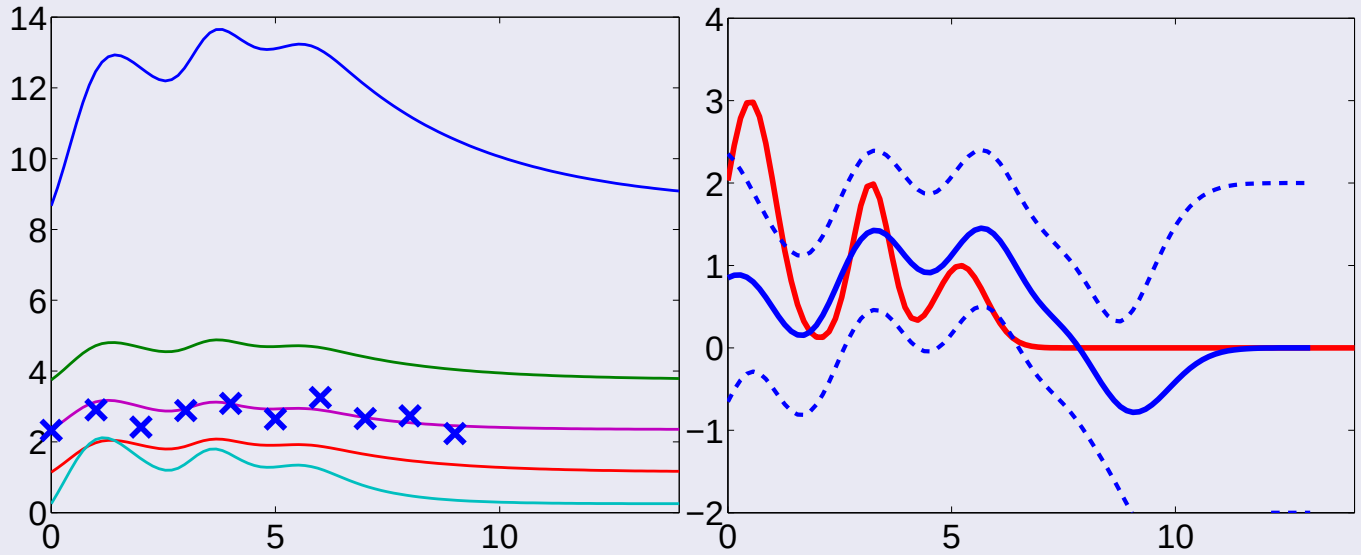


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

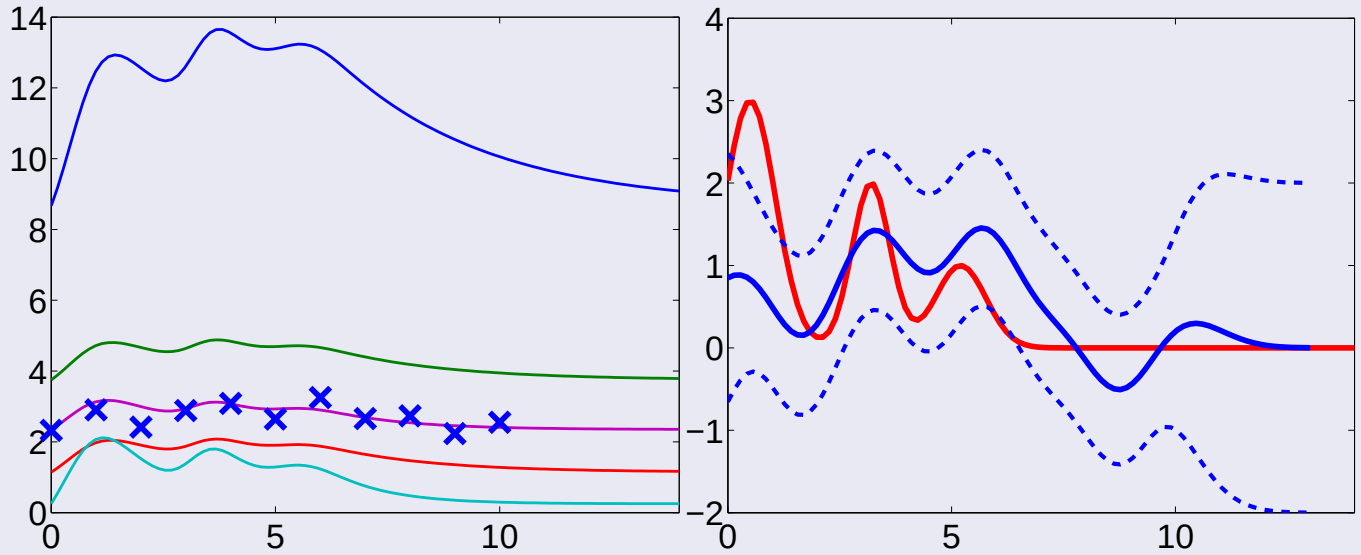


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

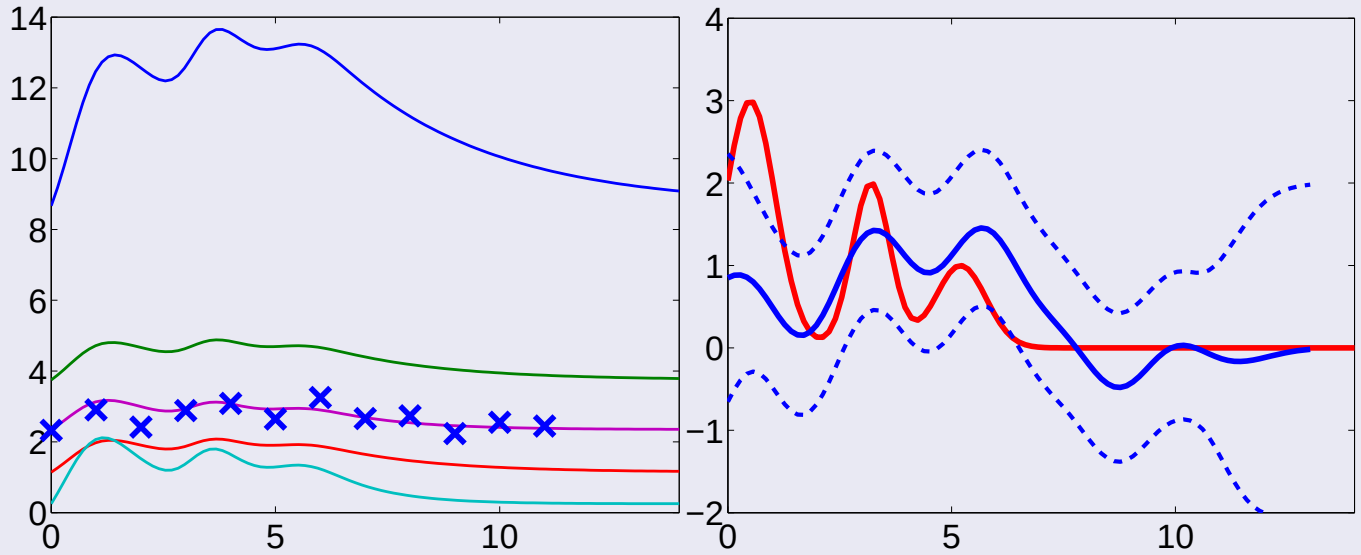


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

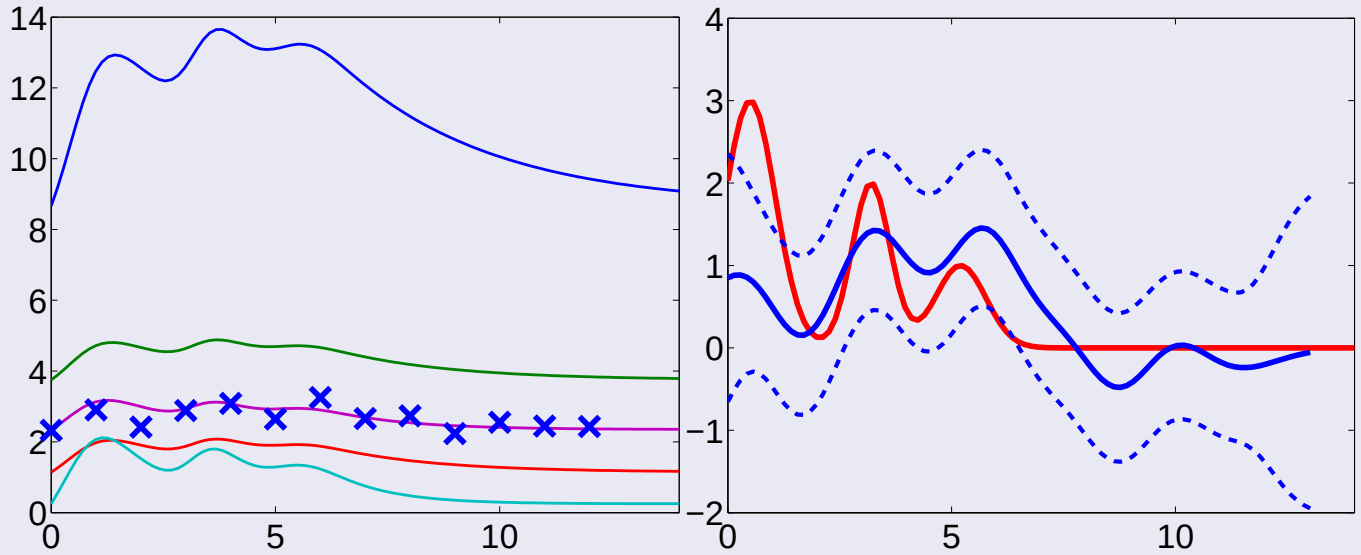


