Multiple Output Processes

Neil D. Lawrence

GPRS
19th–22nd January 2015

Outline

Outline

Multiple Output Gaussian Processes

- In this section we will study Gaussian processes with multiple outputs.
- they have various names, vector valued functions, multiple outputs, multidimensional GPs, multi-task learning.
- Key idea, we want to relate several different functions.
- Sounds more complex, but actually it's a special case of a normal GP where one input is discrete.
- Question: how to embed covariation between the functions.
- Start by introducing *Kalman filter/smoother*.

Simple Markov Chain

- Assume 1-d latent state, a vector over time, $\mathbf{x} = [x_1 \dots x_T]$.
- Markov property,

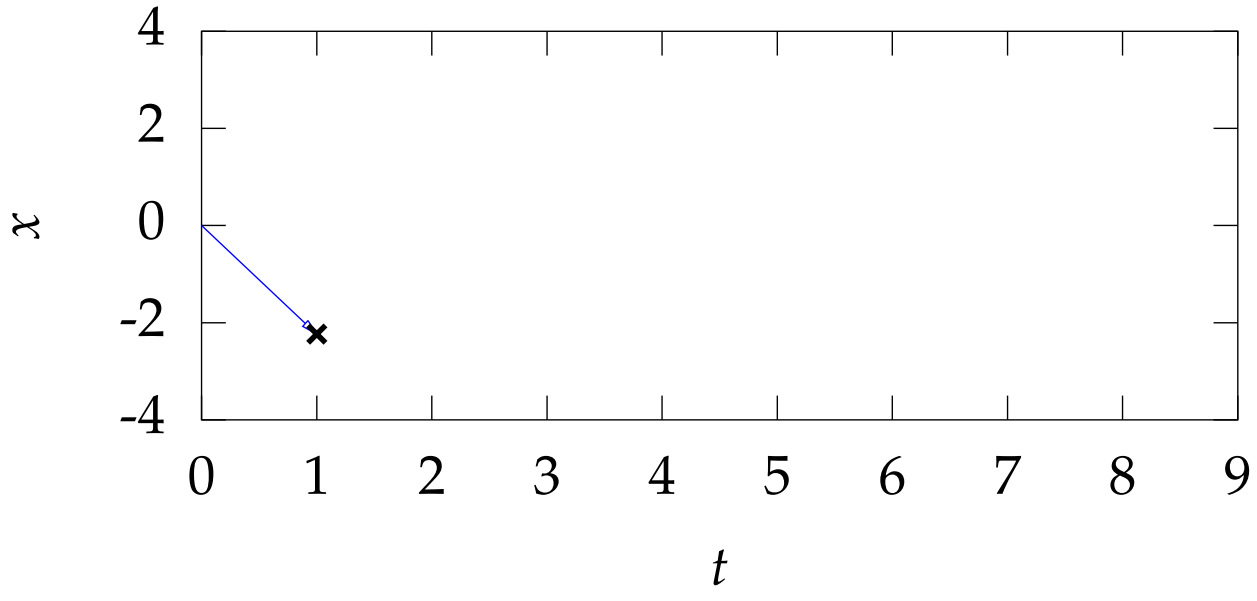
$$\begin{aligned} x_i &= x_{i-1} + \epsilon_i, \\ \epsilon_i &\sim \mathcal{N}(0, \alpha) \\ \implies x_i &\sim \mathcal{N}(x_{i-1}, \alpha) \end{aligned}$$

- Initial state,

$$x_0 \sim \mathcal{N}(0, \alpha_0)$$

- If $x_0 \sim \mathcal{N}(0, \alpha)$ we have a Markov chain for the latent states.
- Markov chain it is specified by an initial distribution (Gaussian) and a transition distribution (Gaussian).

Gauss Markov Chain

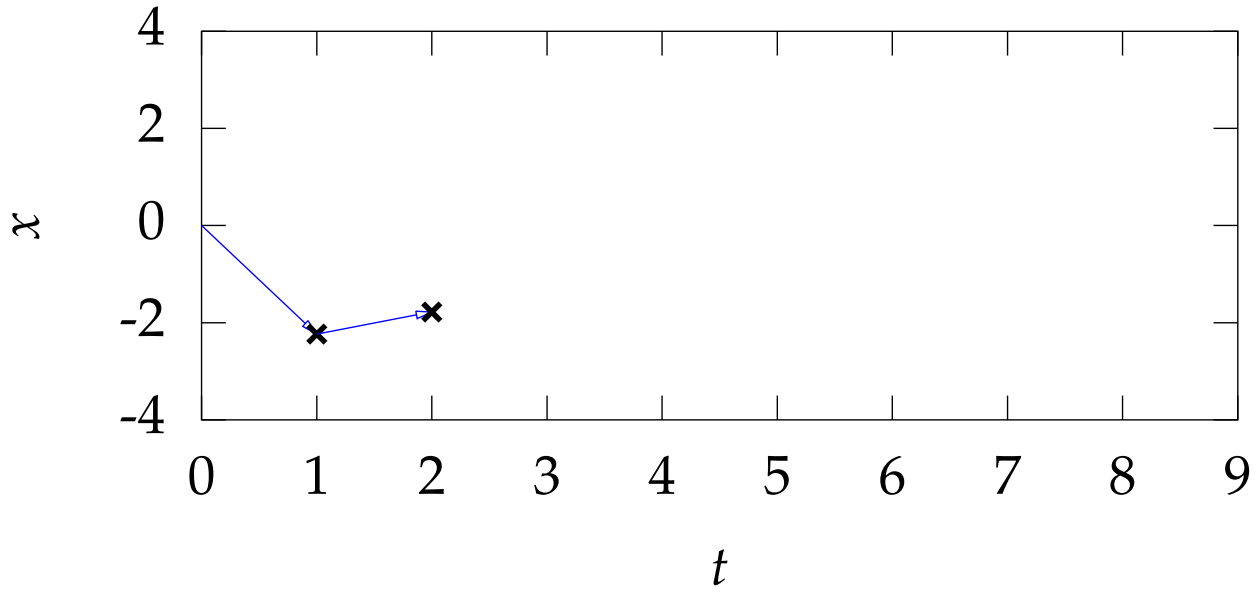


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_0 = 0.000, \quad \epsilon_1 = -2.24$$

$$x_1 = 0.000 - 2.24 = -2.24$$

Gauss Markov Chain

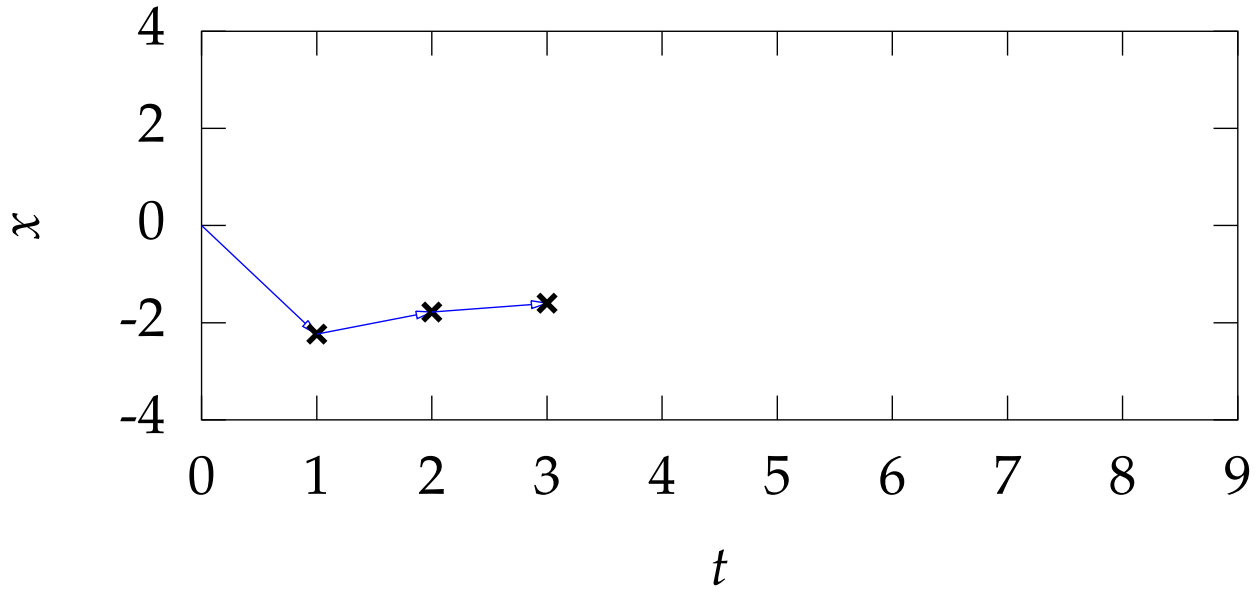


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_1 = -2.24, \quad \epsilon_2 = 0.457$$

$$x_2 = -2.24 + 0.457 = -1.78$$

Gauss Markov Chain

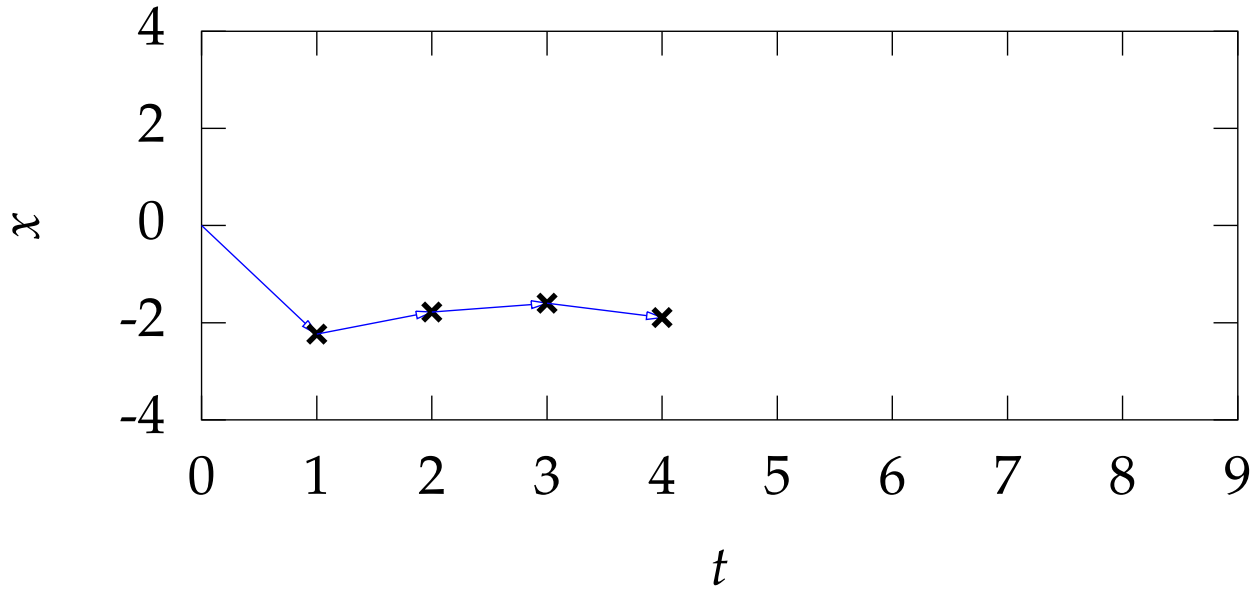


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_2 = -1.78, \quad \epsilon_3 = 0.178$$

$$x_3 = -1.78 + 0.178 = -1.6$$

Gauss Markov Chain

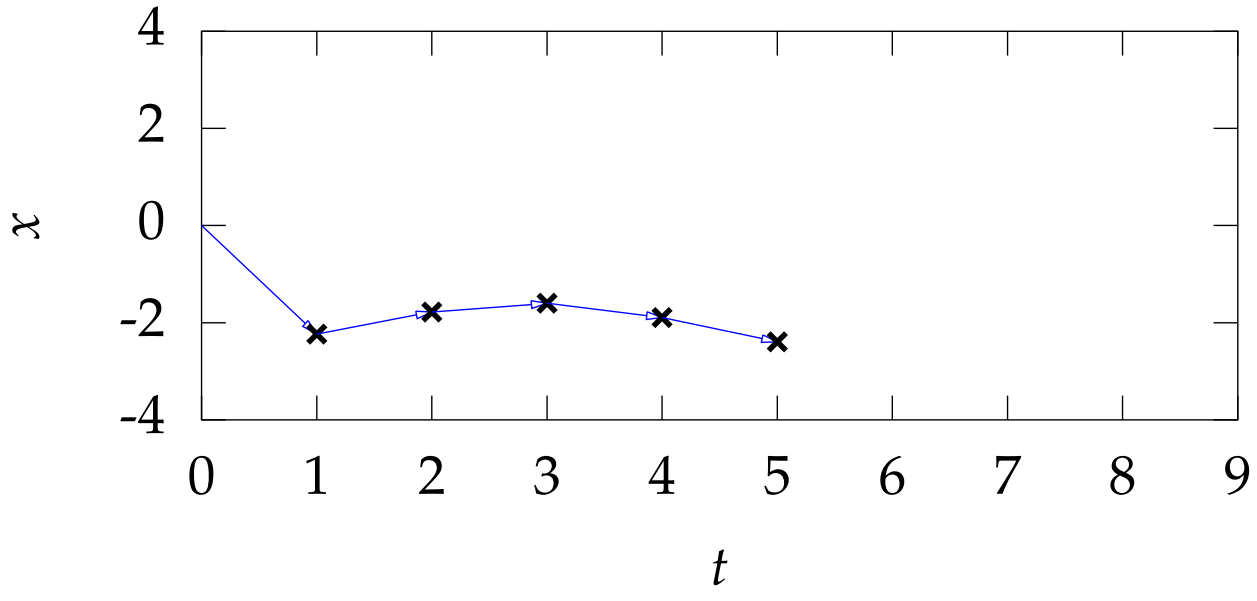


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_3 = -1.6, \quad \epsilon_4 = -0.292$$

$$x_4 = -1.6 - 0.292 = -1.89$$

Gauss Markov Chain

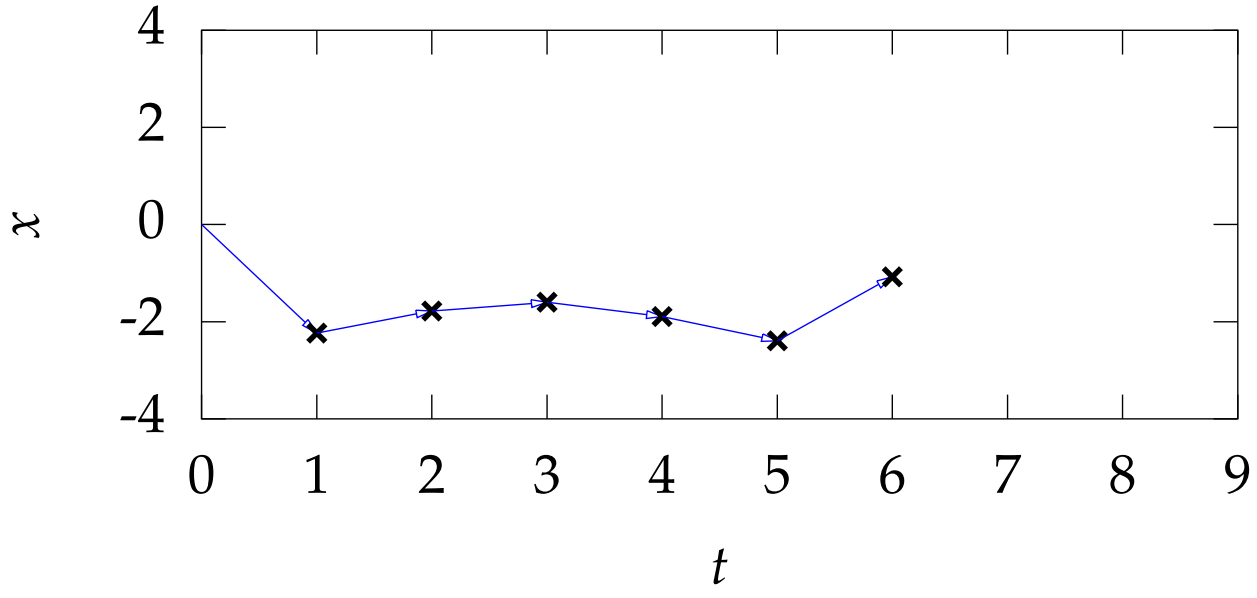


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_4 = -1.89, \quad \epsilon_5 = -0.501$$