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

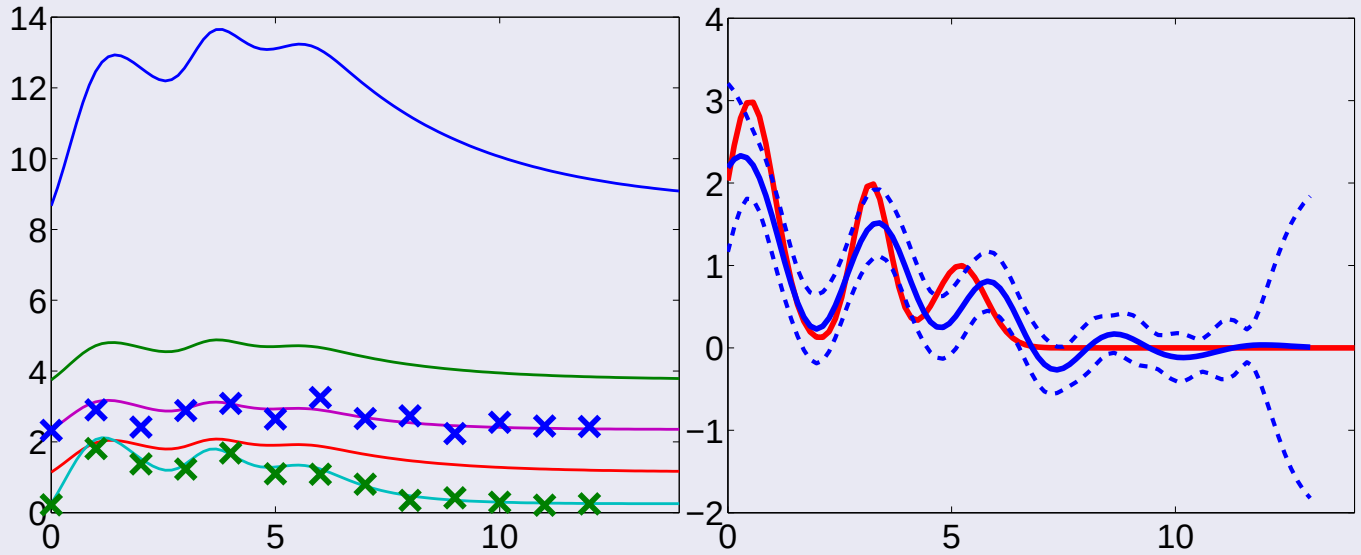


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

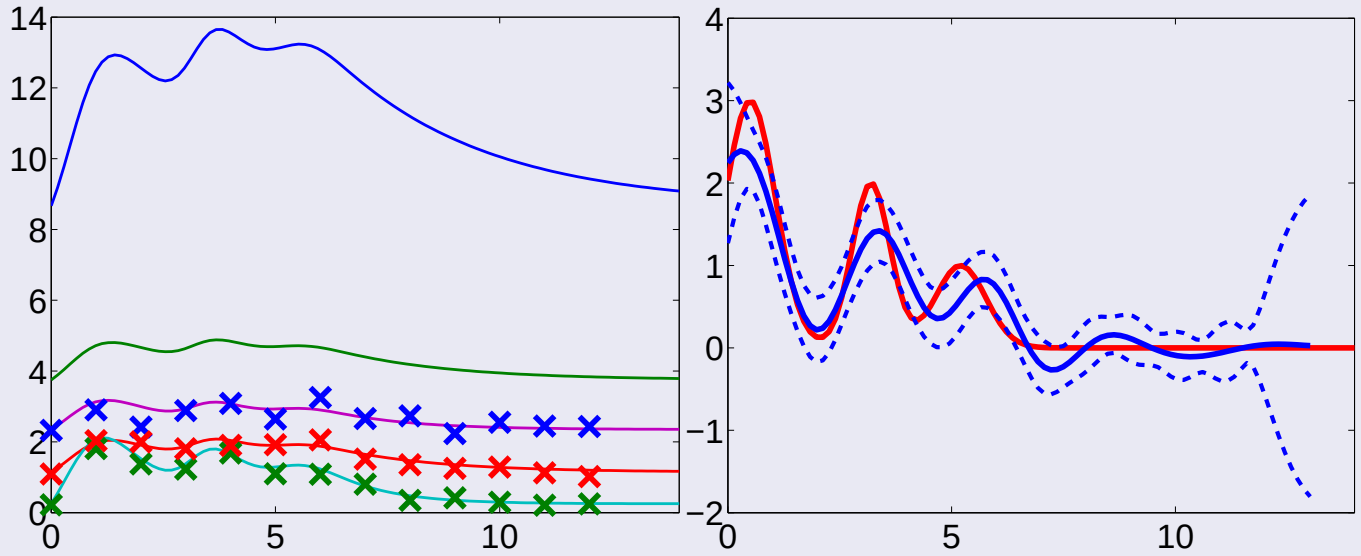


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

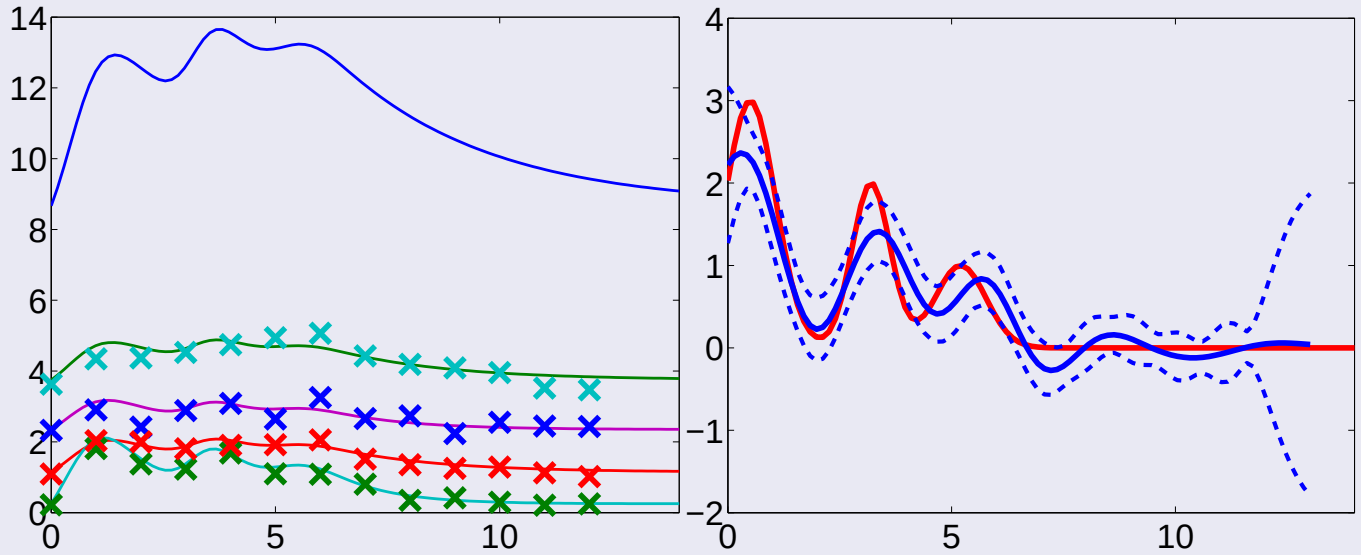


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

demToyProblem1

Artificial Data Results

demToyProblem1

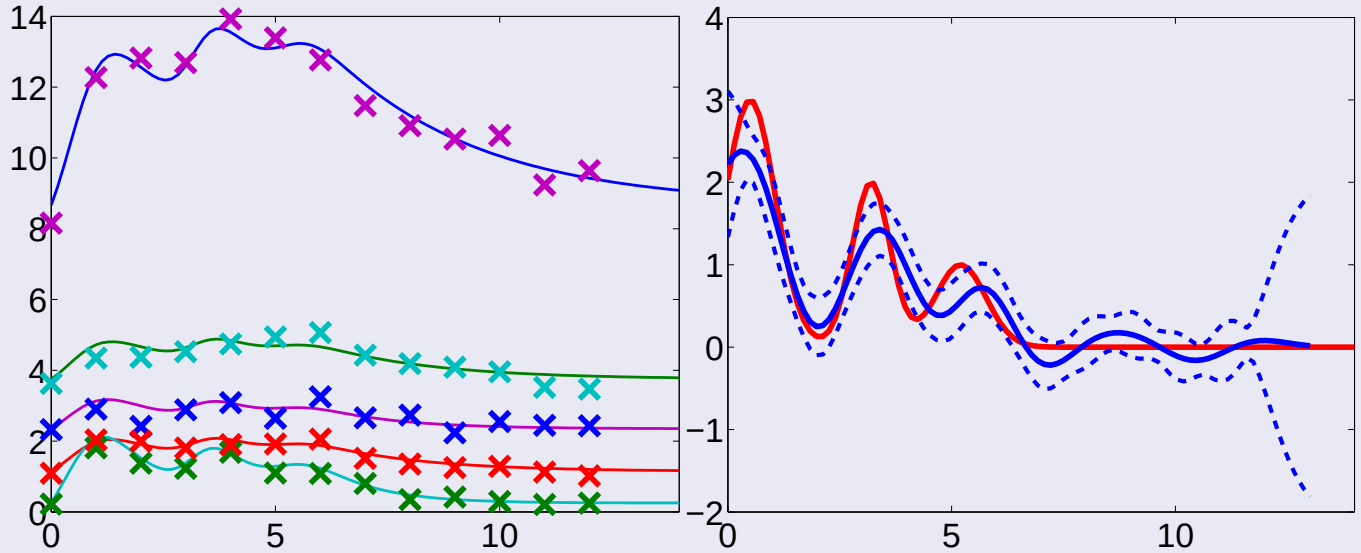


Figure: *Left:* The TFC, $f(t)$, which drives the system. *Middle:* Five gene mRNA concentration profiles each obtained by using different parameter sets $\{B_i, S_i, D_i\}_{i=1}^{5}$ (lines) along with noise corrupted 'data' . *Right:* The inferred TFC (with error bars).