$$x_5 = -1.89 - 0.501 = -2.39$$

Gauss Markov Chain

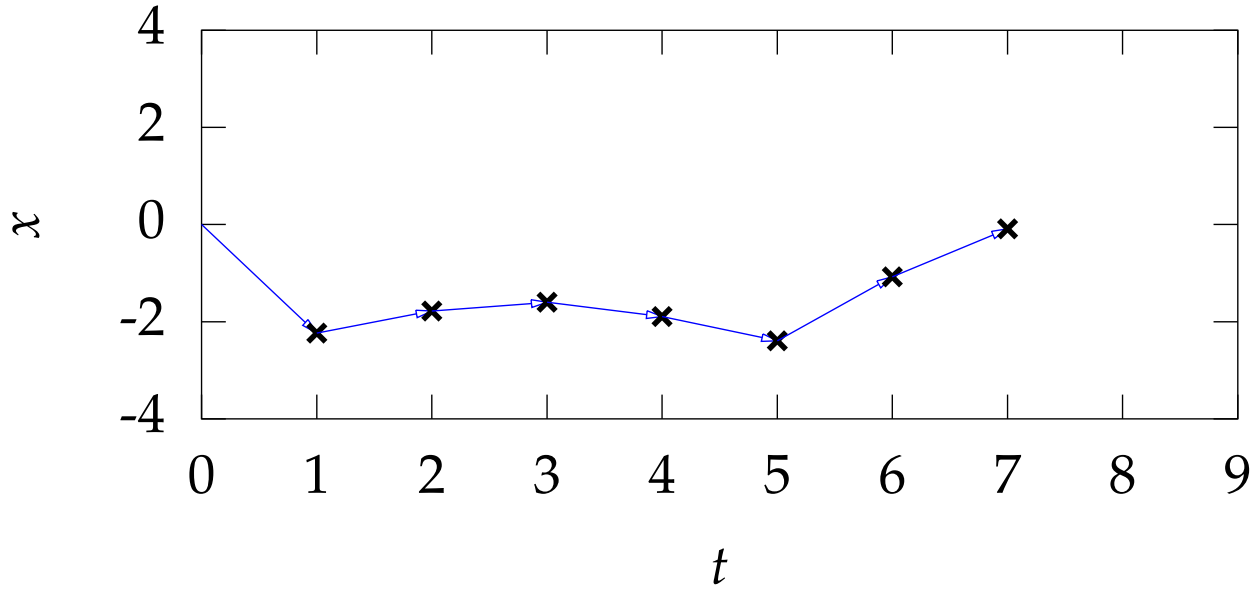


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_5 = -2.39, \quad \epsilon_6 = 1.32$$

$$x_6 = -2.39 + 1.32 = -1.08$$

Gauss Markov Chain

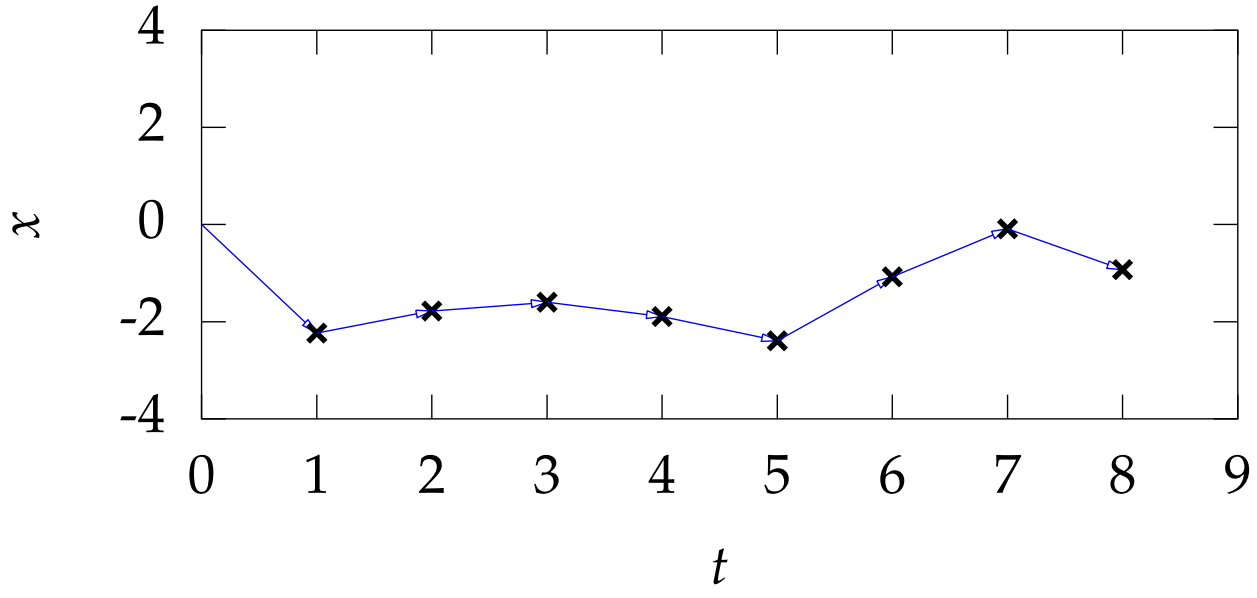


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_6 = -1.08, \quad \epsilon_7 = 0.989$$

$$x_7 = -1.08 + 0.989 = -0.0881$$

Gauss Markov Chain

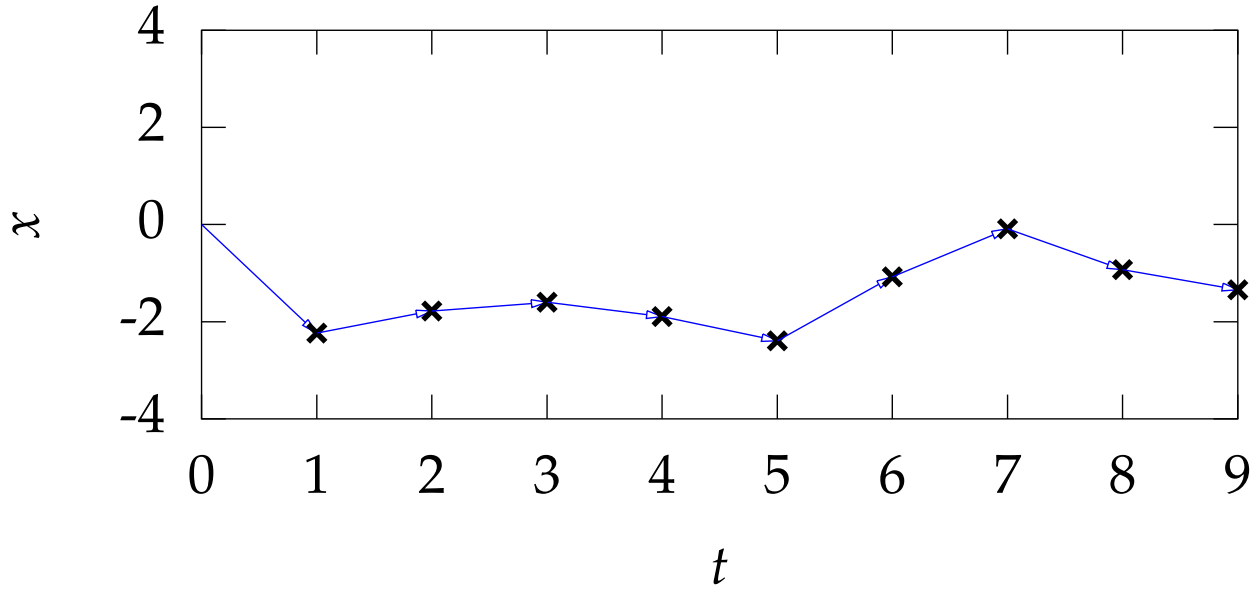


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_7 = -0.0881, \quad \epsilon_8 = -0.842$$

$$x_8 = -0.0881 - 0.842 = -0.93$$

Gauss Markov Chain



$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0,1)$$

$$x_8 = -0.93, \quad \epsilon_9 = -0.41$$

$$x_9 = -0.93 - 0.410 = -1.34$$

Multivariate Gaussian Properties: Reminder

If

$$\mathbf{z} \sim \mathcal{N}(\mu, \mathbf{C})$$

and

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \mathbf{b}$$

then

$$\mathbf{x} \sim \mathcal{N}\left(\mathbf{W}\mu + \mathbf{b}, \mathbf{W}\mathbf{C}\mathbf{W}^{\top}\right)$$

Multivariate Gaussian Properties: Reminder

Simplified: If

$$\mathbf{z} \sim \mathcal{N}\left(0, \sigma^2 \mathbf{I}\right)$$

and

$$\mathbf{x} = \mathbf{W}\mathbf{z}$$

then

$$\mathbf{x} \sim \mathcal{N}\left(0, \sigma^2 \mathbf{W}\mathbf{W}^\top\right)$$

Matrix Representation of Latent Variables

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

$$x_1 = \epsilon_1$$

Matrix Representation of Latent Variables

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

$$x_2 = \epsilon_1 + \epsilon_2$$

Matrix Representation of Latent Variables

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

$$x_3 = \epsilon_1 + \epsilon_2 + \epsilon_3$$

Matrix Representation of Latent Variables

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

$$x_4 = \epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4$$

Matrix Representation of Latent Variables

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}$$

$$x_5 = \epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4 + \epsilon_5$$

Matrix Representation of Latent Variables

$$\mathbf{x} = \mathbf{L}_1 \times \epsilon$$

Multivariate Process

- Since $\mathbf{x}$ is linearly related to $\boldsymbol{\epsilon}$ we know $\mathbf{x}$ is a Gaussian process.
- Trick: we only need to compute the mean and covariance of $\mathbf{x}$ to determine that Gaussian.

Latent Process Mean

$$\mathbf{x} = \mathbf{L}_1 \epsilon$$

Latent Process Mean

$$\langle \mathbf{x} \rangle = \langle \mathbf{L}_1 \boldsymbol{\epsilon} \rangle$$

Latent Process Mean

$$\langle \mathbf{x} \rangle = \mathbf{L}_1 \langle \boldsymbol{\epsilon} \rangle$$

Latent Process Mean

$$\langle \mathbf{x} \rangle = \mathbf{L}_1 \langle \boldsymbol{\epsilon} \rangle$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

Latent Process Mean

$$\langle \mathbf{x} \rangle = \mathbf{L}_1 \mathbf{0}$$

Latent Process Mean

$$\langle \mathbf{x} \rangle = \mathbf{0}$$

Latent Process Covariance

$$\mathbf{x}\mathbf{x}^\top = \mathbf{L}_1 \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top \mathbf{L}_1^\top$$

$$\mathbf{x}^\top = \boldsymbol{\epsilon}^\top \mathbf{L}^\top$$

Latent Process Covariance

$$\left\langle \mathbf{x}\mathbf{x}^\top \right\rangle = \left\langle \mathbf{L}_1 \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top \mathbf{L}_1^\top \right\rangle$$

Latent Process Covariance

$$\left\langle \mathbf{x}\mathbf{x}^\top \right\rangle = \mathbf{L}_1 \left\langle \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top \right\rangle \mathbf{L}_1^\top$$

Latent Process Covariance

$$\left\langle \mathbf{x}\mathbf{x}^\top \right\rangle = \mathbf{L}_1 \left\langle \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top \right\rangle \mathbf{L}_1^\top$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \alpha \mathbf{I}\right)$$

Latent Process Covariance

$$\left\langle \mathbf{x}\mathbf{x}^{\top} \right\rangle = \alpha \mathbf{L}_1 \mathbf{L}_1^{\top}$$

Latent Process

$$\mathbf{x} = \mathbf{L}_1 \epsilon$$

Latent Process

$$\mathbf{x} = \mathbf{L}_1 \boldsymbol{\epsilon}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

Latent Process

$$\mathbf{x} = \mathbf{L}_1 \boldsymbol{\epsilon}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

$$\Longrightarrow$$

Latent Process

$$\mathbf{x} = \mathbf{L}_1 \boldsymbol{\epsilon}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha \mathbf{I})$$

$$\Longrightarrow$$

$$\mathbf{x} \sim \mathcal{N}\left(\mathbf{0}, \alpha \mathbf{L}_1 \mathbf{L}_1^\top\right)$$

Covariance for Latent Process II

- Make the variance dependent on time interval.
- Assume variance grows *linearly* with time.
- Justification: sum of two Gaussian distributed random variables is distributed as Gaussian with sum of variances.
- If variable's movement is additive over time (as described) variance scales linearly with time.

Covariance for Latent Process II

- Given
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I}) \Longrightarrow \boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \alpha\mathbf{L_1}\mathbf{L_1}^\top\right).$$
Then
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \Delta t\alpha\mathbf{I}) \Longrightarrow \boldsymbol{\epsilon} \sim \mathcal{N}\left(\mathbf{0}, \Delta t\alpha\mathbf{L_1}\mathbf{L_1}^\top\right).$$
where Δt is the time interval between observations.

Covariance for Latent Process II

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \alpha \Delta t \mathbf{I}\right), \quad \mathbf{x} \sim \mathcal{N}\left(0, \alpha \Delta t \mathbf{L_1} \mathbf{L_1}^\top\right)$$

Covariance for Latent Process II

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \alpha \Delta t \mathbf{I}\right), \quad \mathbf{x} \sim \mathcal{N}\left(0, \alpha \Delta t \mathbf{L_1} \mathbf{L_1}^\top\right)$$

$$\mathbf{K} = \alpha \Delta t \mathbf{L_1} \mathbf{L_1}^\top$$

Covariance for Latent Process II

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \alpha\Delta t \mathbf{I}\right), \quad \mathbf{x} \sim \mathcal{N}\left(0, \alpha\Delta t \mathbf{L_1}\mathbf{L_1}^\top\right)$$

$$\mathbf{K} = \alpha\Delta t \mathbf{L_1}\mathbf{L_1}^\top$$

$$k_{i,j} = \alpha\Delta t \mathbf{l}_{:,i}^\top \mathbf{l}_{:,j}$$

where $\mathbf{l}_{:,k}$ is a vector from the kth row of $\mathbf{L_1}$: the first k elements are one, the next $T - k$ are zero.

Covariance for Latent Process II

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \alpha\Delta t \mathbf{I}\right), \quad \mathbf{x} \sim \mathcal{N}\left(0, \alpha\Delta t \mathbf{L_1}\mathbf{L_1}^\top\right)$$

$$\mathbf{K} = \alpha\Delta t \mathbf{L_1}\mathbf{L_1}^\top$$

$$k_{i,j} = \alpha\Delta t \mathbf{l}_{:,i}^\top \mathbf{l}_{:,j}$$

where $\mathbf{l}_{:,k}$ is a vector from the kth row of $\mathbf{L_1}$: the first k elements are one, the next $T - k$ are zero.

$$k_{i,j} = \alpha\Delta t \min(i, j)$$

define $\Delta t i = t_i$ so

$$k_{i,j} = \alpha \min(t_i, t_j) = k(t_i, t_j)$$

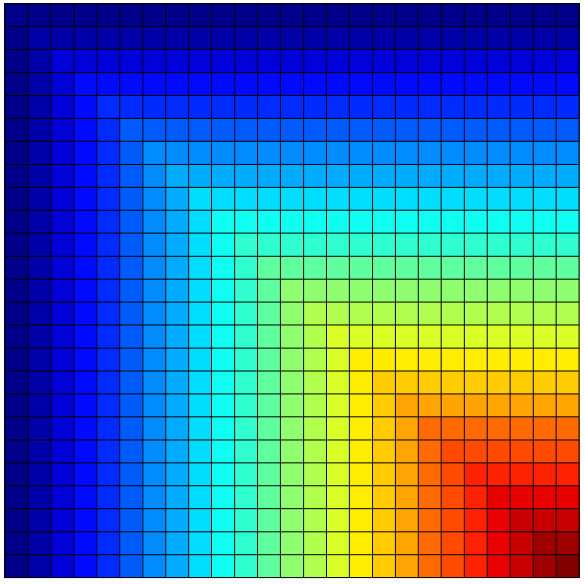
Covariance Functions

Where did this covariance matrix come from?