Results

Linear System

- Recently published biological data set studied using linear response model by Barenco et al. [2006].
- Study focused on the tumour suppressor protein p53.
- mRNA abundance measured for five targets: *DDB2*, *p21*, *SESN1/hPA26*, *BIK* and *TNFRSF10b*.
- Quadratic interpolation for the mRNA production rates to obtain gradients.
- They used MCMC sampling to obtain estimates of the model parameters B_j, S_j, D_j and $f(t)$.

Linear response analysis

Experimental Setup

- We analysed data using the linear response model.
- Raw data was processed using the mmgMOS model of Liu et al. [2005] which provides variance as well as expression level.
- We present posterior distribution over TFCs.
- Results of inference on the values of the hyperparameters B_j, S_j and D_j.
  - Samples from the posterior distribution were obtained using Hybrid Monte Carlo (see *e.g.* Neal, 1996).

Linear Response Results

demBarenco1

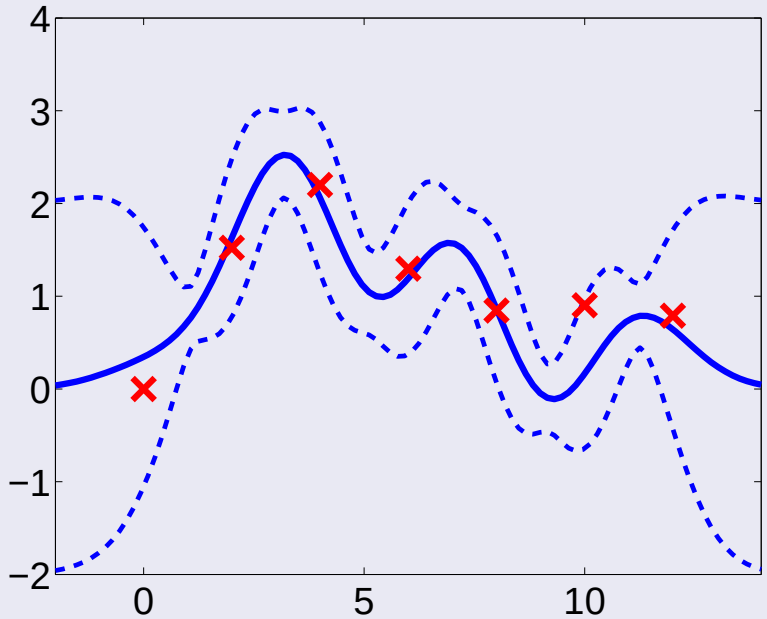


Figure: Predicted protein concentration for p53. Solid line is mean, dashed lines 95% credibility intervals. The prediction of [Barenco et al., 2006] was pointwise and is shown as crosses.

Results — Transcription Rates

Estimation of Equation Parameters `demBarenco1`

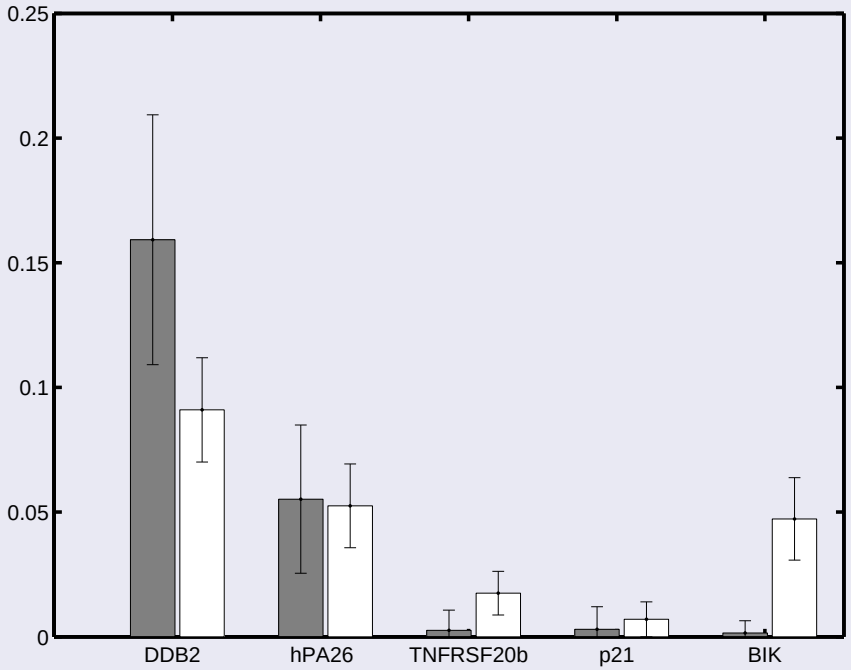


Figure: Basal transcription rates. Our results (black) compared with Barenco et al. [2006] (white).

Results — Transcription Rates

Estimation of Equation Parameters `demBarenco1`

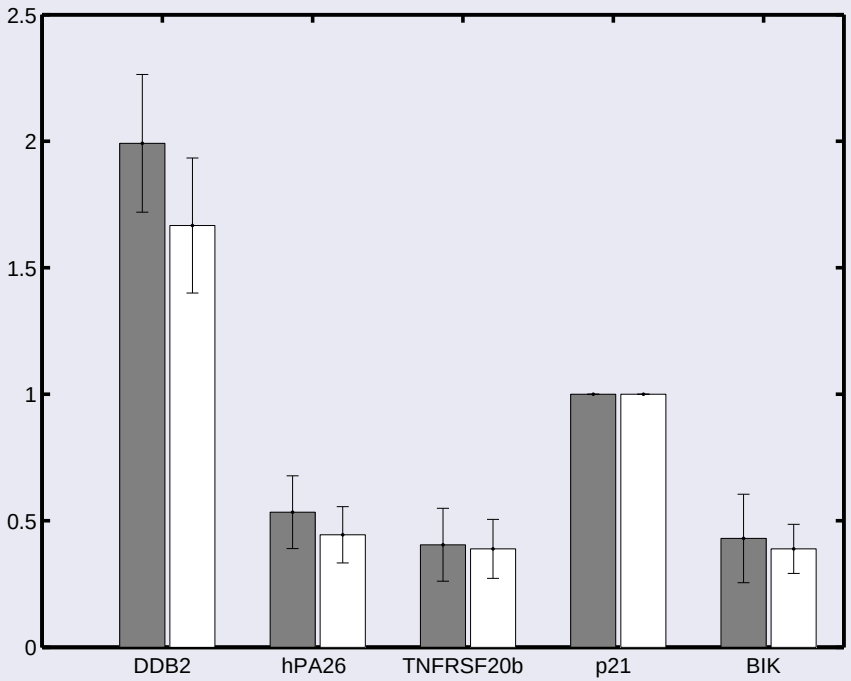


Figure: Sensitivities. Our results (black) compared with Barenco et al. [2006] (white).