Markov Process

$$k(t, t') = \alpha \min(t, t')$$

- Covariance matrix is built using the *inputs* to the function t.



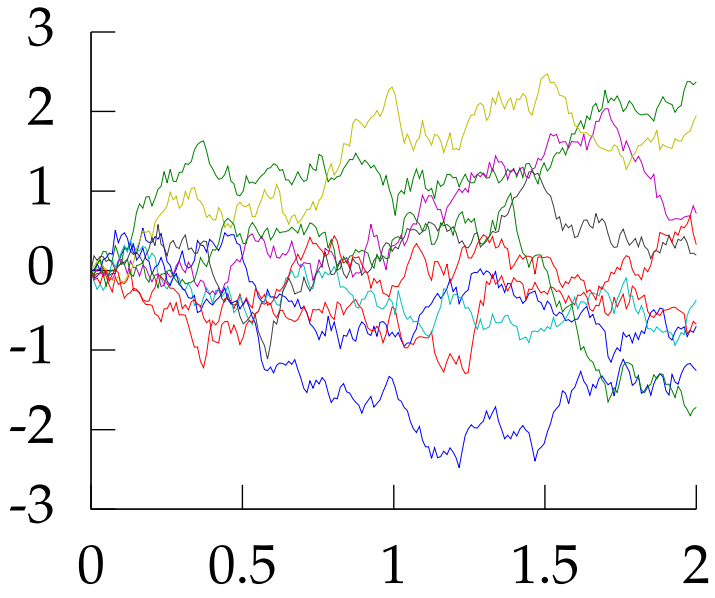
Covariance Functions

Where did this covariance matrix come from?

Markov Process

$$k(t, t') = \alpha \min(t, t')$$

- Covariance matrix is built using the *inputs* to the function t.



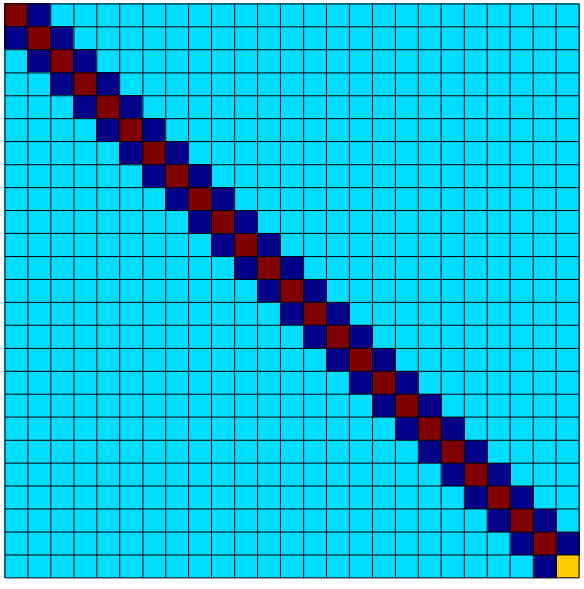
Covariance Functions

Where did this covariance matrix come from?

Markov Process

Visualization of inverse covariance (precision).

- Precision matrix is sparse: only neighbours in matrix are non-zero.
- This reflects *conditional* independencies in data.
- In this case *Markov* structure.



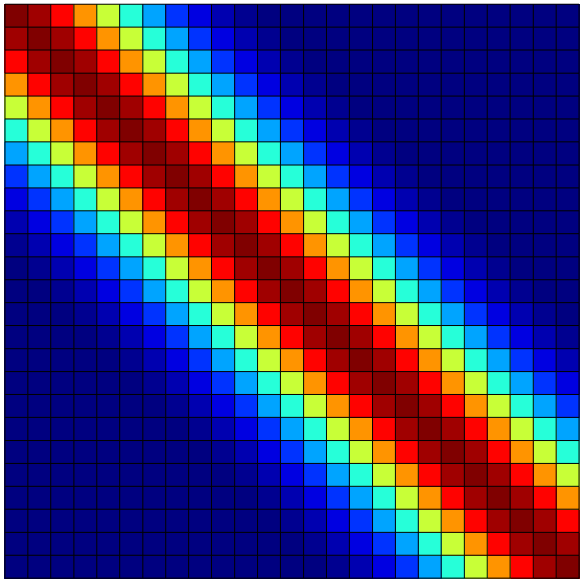
Covariance Functions

Where did this covariance matrix come from?

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k(\mathbf{x}, \mathbf{x}') = \alpha \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\ell^2}\right)$$

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



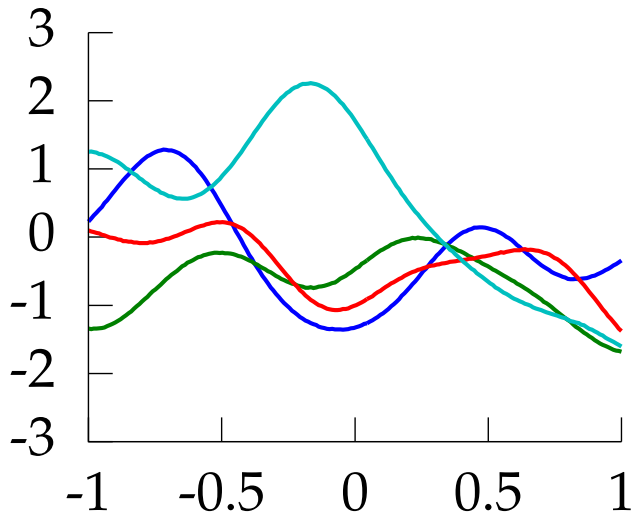
Covariance Functions

Where did this covariance matrix come from?

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k(\mathbf{x}, \mathbf{x}') = \alpha \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\ell^2}\right)$$

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



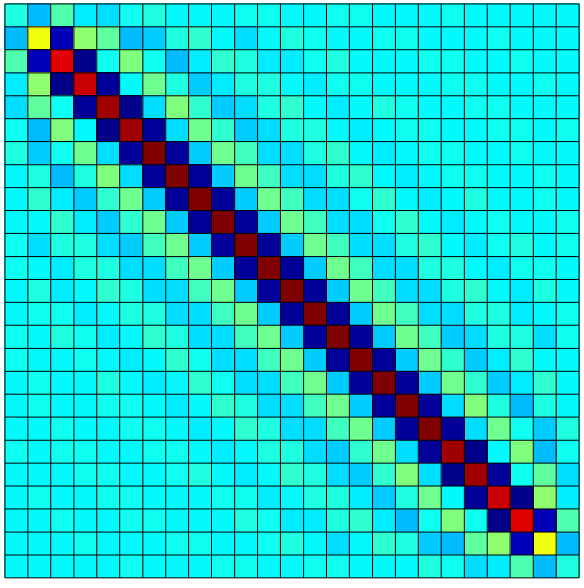
Covariance Functions

Where did this covariance matrix come from?

Exponentiated Quadratic

Visualization of inverse covariance (precision).

- Precision matrix is not sparse.
- Each point is dependent on all the others.
- In this case non-Markovian.



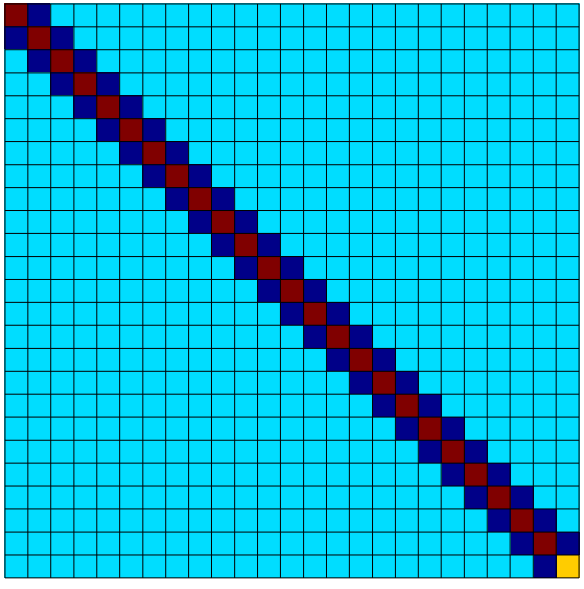
Covariance Functions

Where did this covariance matrix come from?

Markov Process

Visualization of inverse covariance (precision).

- Precision matrix is sparse: only neighbours in matrix are non-zero.
- This reflects *conditional* independencies in data.
- In this case *Markov* structure.



Simple Kalman Filter I

- We have state vector $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \dots \mathbf{x}_q \end{bmatrix} \in \mathbb{R}^{T \times q}$ and if each state evolves independently we have

$$p(\mathbf{X}) = \prod_{i=1}^{q} p(\mathbf{x}_{:,i})$$
$$p(\mathbf{x}_{:,i}) = \mathcal{N}\left(\mathbf{x}_{:,i} | \mathbf{0}, \mathbf{K}\right).$$

- We want to obtain outputs through:

$$\mathbf{y}_{i,:} = \mathbf{W}\mathbf{x}_{i,:}$$

Stacking and Kronecker Products I

- Represent with a 'stacked' system:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{I} \otimes \mathbf{K})$$

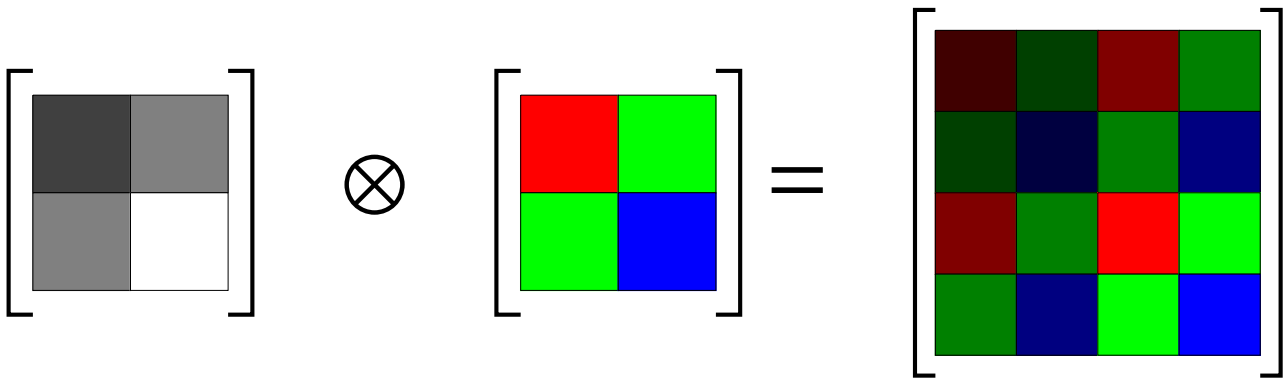
where the stacking is placing each column of $\mathbf{X}$ one on top of another as

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{:,1} \\ \mathbf{x}_{:,2} \\ \vdots \\ \mathbf{x}_{:,q} \end{bmatrix}$$

Kronecker Product

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \otimes \mathbf{K} = \begin{bmatrix} a\mathbf{K} & b\mathbf{K} \\ c\mathbf{K} & d\mathbf{K} \end{bmatrix}$$

Kronecker Product



Stacking and Kronecker Products I

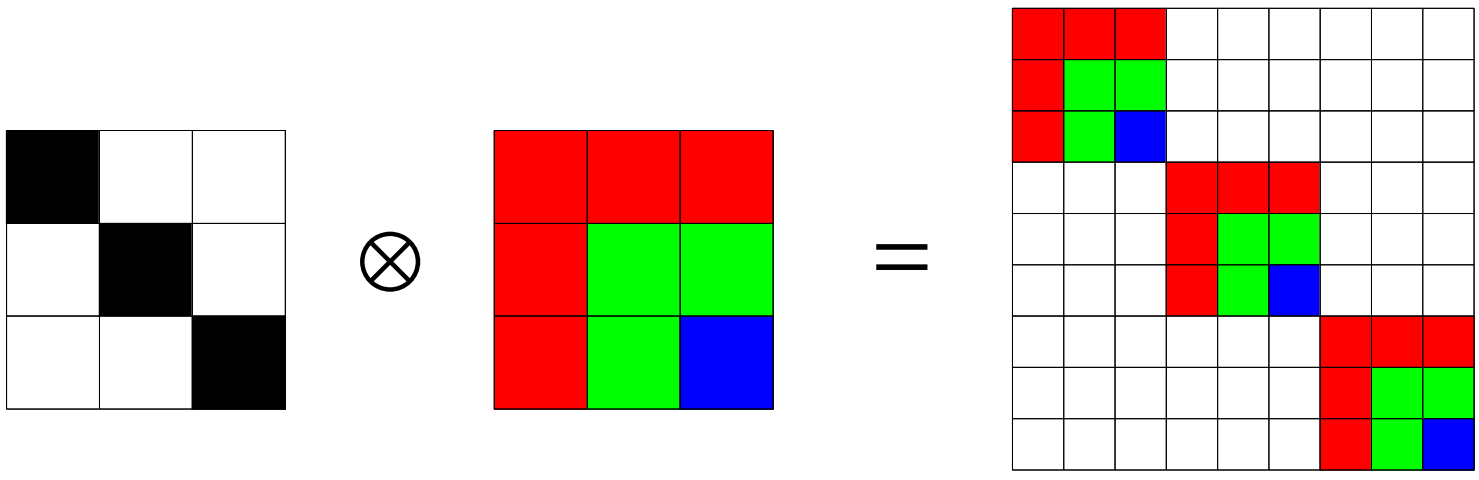
- Represent with a 'stacked' system:

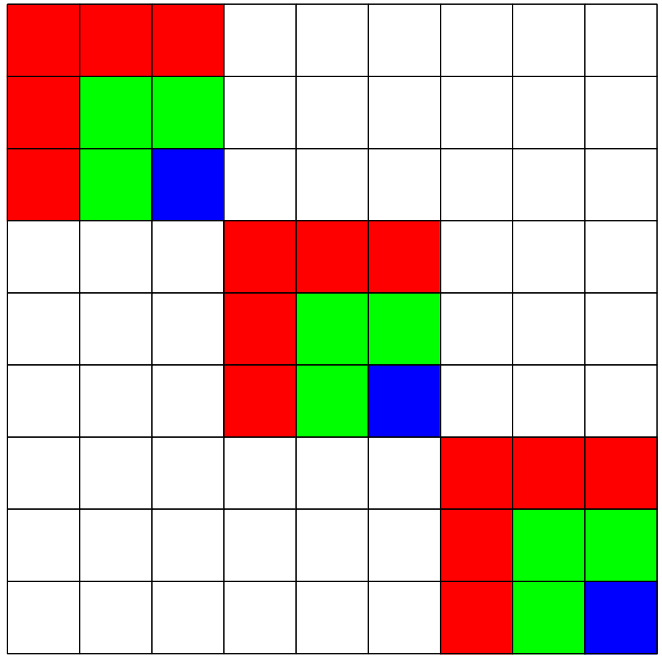
$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{I} \otimes \mathbf{K})$$

where the stacking is placing each column of $\mathbf{X}$ one on top of another as

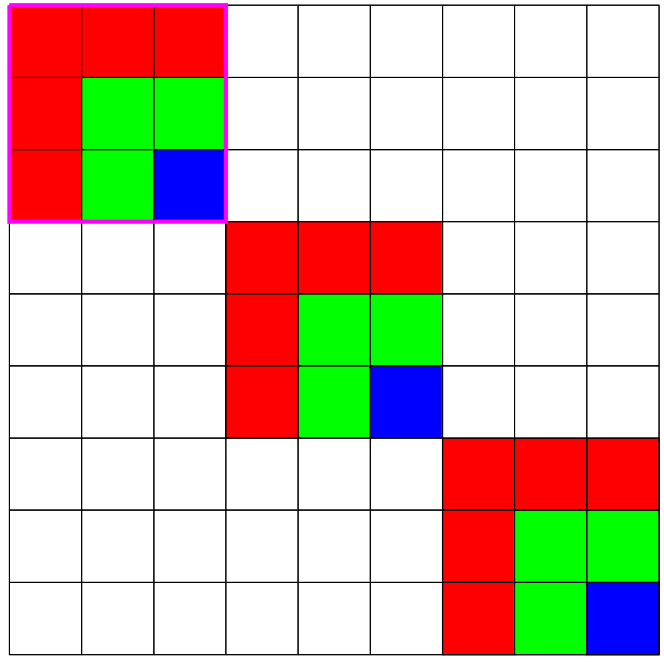
$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{:,1} \\ \mathbf{x}_{:,2} \\ \vdots \\ \mathbf{x}_{:,q} \end{bmatrix}$$

Column Stacking

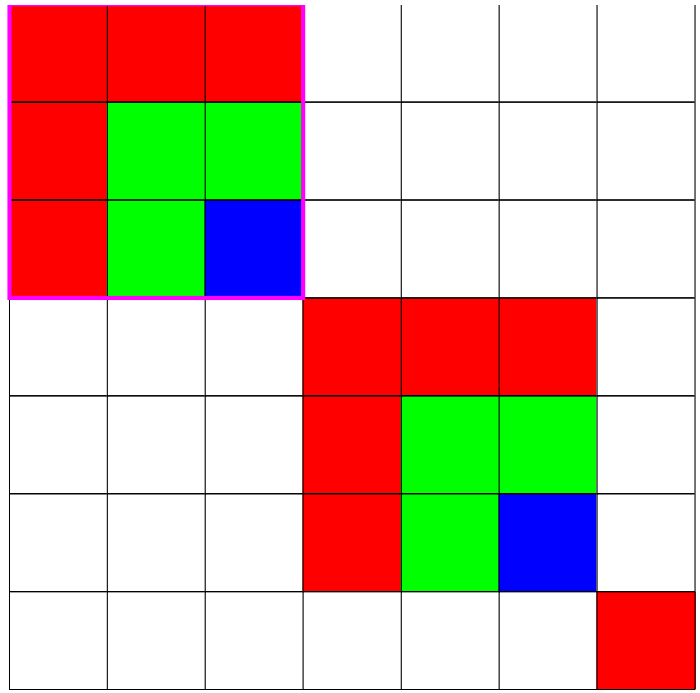




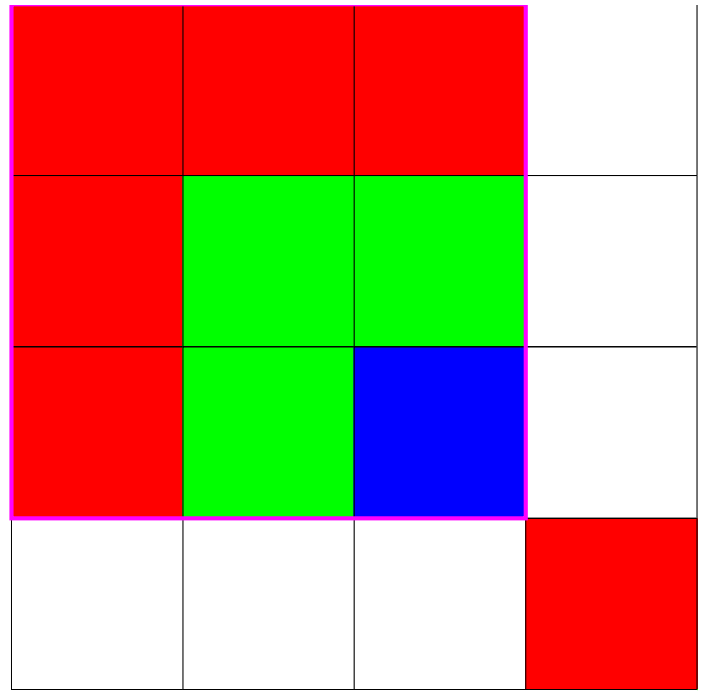
For this stacking the marginal distribution over *time* is given by the block diagonals.



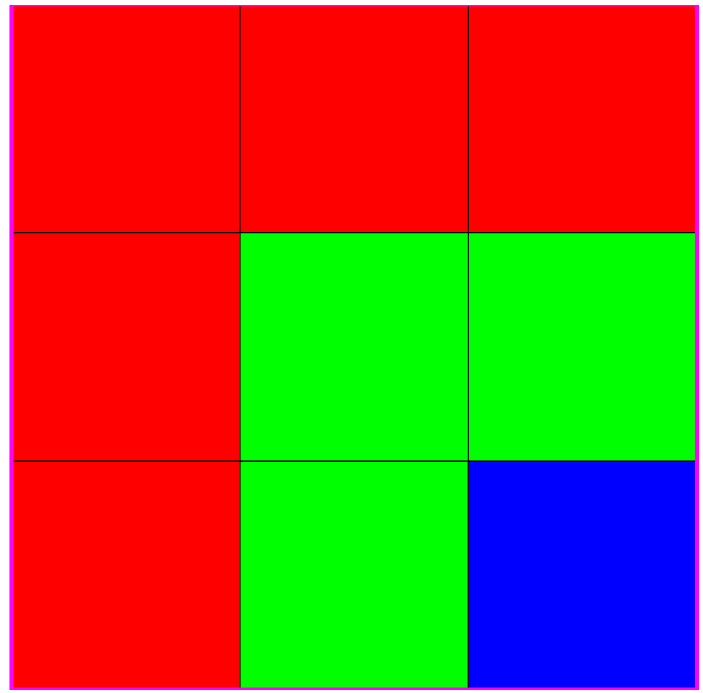
For this stacking the marginal distribution over *time* is given by the block diagonals.



For this stacking the marginal distribution over *time* is given by the block diagonals.



For this stacking the marginal distribution over *time* is given by the block diagonals.



For this stacking the marginal distribution over *time* is given by the block diagonals.

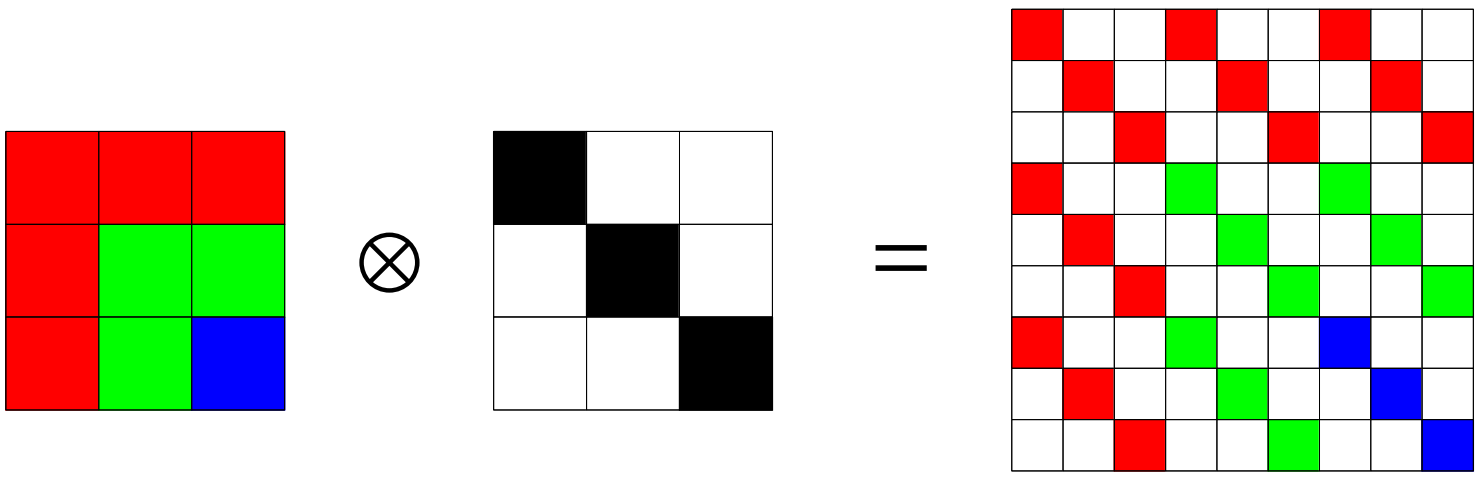
Two Ways of Stacking

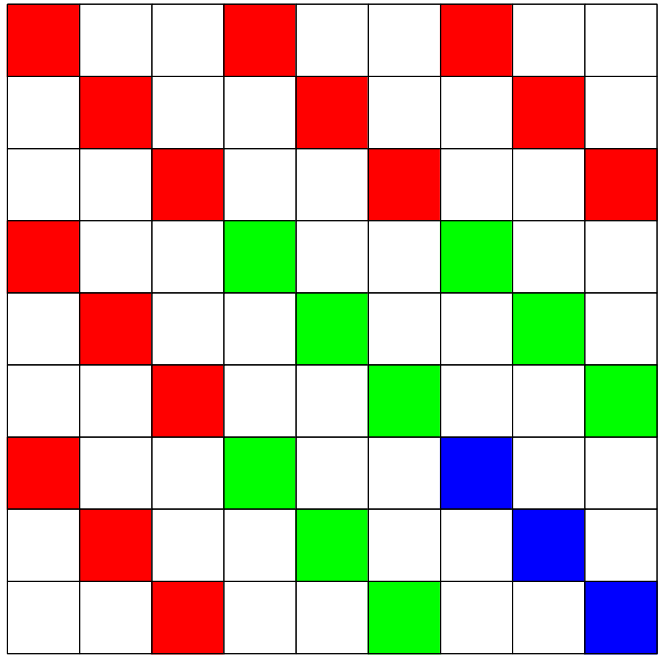
Can also stack each row of $\mathbf{X}$ to form column vector:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{1,:} \\ \mathbf{x}_{2,:} \\ \vdots \\ \mathbf{x}_{T,:} \end{bmatrix}$$

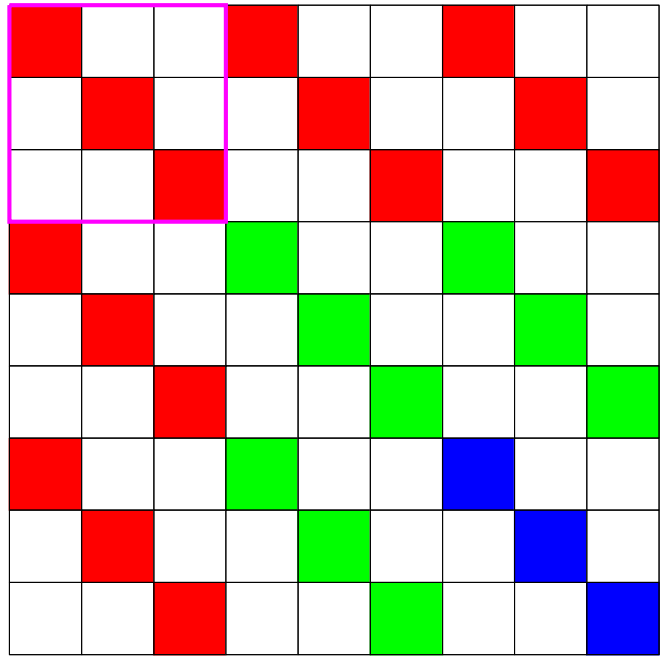
$$p(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}|\mathbf{0}, \mathbf{K} \otimes \mathbf{I}\right)$$

Row Stacking

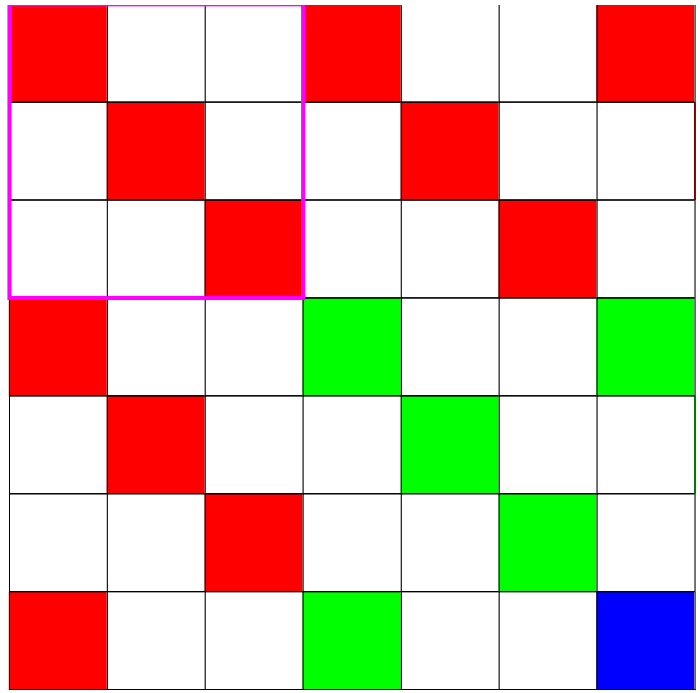




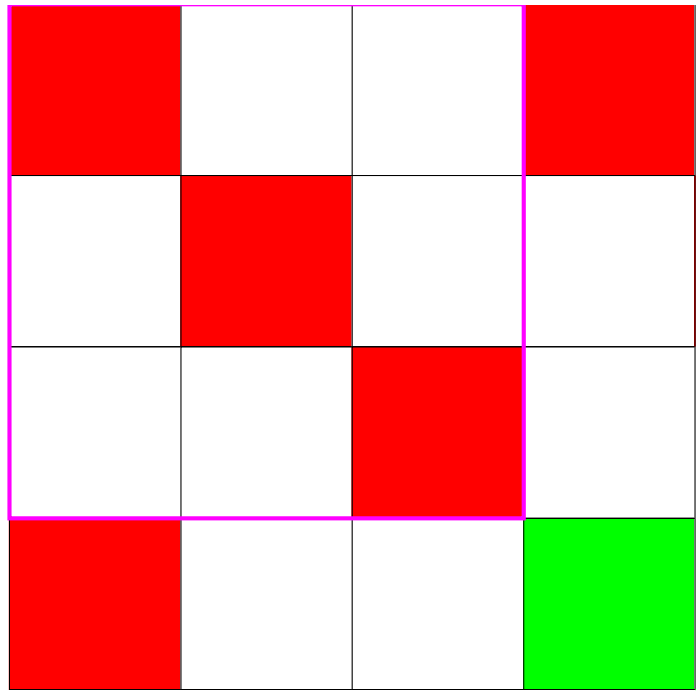
For this stacking the marginal distribution over the latent *dimensions* is given by the block diagonals.



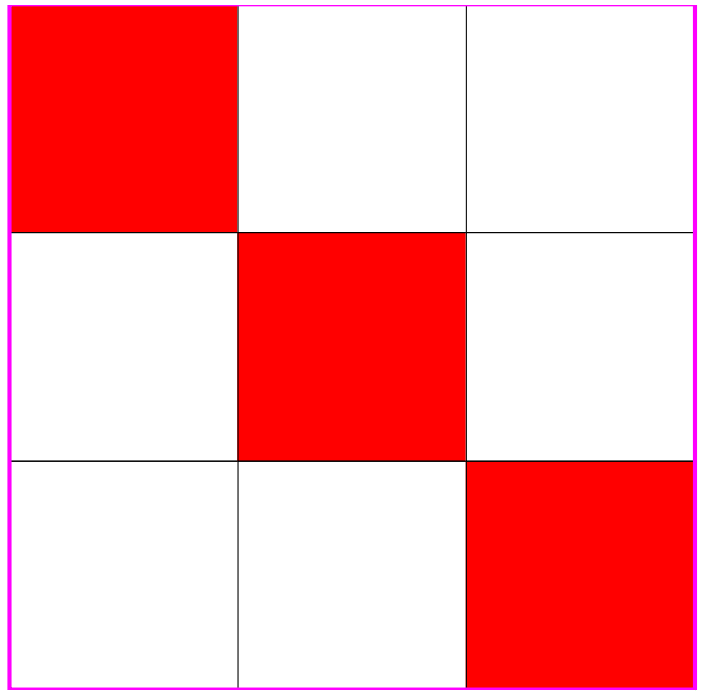
For this stacking the marginal distribution over the latent *dimensions* is given by the block diagonals.