Results — Transcription Rates

Estimation of Equation Parameters `demBarenco1`

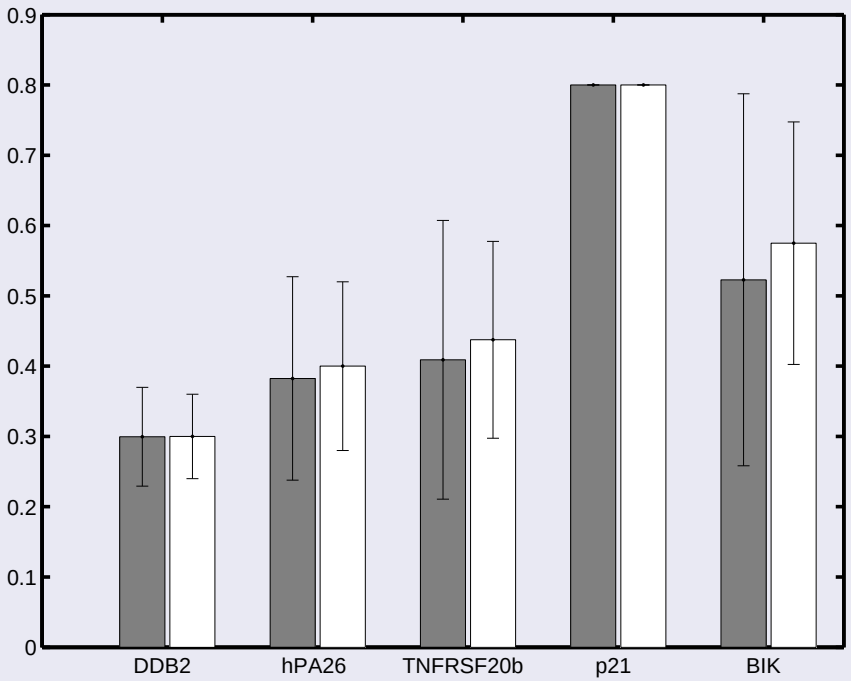


Figure: Decays. Our results (black) compared with Barenco et al. [2006] (white).

Linear Response Discussion

GP Results

- Note oscillatory behaviour, possible artifact of RBF covariance Rasmussen and Williams [see page 123 in 2006].
- Results are in good accordance with the results obtained by Barenco et al..
- Differences in estimates of the basal transcription rates probably due to:
  - different methods used for probe-level processing of the microarray data.
  - Our failure to constrain $f(0) = 0$.
- Our results take about 13 minutes to produce Barenco et al. required 10 million iterations of Monte Carlo.

Non-linear Response Model

More Realistic Response

- All the quantities in equation (1) are positive, but direct samples from a GP will not be.
- Linear models don't account for saturation.
- *Solution*: model response using a positive nonlinear function.

Formalism

Non-linear Response

- Introduce a non-linearity $g(\cdot)$ parameterised by $\boldsymbol{\theta}_j$

$$\frac{dx_j}{dt} = B_j + g(f(t), \boldsymbol{\theta}_j) - D_j x_j$$

$$x_j(t) = \frac{B_j}{D_j} + \exp\left(-D_j t\right) \int_0^t \mathrm{d}u \, g(f(u), \boldsymbol{\theta}_j) \exp\left(D_j u\right) \ .$$

- The induced distribution of $x_j(t)$ is no longer a GP.
- Derive the functional gradient and learn a MAP solution for $f(t)$.
- Also compute Hessian so we can approximate the marginal likelihood.

Example: linear response

Using non-RBF kernels

- Start by taking $g(\cdot)$ to be linear.
- Provides 'sanity check' and allows arbitrary covariance functions.
- Avoids double numerical integral that would normally be required.