For this stacking the marginal distribution over the latent *dimensions* is given by the block diagonals.



For this stacking the marginal distribution over the latent *dimensions* is given by the block diagonals.



For this stacking the marginal distribution over the latent *dimensions* is given by the block diagonals.

Observed Process

The observations are related to the latent points by a linear mapping matrix,

$$\mathbf{y}_{i,:} = \mathbf{W}\mathbf{x}_{i,:} + \boldsymbol{\epsilon}_{i,:}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}\left(0, \sigma^2\mathbf{I}\right)$$

Mapping from Latent Process to Observed

$$\begin{bmatrix} \mathbf{W} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{W} \end{bmatrix} \times \begin{bmatrix} \mathbf{x}_{1,:} \\ \mathbf{x}_{2,:} \\ \mathbf{x}_{3,:} \end{bmatrix} = \begin{bmatrix} \mathbf{W}\mathbf{x}_{1,:} \\ \mathbf{W}\mathbf{x}_{2,:} \\ \mathbf{W}\mathbf{x}_{3,:} \end{bmatrix}$$

Output Covariance

This leads to a covariance of the form

$$(\mathbf{I} \otimes \mathbf{W})(\mathbf{K} \otimes \mathbf{I})(\mathbf{I} \otimes \mathbf{W}^\top) + \mathbf{I}\sigma^2$$

Using $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$ This leads to

$$\mathbf{K} \otimes \mathbf{W}\mathbf{W}^\top + \mathbf{I}\sigma^2$$

or

$$\mathbf{y} \sim \mathcal{N}\left(0, \mathbf{W}\mathbf{W}^\top \otimes \mathbf{K} + \mathbf{I}\sigma^2\right)$$

Foundations and Trends® in
Machine Learning
Vol. 4, No. 3 (2011) 195–266

DOI: 10.1561/2200000036

now
the essence of knowledge

Kernels for Vector-Valued Functions: A Review

By Mauricio A. Álvarez,
Lorenzo Rosasco and Neil D. Lawrence

Kronecker Structure GPs

- This Kronecker structure leads to several published models.

$$(\mathbf{K}(\mathbf{x}, \mathbf{x}'))_{j,j'} = k(\mathbf{x}, \mathbf{x}')k_T(j, j'),$$

  where k has $\mathbf{x}$ and k_T has i as inputs.
- Can think of multiple output covariance functions as covariances with augmented input.
- Alongside $\mathbf{x}$ we also input the j associated with the *output* of interest.

Separable Covariance Functions

- Taking $\mathbf{B} = \mathbf{W}\mathbf{W}^\top$ we have a matrix expression across outputs.

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')\mathbf{B},$$

  where $\mathbf{B}$ is a $p \times p$ symmetric and positive semi-definite matrix.
- $\mathbf{B}$ is called the *coregionalization* matrix.
- We call this class of covariance functions *separable* due to their product structure.

Sum of Separable Covariance Functions

- In the same spirit a more general class of kernels is given by

$$\mathbf{K}(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{q} k_j(\mathbf{x}, \mathbf{x}')\mathbf{B}_j.$$

- This can also be written as

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \sum_{j=1}^{q} \mathbf{B}_j \otimes k_j(\mathbf{X}, \mathbf{X}),$$

- This is like several Kalman filter-type models added together, but each one with a different set of latent functions.
- We call this class of kernels sum of separable kernels (SoS kernels).

Geostatistics

- Use of GPs in Geostatistics is called kriging.
- These multi-output GPs pioneered in geostatistics: prediction over vector-valued output data is known as *cokriging*.
- The model in geostatistics is known as the *linear model of coregionalization* (LMC, Journel and Huijbregts (1978); Goovaerts (1997)).
- Most machine learning multitask models can be placed in the context of the LMC model.

Weighted sum of Latent Functions

- In the linear model of coregionalization (LMC) outputs are expressed as linear combinations of independent random functions.
- In the LMC, each component f_j is expressed as a linear sum

$$f_j(\mathbf{x}) = \sum_{j=1}^{q} w_{j,j} u_j(\mathbf{x}).$$

  where the latent functions are independent and have covariance functions $k_j(\mathbf{x}, \mathbf{x}')$.
- The processes $\{f_j(\mathbf{x})\}_{j=1}^{q}$ are independent for $q \neq j'$.

Kalman Filter Special Case

- The Kalman filter is an example of the LMC where $u_i(\mathbf{x}) \rightarrow x_i(t)$.
- I.e. we've moved form time input to a more general input space.
- In matrix notation:
  1. Kalman filter
  $$\mathbf{F} = \mathbf{W}\mathbf{X}$$
  2. LMC
  $$\mathbf{F} = \mathbf{W}\mathbf{U}$$

  where the rows of these matrices $\mathbf{F}$, $\mathbf{X}$, $\mathbf{U}$ each contain q samples from their corresponding functions at a different time (Kalman filter) or spatial location (LMC).

Intrinsic Coregionalization Model

- If one covariance used for latent functions (like in Kalman filter).
- This is called the intrinsic coregionalization model (ICM, Goovaerts (1997)).
- The kernel matrix corresponding to a dataset $\mathbf{X}$ takes the form

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

Autokrigeability

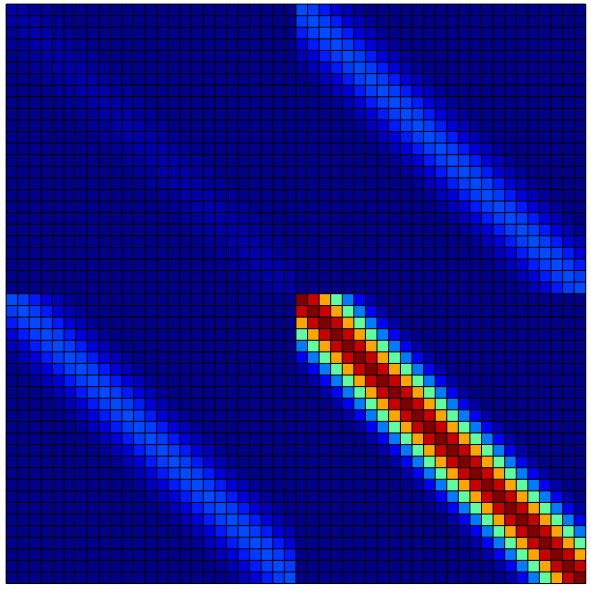
- If outputs are noise-free, maximum likelihood is equivalent to independent fits of $\mathbf{B}$ and $k(\mathbf{x}, \mathbf{x}')$ (Helterbrand and Cressie, 1994).
- In geostatistics this is known as autokrigeability (Wackernagel, 2003).
- In multitask learning its the cancellation of intertask transfer (Bonilla et al., 2008).

Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{w}\mathbf{w}^\top \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{w} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$

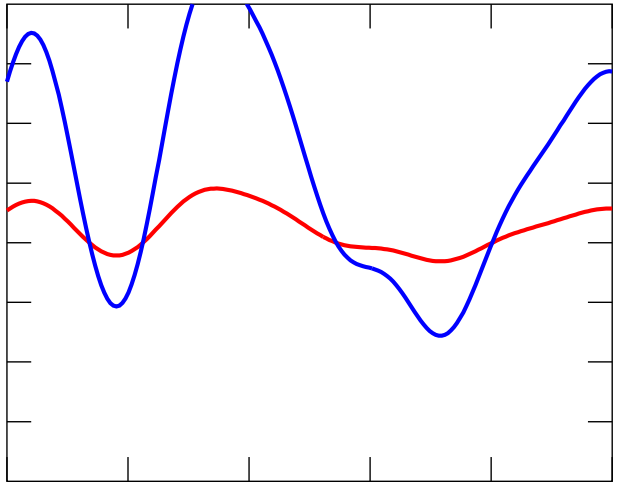


Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{w}\mathbf{w}^\top \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{w} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$

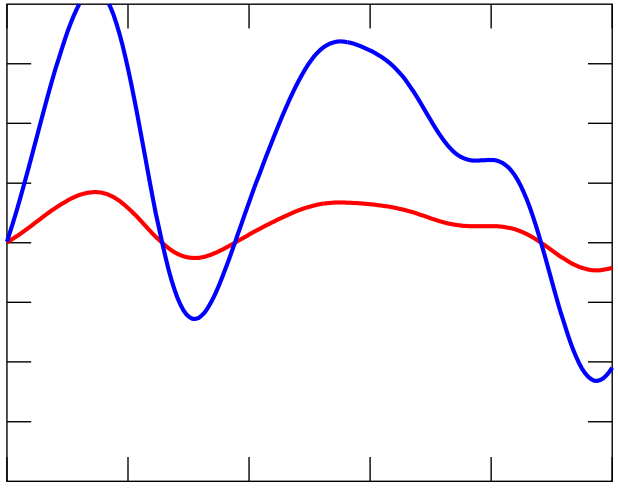


Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{w}\mathbf{w}^\top \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{w} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$

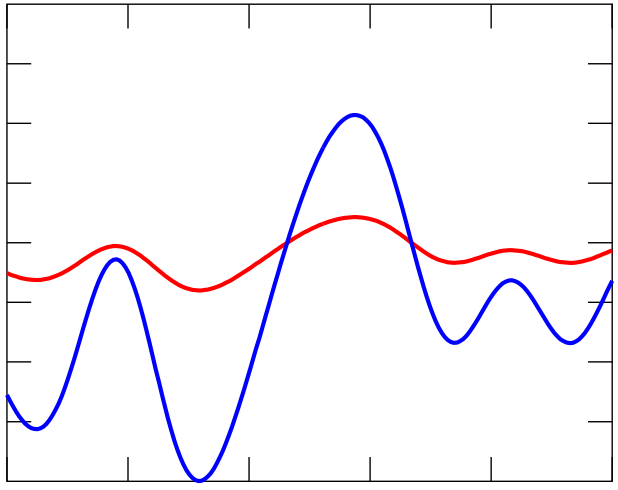


Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{w}\mathbf{w}^\top \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{w} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$

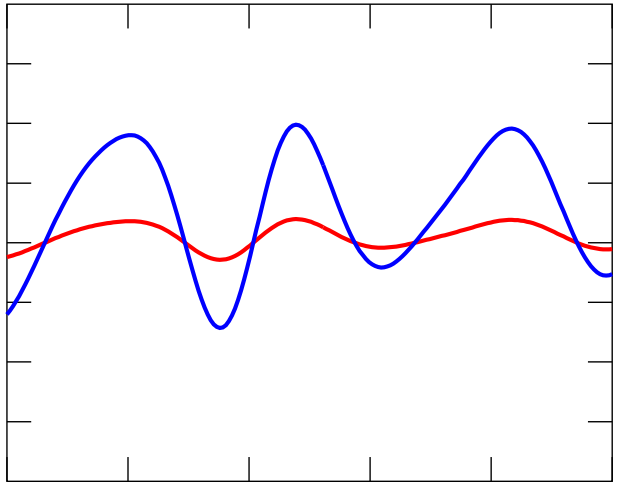


Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{w}\mathbf{w}^\top \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{w} = \begin{bmatrix} 1 \\ 5 \end{bmatrix}$$

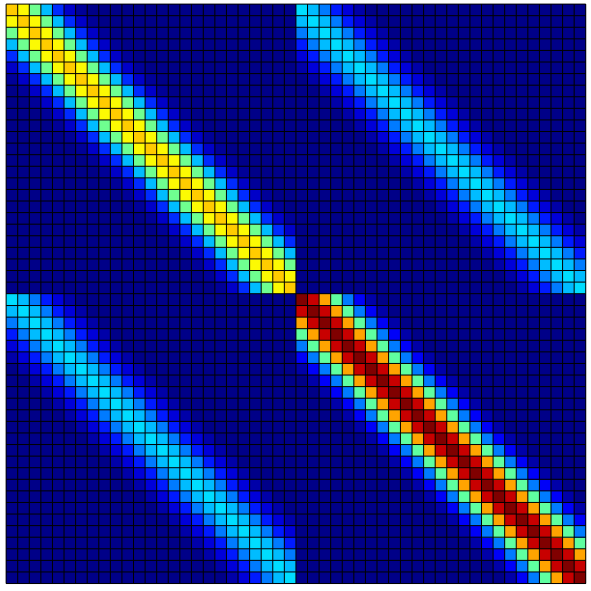
$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$



Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

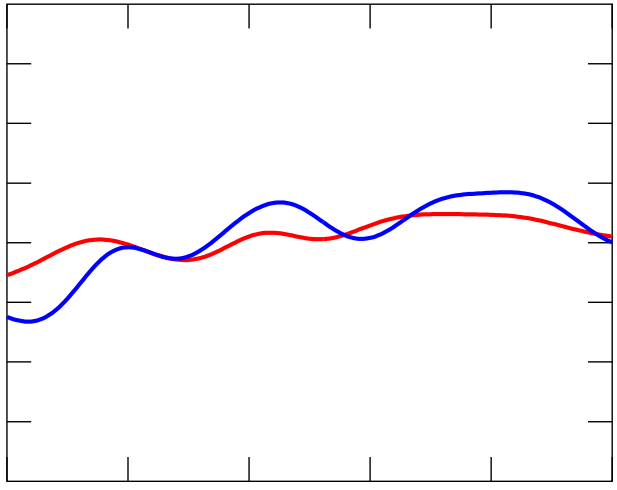
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

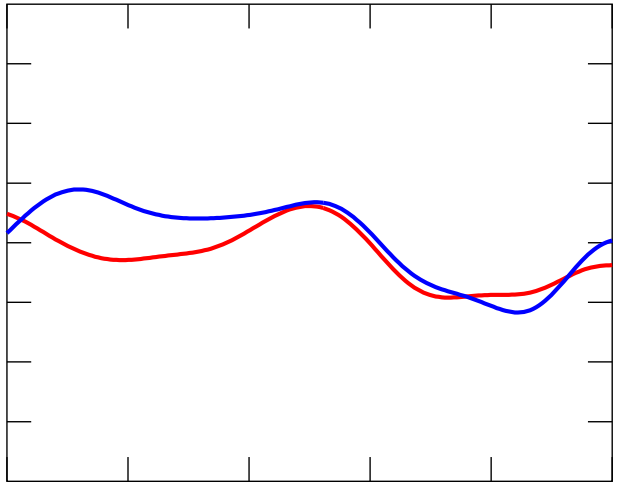
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

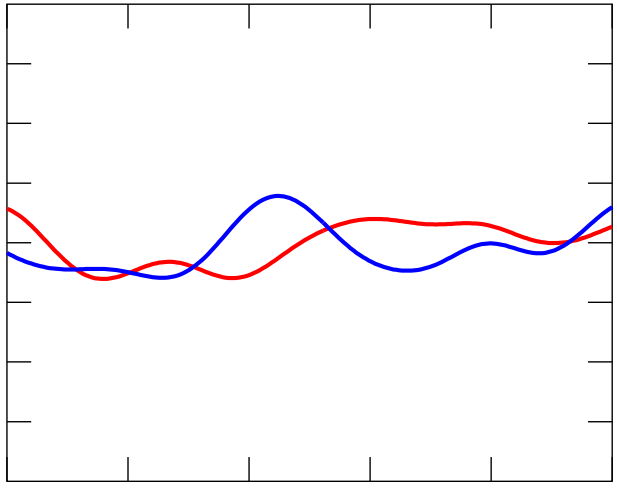
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

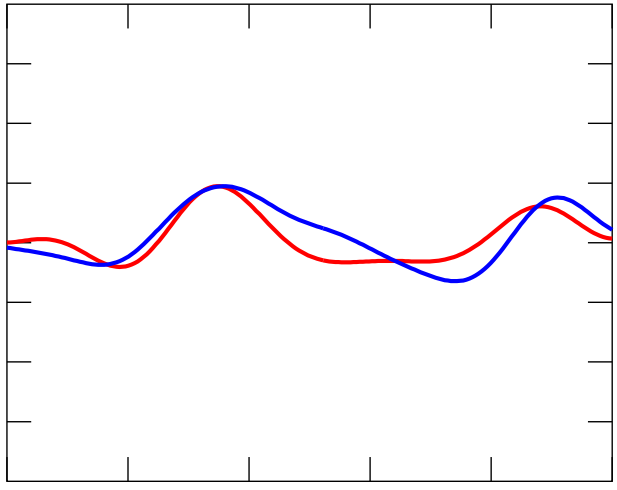
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B} \otimes k(\mathbf{X}, \mathbf{X}).$$

$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



LMC Samples

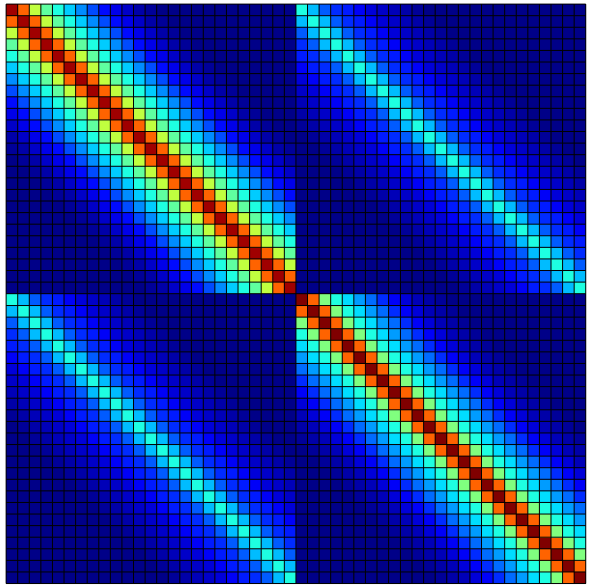
$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{B}_1 = \begin{bmatrix} 1.4 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$$

$$\ell_1 = 1$$

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.3 \end{bmatrix}$$

$$\ell_2 = 0.2$$



LMC Samples

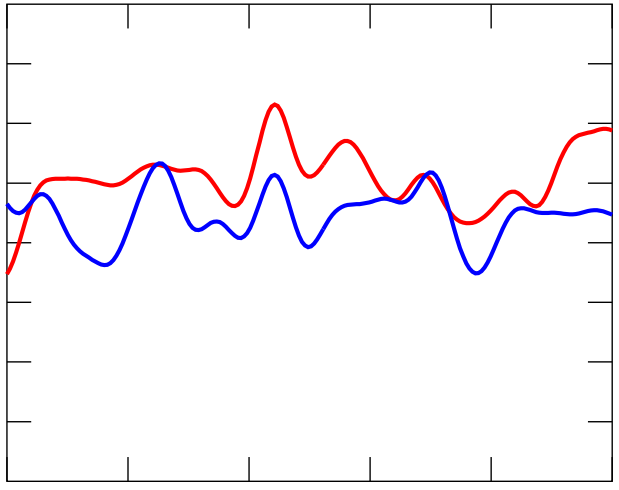
$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{B}_1 = \begin{bmatrix} 1.4 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$$

$$\ell_1 = 1$$

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.3 \end{bmatrix}$$

$$\ell_2 = 0.2$$



LMC Samples

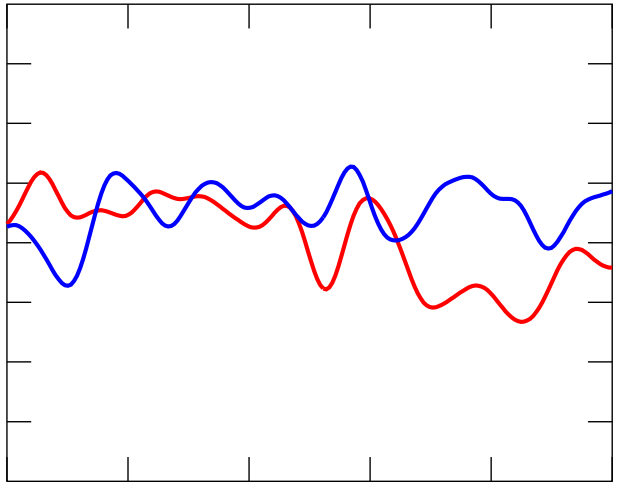
$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{B}_1 = \begin{bmatrix} 1.4 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$$

$$\ell_1 = 1$$

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.3 \end{bmatrix}$$

$$\ell_2 = 0.2$$



LMC Samples

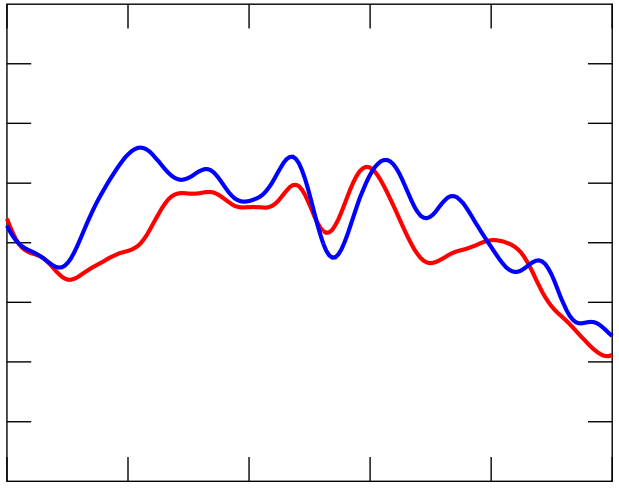
$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{B}_1 = \begin{bmatrix} 1.4 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$$

$$\ell_1 = 1$$

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.3 \end{bmatrix}$$

$$\ell_2 = 0.2$$



LMC Samples

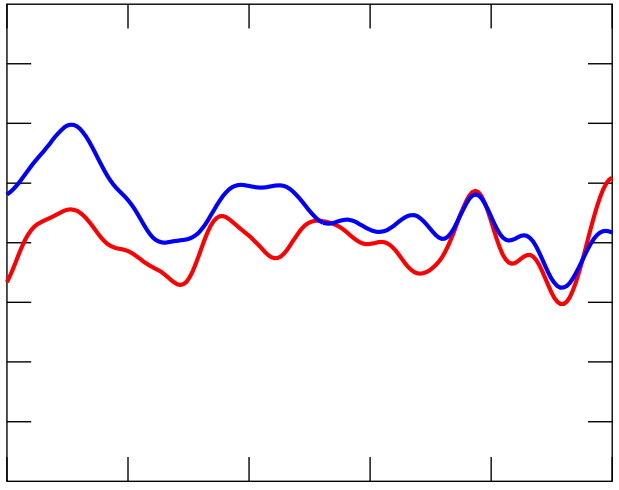
$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{B}_1 = \begin{bmatrix} 1.4 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$$

$$\ell_1 = 1$$

$$\mathbf{B}_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.3 \end{bmatrix}$$

$$\ell_2 = 0.2$$



LMC in Machine Learning and Statistics

- Used in machine learning for GPs for multivariate regression and in statistics for computer emulation of expensive multivariate computer codes.
- Imposes the correlation of the outputs explicitly through the set of coregionalization matrices.
- Setting $\mathbf{B} = \mathbf{I}_p$ assumes outputs are conditionally independent given the parameters $\boldsymbol{\theta}$. (Minka and Picard, 1997; Lawrence and Platt, 2004; Yu et al., 2005).
- More recent approaches for multiple output modeling are different versions of the linear model of coregionalization.

Semiparametric Latent Factor Model

- Coregionalization matrices are rank 1 Teh et al. (2005). rewrite equation (**??**) as

$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \sum_{j=1}^{q} \mathbf{w}_{:,j}\mathbf{w}_{:,j}^{\top} \otimes k_j(\mathbf{X}, \mathbf{X}).$$

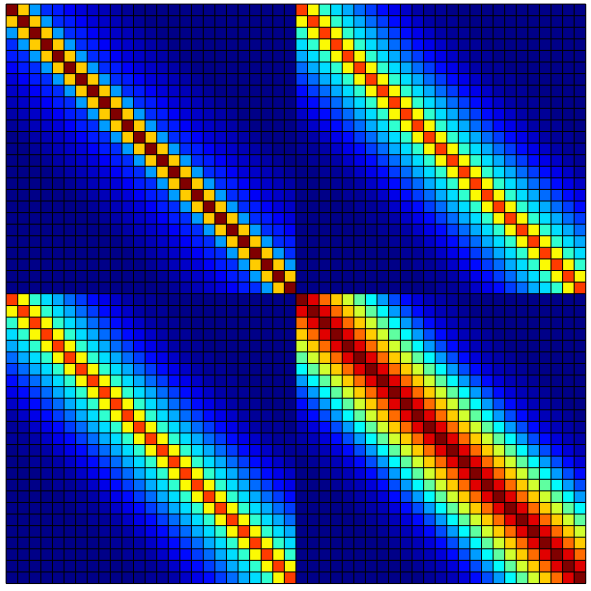
- Like the Kalman filter, but each latent function has a *different* covariance.
- Authors suggest using an exponentiated quadratic characteristic length-scale for each input dimension.

Semiparametric Latent Factor Model Samples

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{w}_{:,1}\mathbf{w}_{:,1}^{\top} \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{w}_{:,2}\mathbf{w}_{:,2}^{\top} \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{w}_1 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

$$\mathbf{w}_2 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$

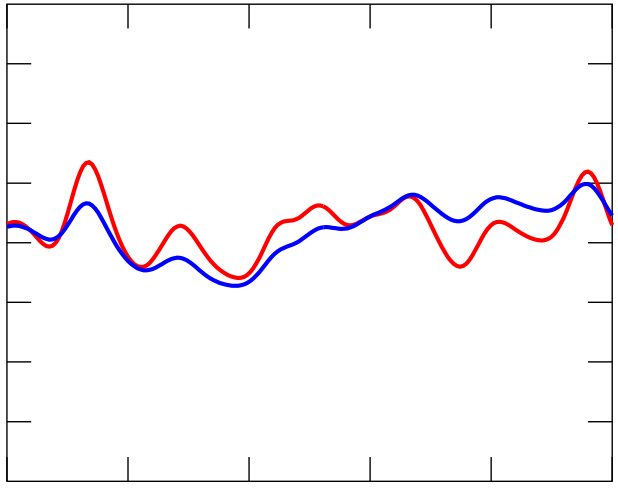


Semiparametric Latent Factor Model Samples

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{w}_{:,1}\mathbf{w}_{:,1}^{\top} \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{w}_{:,2}\mathbf{w}_{:,2}^{\top} \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{w}_1 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

$$\mathbf{w}_2 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$

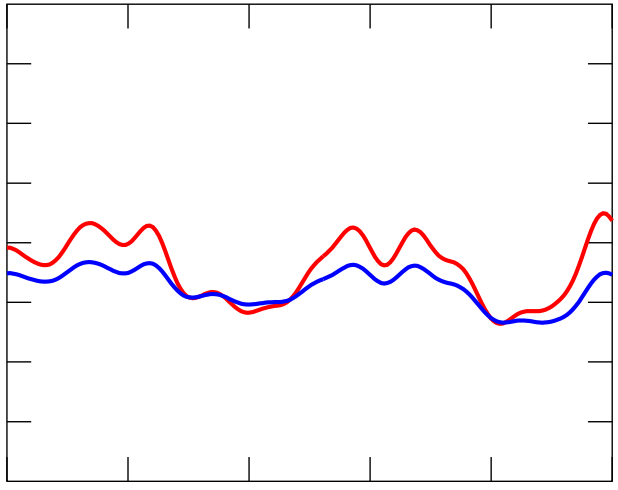


Semiparametric Latent Factor Model Samples

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{w}_{:,1}\mathbf{w}_{:,1}^{\top} \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{w}_{:,2}\mathbf{w}_{:,2}^{\top} \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{w}_1 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

$$\mathbf{w}_2 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$

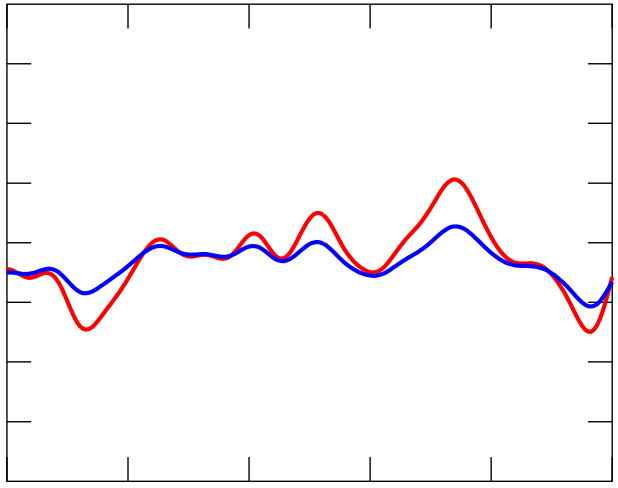


Semiparametric Latent Factor Model Samples

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{w}_{:,1}\mathbf{w}_{:,1}^{\top} \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{w}_{:,2}\mathbf{w}_{:,2}^{\top} \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{w}_1 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

$$\mathbf{w}_2 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$

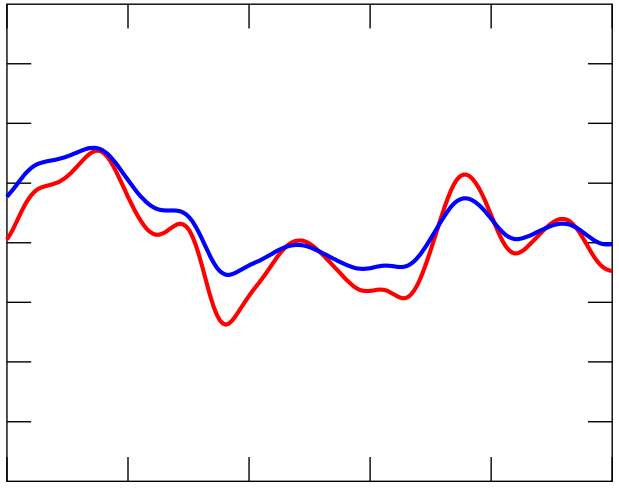


Semiparametric Latent Factor Model Samples

$$\mathbf{K}(\mathbf{X},\mathbf{X}) = \mathbf{w}_{:,1}\mathbf{w}_{:,1}^{\top} \otimes k_1(\mathbf{X},\mathbf{X}) + \mathbf{w}_{:,2}\mathbf{w}_{:,2}^{\top} \otimes k_2(\mathbf{X},\mathbf{X})$$

$$\mathbf{w}_1 = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

$$\mathbf{w}_2 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$



Gaussian processes for Multi-task, Multi-output and Multi-class

- Bonilla et al. (2008) suggest ICM for multitask learning.
- Use a PPCA form for $\mathbf{B}$: similar to our Kalman filter example.
- Refer to the autokrigeability effect as the cancellation of inter-task transfer.
- Also discuss the similarities between the multi-task GP and the ICM, and its relationship to the SLFM and the LMC.

Multitask Classification

- Mostly restricted to the case where the outputs are conditionally independent given the hyperparameters ϕ (Minka and Picard, 1997; **?**; Lawrence and Platt, 2004; Seeger and Jordan, 2004; Yu et al., 2005; Rasmussen and Williams, 2006).
- Intrinsic coregionalization model has been used in the multiclass scenario. Skolidis and Sanguinetti (2011) use the intrinsic coregionalization model for classification, by introducing a probit noise model as the likelihood.
- Posterior distribution is no longer analytically tractable: approximate inference is required.

Computer Emulation

- A statistical model used as a surrogate for a computationally expensive computer model.
- Higdon et al. (2008) use the linear model of coregionalization to model images representing the evolution of the implosion of steel cylinders.
- In Conti and O'Hagan (2009) use the ICM to model a vegetation model: called the Sheffield Dynamic Global Vegetation Model (Woodward et al., 1998).