Response Results

demBarencoMap1, demBarencoMap2

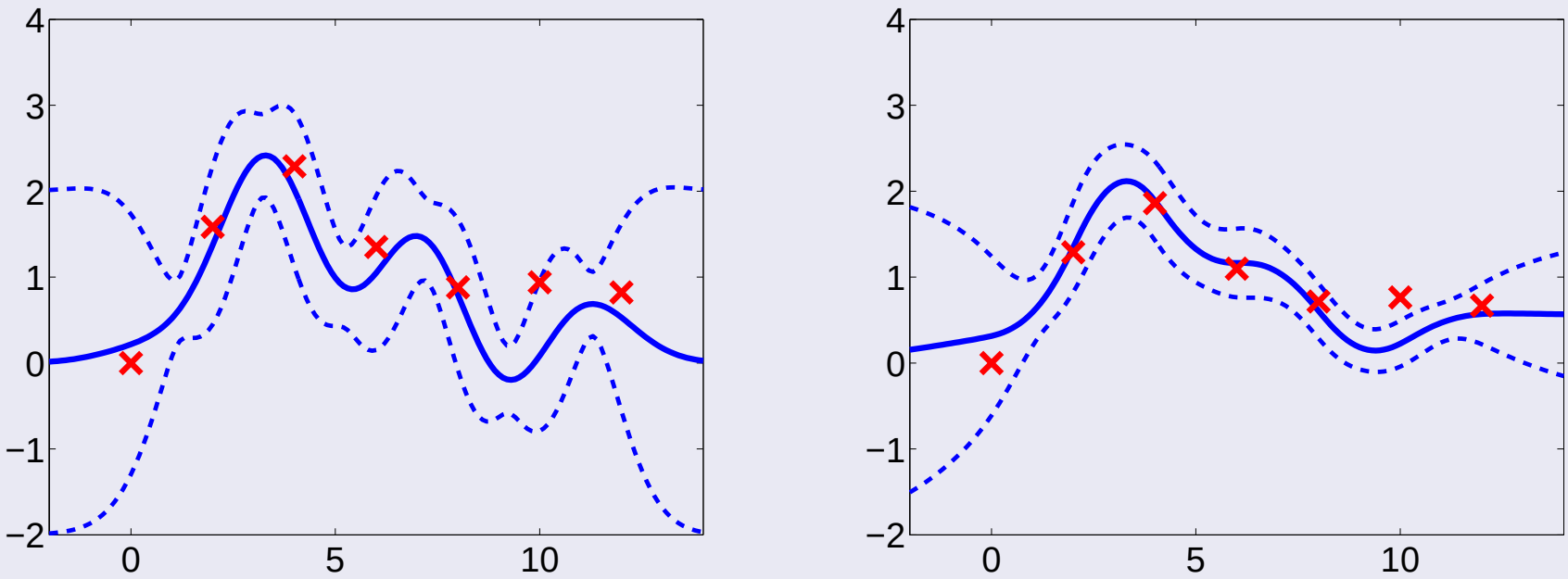


Figure: *Left:* RBF prior on f (log likelihood -101.4); *Right:* MLP prior on f (log likelihood -105.6).

Non-linear response analysis

Non-linear responses

- Exponential response model (constrains protein concentrations positive).
- $\log(1 + \exp(f))$ response model.
- $\frac{3}{1+\exp(-f)}$
- Inferred MAP solutions for the latent function f are plotted below.

exp (·) Response Results

demBarencoMap3, demBarencoMap4

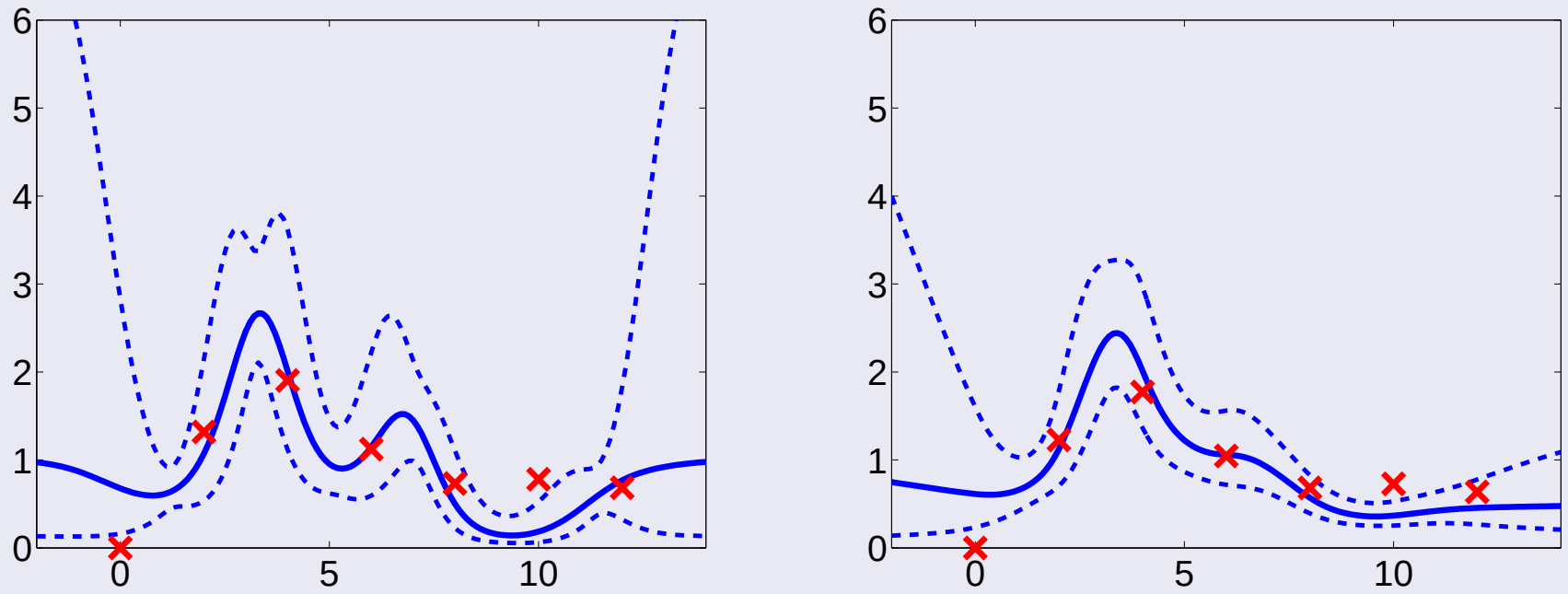


Figure: *Left:* shows results of using a squared exponential prior covariance on f (log likelihood -100.6); *Right:* shows results of using an MLP prior covariance on f (log likelihood -106.4).