Latent Force Models

Neil D. Lawrence

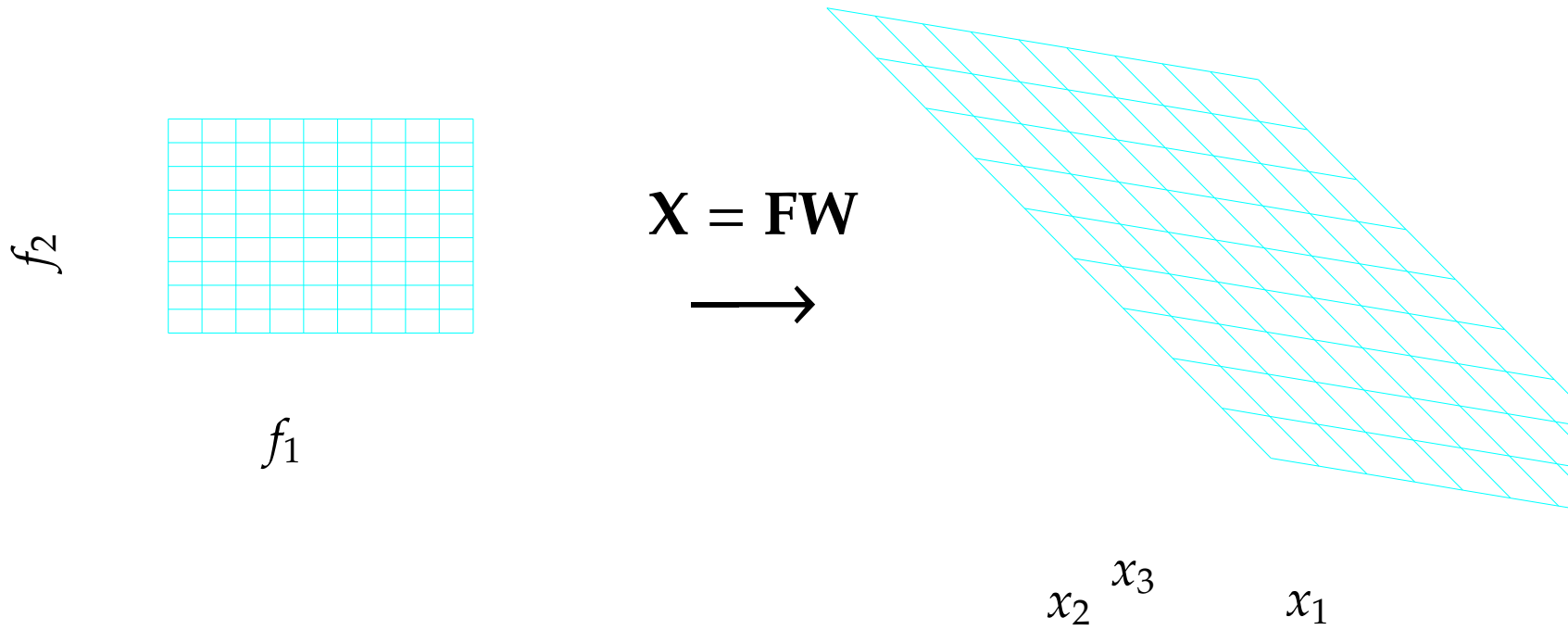
GPRS
19th–22nd January 2015

Outline

Outline

Linear Dimensionality Reduction

- Find a lower dimensional plane embedded in a higher dimensional space.
- The plane is described by the matrix $\mathbf{W} \in \Re^{p \times q}$.



Dimensionality Reduction

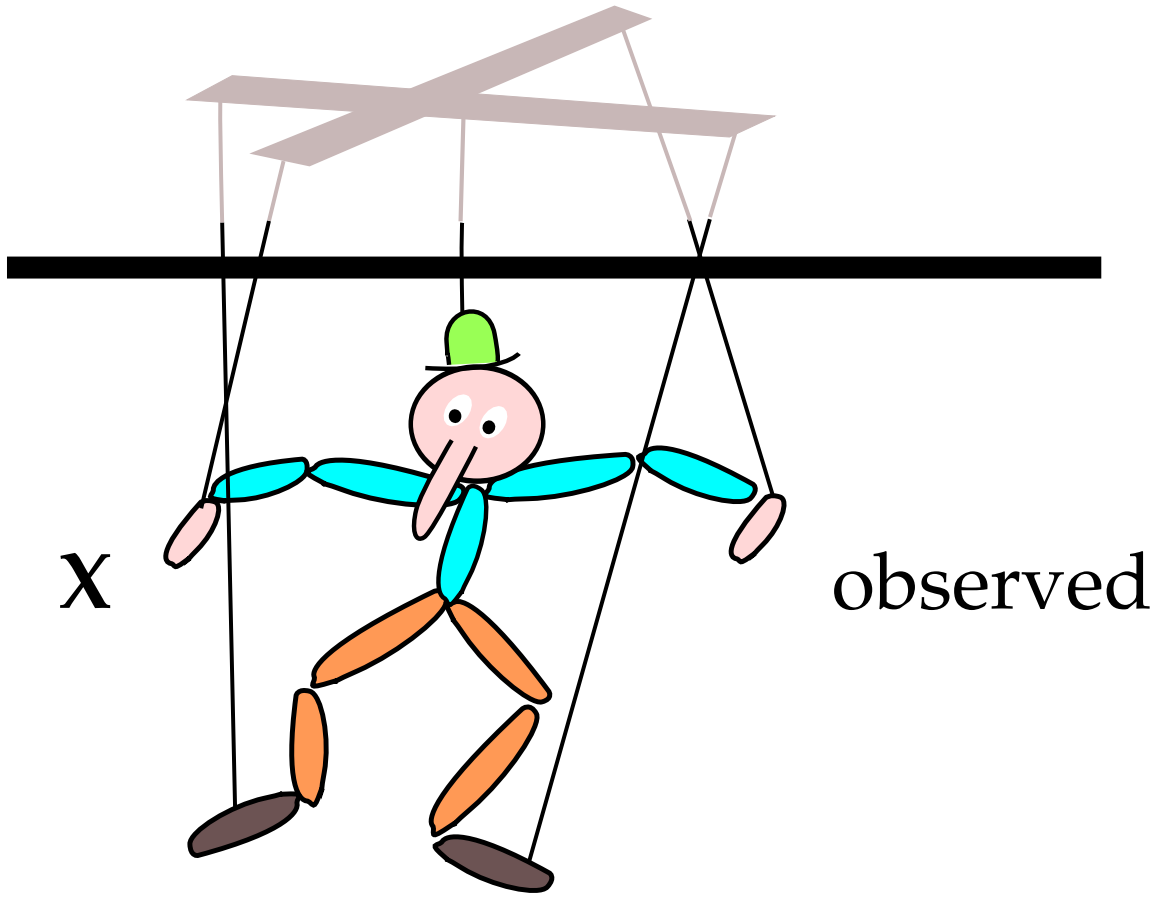
- Linear relationship between the data, $\mathbf{X}$, and a reduced dimensional representation, $\mathbf{F}$.

$$\mathbf{X} = \mathbf{F}\mathbf{W} + \boldsymbol{\epsilon},$$

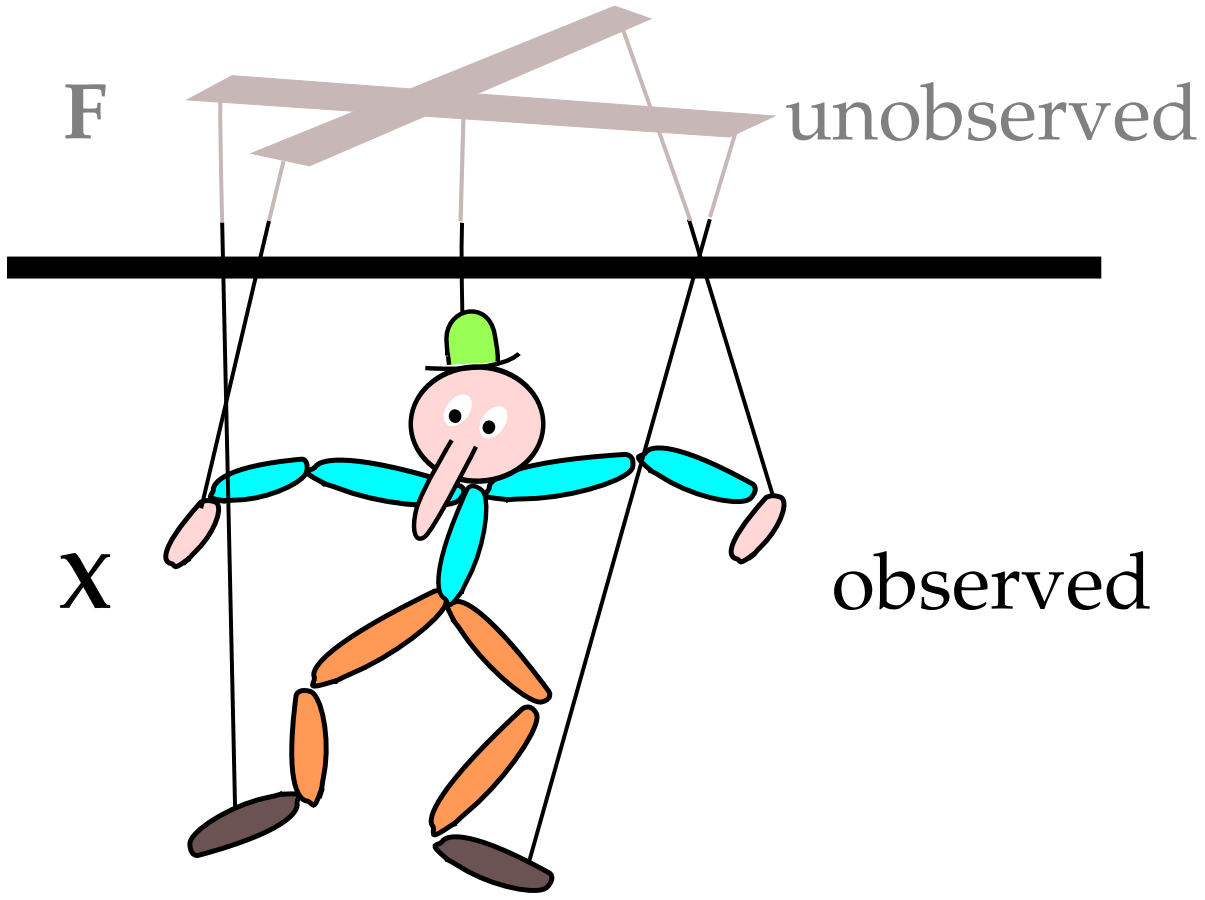
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

- Problem is we don't know what $\mathbf{F}$ should be!

Marionette Analogy



Marionette Analogy



$\mathbf{F}$ is a Latent Variable

- Define a *probability distribution* for $\mathbf{F}$.
- Marginalize out $\mathbf{F}$ (integrate over).
- Optimize with respect to $\mathbf{W}$.
- For Gaussian distribution, $\mathbf{F} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - and $\Sigma = \sigma^2\mathbf{I}$ we have probabilistic PCA (Tipping and Bishop, 1999; Roweis, 1998).
  - and Σ constrained to be diagonal, we have factor analysis.

Dimensionality Reduction: Temporal Data

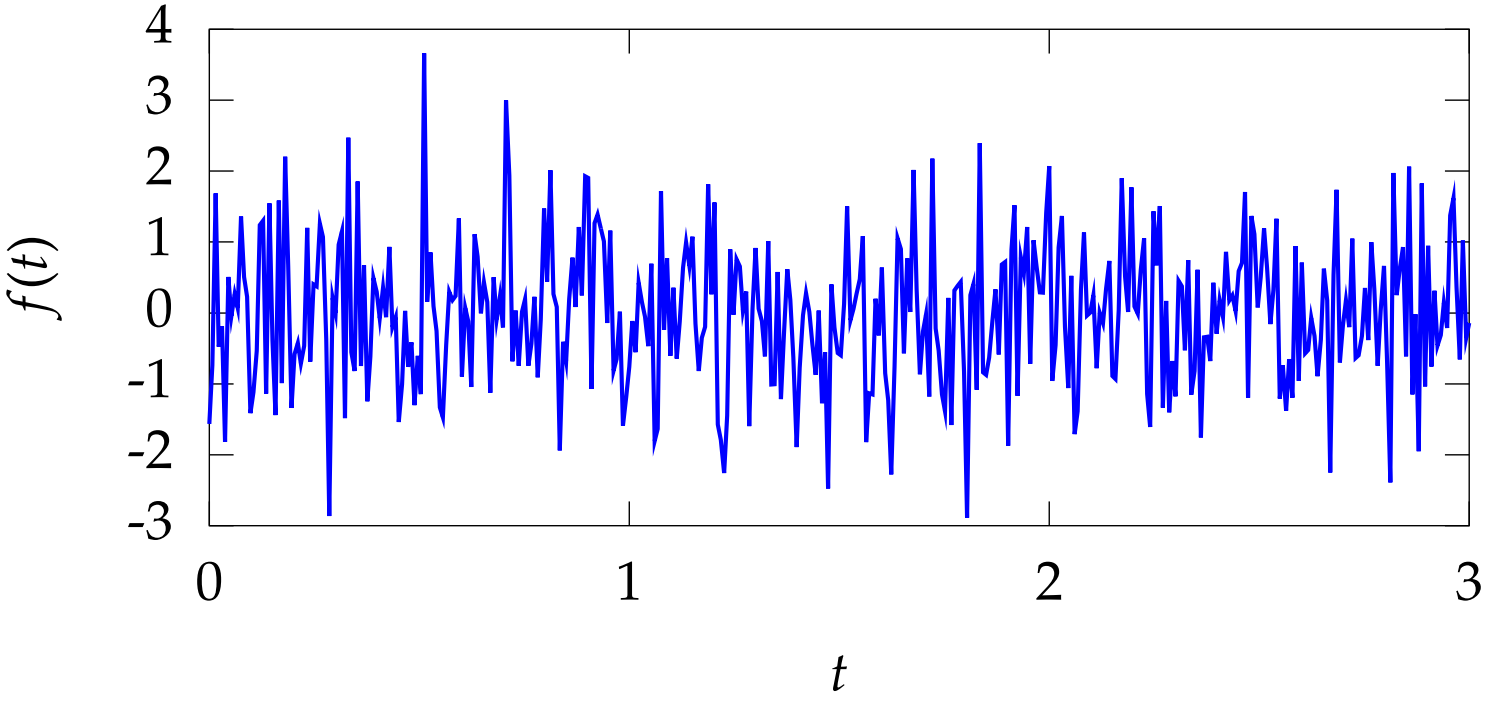


Figure : PCA: Pure sampling from a Gaussian does not retain temporal effects.