$\log(1 + \exp(f))$ Response Results

demBarencoMap5, demBarencoMap6

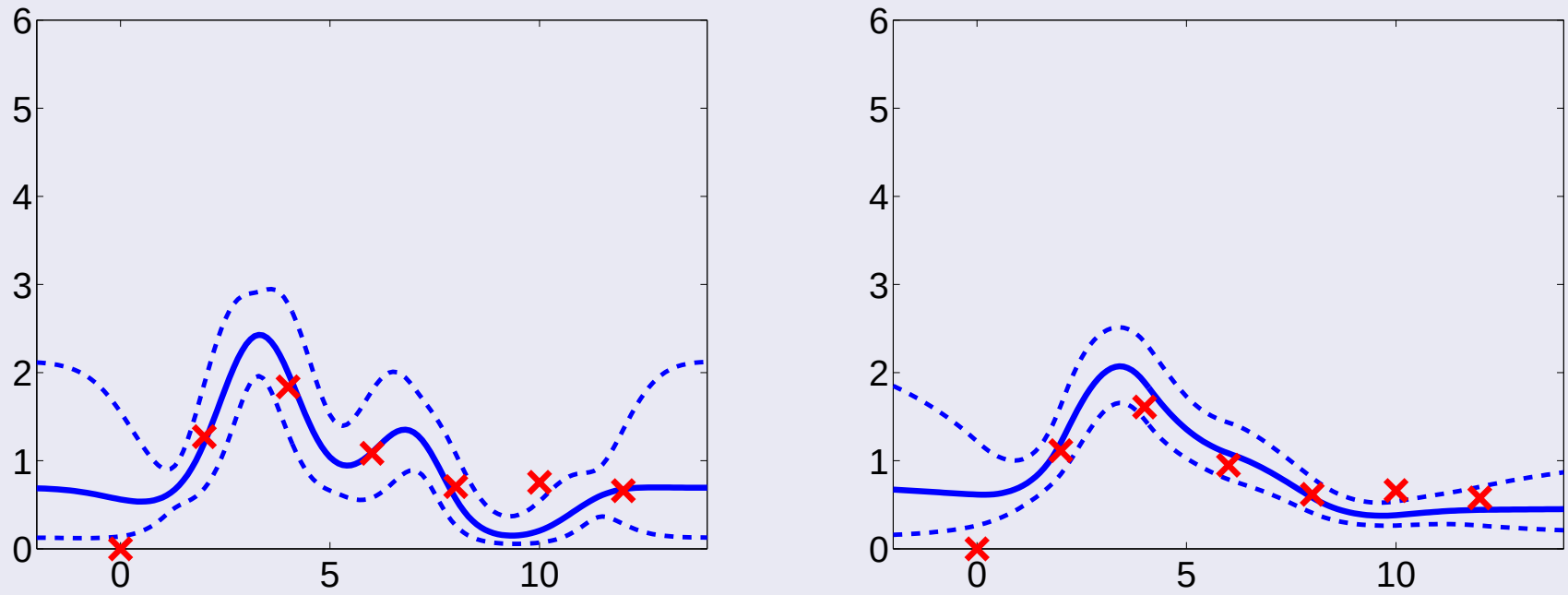


Figure: *Left:* shows results of using a squared exponential prior covariance on f (log likelihood -100.9); *Right:* shows results of using an MLP prior covariance on f (log likelihood -110.0).

$\frac{3}{1+\exp(-f)}$ Response Results

demBarencoMap7, demBarencoMap8

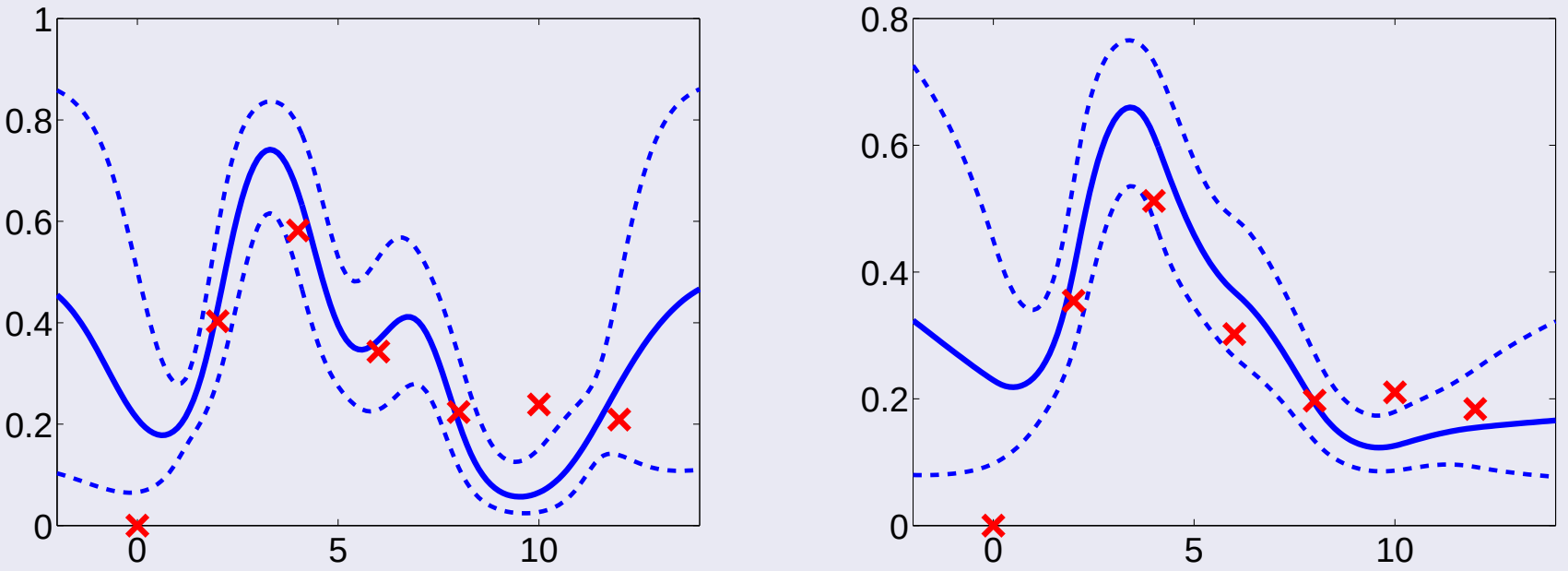


Figure: *Left:* shows results of using a squared exponential prior covariance on f (log likelihood -104.1); *Right:* shows results of using an MLP prior covariance on f (log likelihood -111.2).

Discussion

- We have described how GPs can be used in modelling dynamics of a simple regulatory network motif.
- Our approach has advantages over standard parametric approaches:
  - there is no need to restrict the inference to the observed time points, the temporal continuity of the inferred functions is accounted for naturally.
  - GPs allow us to handle uncertainty in a natural way.
  - MCMC parameter estimation in a discretised model can be computationally expensive. Parameter estimation can be achieved easily in our framework by type II maximum likelihood or by using efficient hybrid Monte Carlo sampling techniques
- All code on-line `http://www.cs.man.ac.uk/~neill/gpsim/`.

Future Directions

What Next?

- This is still a very simple modelling situation.
  - We are ignoring transcriptional delays.
  - Here we have single transcription factor: our ultimate goal is to describe regulatory pathways with more genes.
  - All these issues can be dealt with in the general framework we have described.
  - Need to overcome the greater computational difficulties.

Acknowledgements

Data and Support

We thank Martino Barenco for useful discussions and for providing the data. We gratefully acknowledge support from BBSRC Grant No BBS/B/0076X "Improved processing of microarray data with probabilistic models".

Covariance Result

Covariance Result

$$k_{x_j x_k}(t, t') = S_j S_k \frac{\sqrt{\pi}}{2} \left[h_{kj}(t', t) + h_{jk}(t, t') \right]$$

where

$$\begin{aligned} h_{kj}(t', t) = & \frac{\exp(\gamma_k)^2}{D_j + D_k} \\ & \times \left\{ \exp\left[-D_k (t' - t)\right] \left[\operatorname{erf}\left(\frac{t' - t}{l} - \gamma_k \right) + \operatorname{erf}\left(\frac{t}{l} + \gamma_k \right) \right] \right. \\ & \left. - \exp\left[-(D_k t' + D_j) \right] \left[\operatorname{erf}\left(\frac{t'}{l} - \gamma_k \right) + \operatorname{erf}(\gamma_k) \right] \right\}. \end{aligned}$$

Here $\gamma_k = \frac{D_k l}{2}$.

Cross Covariance

Correlation of $x_j(t)$ and $f(t')$

- Need the "cross-covariance" terms between $x_j(t)$ and $f(t')$, which is obtained as

$$k_{x_j f}(t, t') = S_j \exp(-D_j t) \int_0^t \exp(D_j u) k_{ff}(u, t') du. \quad (5)$$

- For RBF we have

$$k_{x_j f}(t', t) = \frac{\sqrt{\pi} l S_j e^{2\gamma_j}}{2} \exp\left[-D_j(t' - t)\right] \left[\operatorname{erf}\left(\frac{t' - t}{l} - \gamma_j\right) + \operatorname{erf}\left(\frac{t}{l} + \gamma_j\right)\right].$$

Posterior for f

Prediction for TFC

- Standard Gaussian process regression techniques [see *e.g.* Rasmussen and Williams, 2006] yield

$$\langle f \rangle_{\text{post}} = K_{f\mathbf{x}} K_{\mathbf{xx}}^{-1} \mathbf{x}$$
$$K_{ff}^{\text{post}} = K_{ff} - K_{f\mathbf{x}} K_{\mathbf{xx}}^{-1} K_{\mathbf{x}f}$$

- Model parameters B_j, D_j and S_j estimated by type II maximum likelihood,

$$\log p(\mathbf{x}) = N(\mathbf{x}|\mathbf{0}, K_{\mathbf{xx}})$$

Implementation

Riemann quadrature

- Implementation requires a discretised time.
- Compute the gradient and Hessian on a grid.
- Integrate them by approximate Riemann quadrature.
- We choose a uniform grid $\{t_p\}_{p=1}^{M}$ so that $\Delta = t_p - t_{p-1}$ is constant.
- The vector $\mathbf{f} = \{f_p\}_{p=1}^{M}$ is the function f at the grid points.

$$I(t) = \int_0^t f(u) \exp(D_j u)\, du$$

$$I(t) \approx \sum_{p=1}^{M} f(t_p) \exp(D_j t_p)\, \Delta$$

Log Likelihood

Functional Gradient

- Given noise-corrupted data $y_j(t_i)$ the log-likelihood is

$$\log p(Y|f,\theta_j) = -\frac{1}{2}\sum_{i=1}^{T}\sum_{j=1}^{N}\left[\frac{(x_j(t_i)-y_j(t_i))^2}{\sigma_{ji}^2} - \log\left(\sigma_{ji}^2\right)\right] - \frac{NT}{2}\log(2\pi)$$

- The functional derivative of the log-likelihood wrt f is

$$\frac{\delta \log p(Y|f)}{\delta f(t)} = -\sum_{i=1}^{T}\Theta(t_i - t)\sum_{j=1}^{N}\frac{(x_j(t_i)-y_j(t_i))}{\sigma_{ji}^2}g'(f(t))\mathrm{e}^{-D_j(t_i-t)}$$

$\Theta(x)$ — Heaviside step function.

Log Likelihood

Functional Hessian

- Given noise-corrupted data $y_j(t_i)$ the log-likelihood is

$$\log p(Y|f,\theta_j) = -\frac{1}{2}\sum_{i=1}^{T}\sum_{j=1}^{N}\left[\frac{\left(x_j(t_i)-y_j(t_i)\right)^2}{\sigma_{ji}^2} - \log\left(\sigma_{ji}^2\right)\right] - \frac{NT}{2}\log(2\pi)$$

- The negative Hessian of the log-likelihood wrt f is

$$w(t,t') = \sum_{i=1}^{T}\Theta(t_i-t)\delta\left(t-t'\right)\sum_{j=1}^{N}\frac{\left(x_j(t_i)-y_j(t_i)\right)}{\sigma_{ji}^2}g''(f(t))\mathrm{e}^{-D_j(t_i-t)}$$
$$+\sum_{i=1}^{T}\Theta(t_i-t)\Theta\left(t_i-t'\right)\sum_{j=1}^{N}\sigma_{ji}^{-2}g'\left(f(t)\right)g'\left(f(t')\right)\mathrm{e}^{-D_j(2t_i-t-t')}$$

$$g'(f) = \partial g/\partial f \text{ and } g''(f) = \partial^2 g/\partial f^2.$$

Implementation II

Combine with Prior

- Combine these with prior to compute gradient and Hessian of log posterior $\Psi(\mathbf{f}) = \log p(Y|\mathbf{f}) + \log p(\mathbf{f})$ [see Rasmussen and Williams, 2006, chapter 3]

$$\begin{aligned}\frac{\partial \Psi(\mathbf{f})}{\partial \mathbf{f}} &= \frac{\partial \log p(Y|\mathbf{f})}{\partial \mathbf{f}} - K^{-1}\mathbf{f}\\ \frac{\partial^2 \Psi(\mathbf{f})}{\partial \mathbf{f}^2} &= -(W + K^{-1})\end{aligned} \tag{6}$$

K prior covariance evaluated at the grid points.

- Use to find a MAP solution via, $\hat{\mathbf{f}}$, using Newton's algorithm.
- The Laplace approximation is then

$$\log p(Y) \simeq \log p(Y|\hat{\mathbf{f}}) - \tfrac{1}{2}\hat{\mathbf{f}}^T K^{-1}\hat{\mathbf{f}} - \tfrac{1}{2}\log|I + KW|. \tag{7}$$

The University of Manchester

References

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.

T. Graepel. Solving noisy linear operator equations by Gaussian processes: Application to ordinary and partial differential equations. In T. Fawcett and N. Mishra, editors, *Proceedings of the International Conference in Machine Learning*, volume 20, pages 234–241. AAAI Press, 2003. ISBN 1-57735-189-4.

X. Liu, M. Milo, N. D. Lawrence, and M. Rattray. A tractable probabilistic model for Affymetrix probe-level analysis across multiple chips. *Bioinformatics*, 21(18):3637–3644, 2005.

R. Murray-Smith and B. A. Pearlmutter. Transformations of Gaussian process priors.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, 1996. Lecture Notes in Statistics 118.

A. O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society, B*, 40:1–42, 1978.

A. O'Hagan. Some Bayesian numerical analysis. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 4*, pages 345–363, Valencia, 1992. Oxford University Press.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 026218253X.

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.