Dimensionality Reduction: Temporal Data

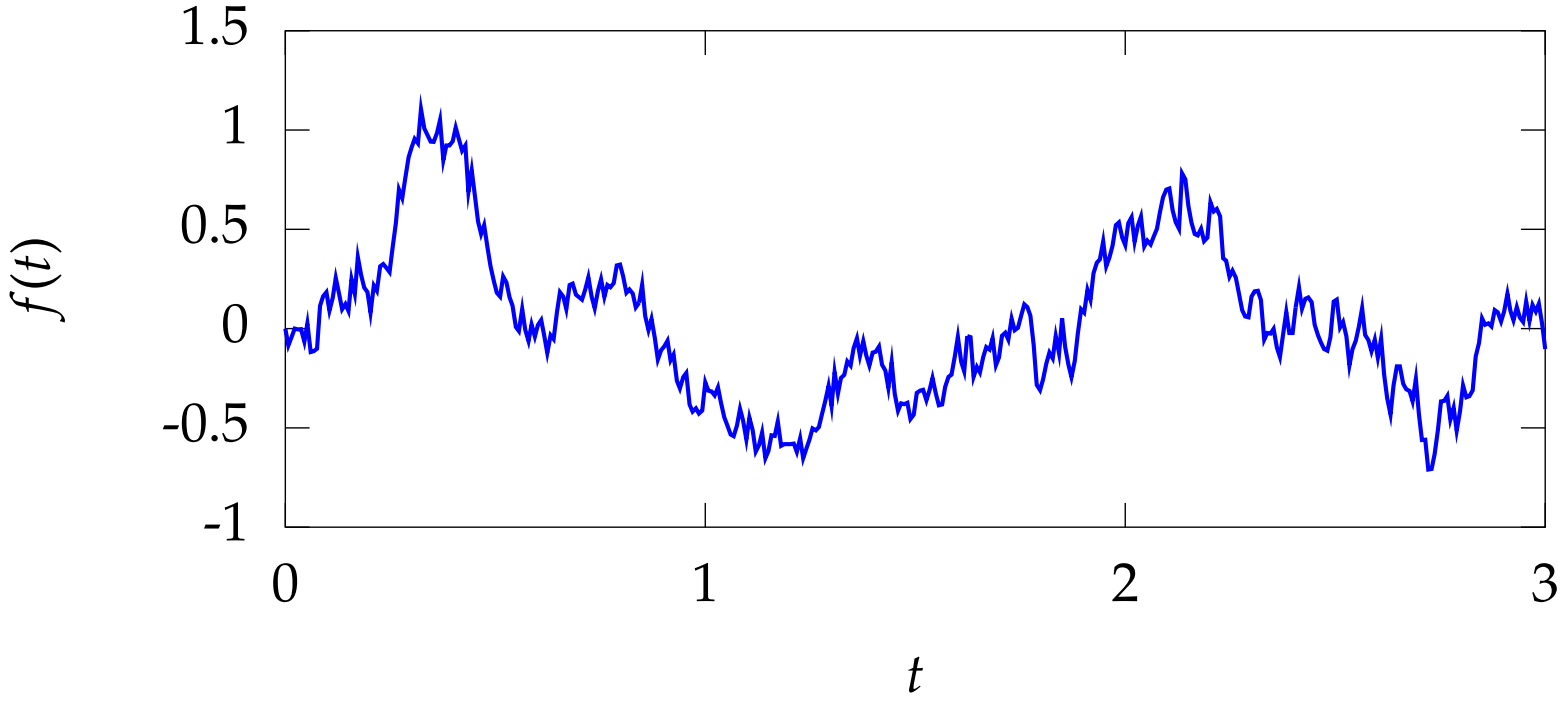


Figure : Kalman filter (Rauch-Tung-Striebel smoother) is Markov-Gaussian (non smooth).

Dimensionality Reduction: Temporal Data

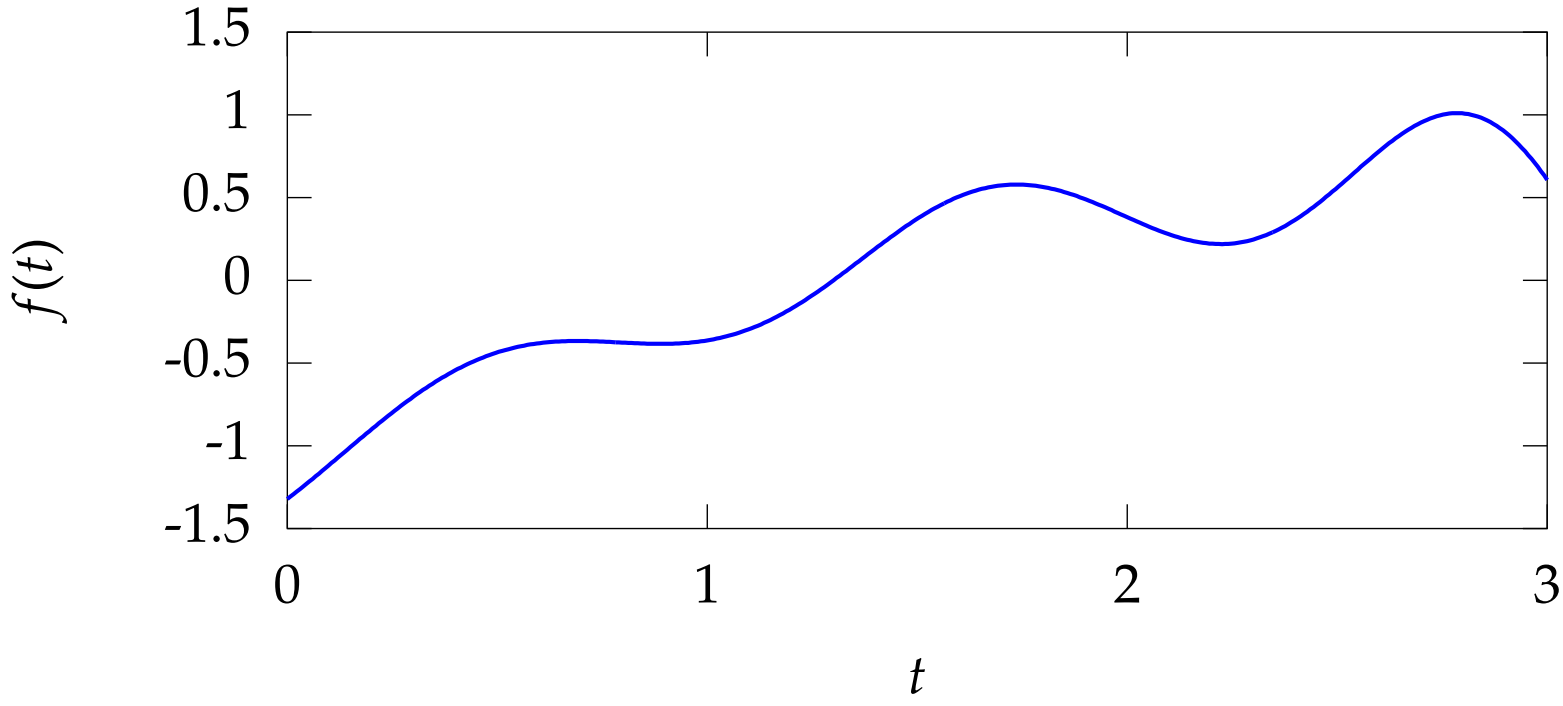


Figure : General Gaussian processes allow for priors over *smooth* functions.

Mechanical Analogy

Back to Mechanistic Models!

- These models rely on the latent variables to provide the dynamic information.
- We now introduce a further dynamical system with a *mechanistic* inspiration.
- Physical Interpretation:
  - the latent functions, $f_i(t)$ are q forces.
  - We observe the displacement of p springs to the forces.,
  - Interpret system as the force balance equation, $\mathbf{XD} = \mathbf{FS} + \epsilon$.
  - Forces act, e.g. through levers — a matrix of sensitivities, $\mathbf{S} \in \Re^{q \times p}$.
  - Diagonal matrix of spring constants, $\mathbf{D} \in \Re^{p \times p}$.
  - Original System: $\mathbf{W} = \mathbf{S}\mathbf{D}^{-1}$.

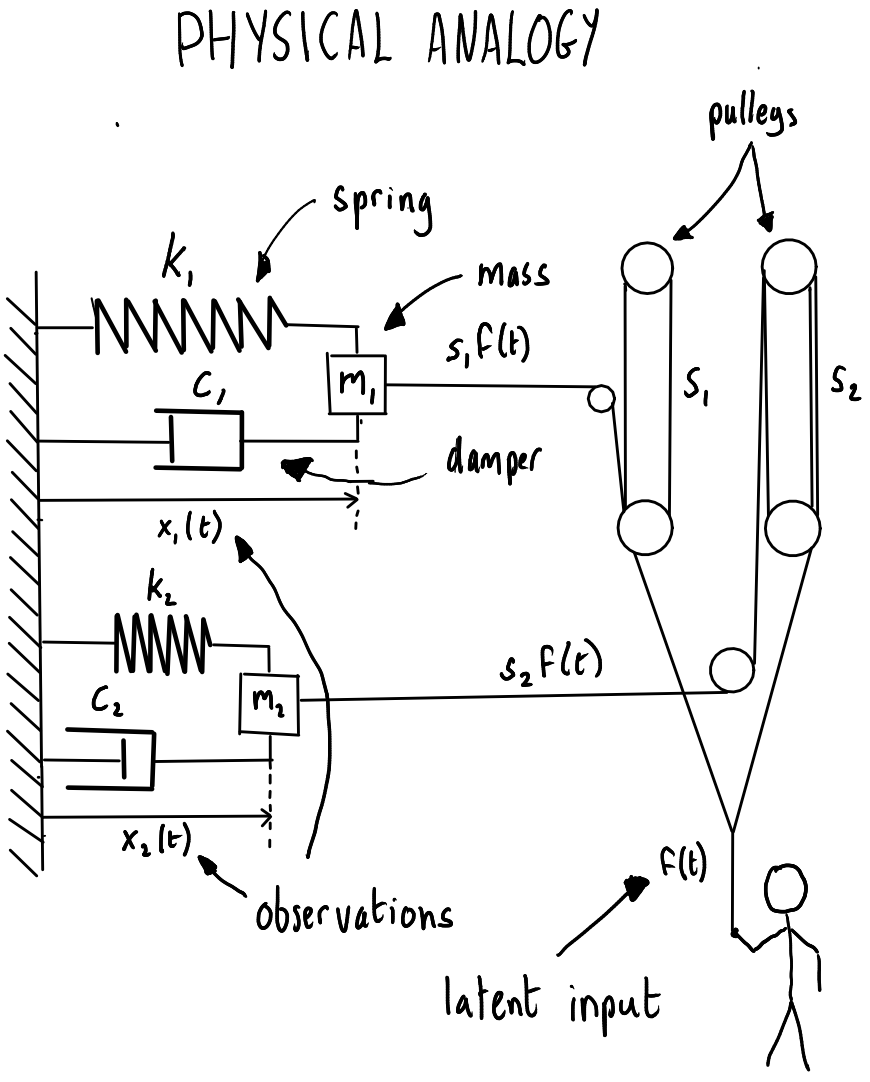
Extend Model

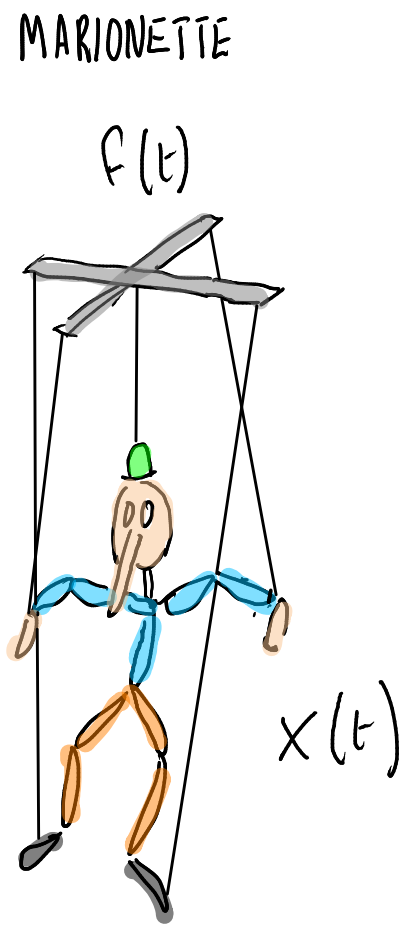
- Add a damper and give the system mass.

$$\mathbf{FS} = \ddot{\mathbf{X}}\mathbf{M} + \dot{\mathbf{X}}\mathbf{C} + \mathbf{X}\mathbf{D} + \boldsymbol{\epsilon}.$$

- Now have a second order mechanical system.
- It will exhibit inertia and resonance.
- There are many systems that can also be represented by differential equations.
  - When being forced by latent function(s), $\{f_i(t)\}_{i=1}^{q}$, we call this a *latent force model*.

Physical Analogy





Gaussian Process priors and Latent Force Models

Driven Harmonic Oscillator

- For Gaussian process we can compute the covariance matrices for the output displacements.
- For one displacement the model is

$$m_k \ddot{x}_k(t) + c_k \dot{x}_k(t) + d_k x_k(t) = b_k + \sum_{i=0}^{q} s_{ik} f_i(t), \tag{3}$$

where, m_k is the kth diagonal element from $\mathbf{M}$ and similarly for c_k and d_k. s_{ik} is the i,kth element of $\mathbf{S}$.

- Model the latent forces as q independent, GPs with exponentiated quadratic covariances

$$k_{f_i f_l}(t, t') = \exp\left(-\frac{(t-t')^2}{2\ell_i^2}\right)\delta_{il}.$$

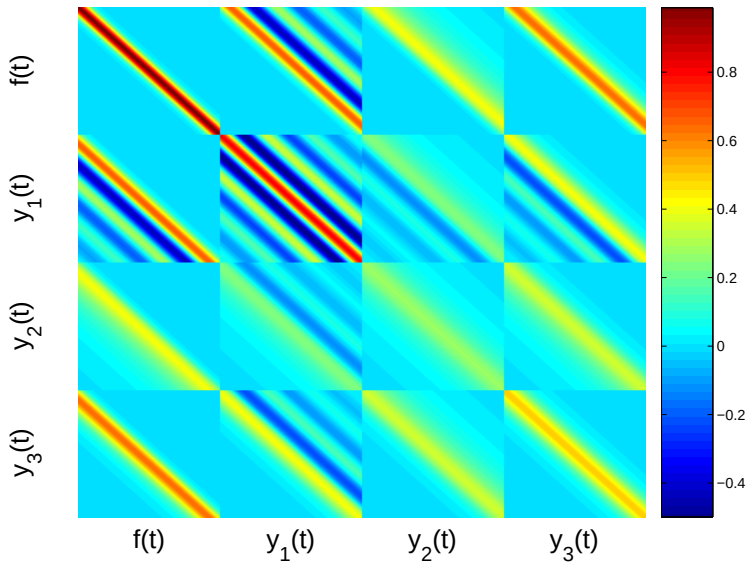
Covariance for ODE Model

- Exponentiated Quadratic Covariance function for $f(t)$

$$x_j(t) = \frac{1}{m_j \omega_j} \sum_{i=1}^{q} s_{ji} \exp(-\alpha_j t) \int_0^t f_i(\tau) \exp(\alpha_j \tau) \sin(\omega_j (t-\tau)) \mathrm{d}\tau$$

- Joint distribution for $x_1(t)$, $x_2(t)$, $x_3(t)$ and $f(t)$.
  Damping ratios:

ζ_1	ζ_2	ζ_3
0.125	2	1



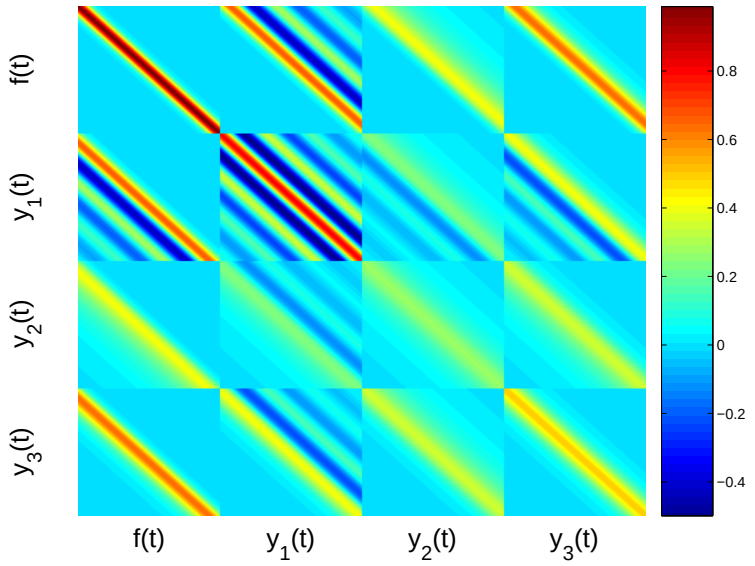
Covariance for ODE Model

- Analogy

$$x = \sum_i \mathbf{e}_i^\top \mathbf{f}_i \quad \mathbf{f}_i \sim \mathcal{N}(\mathbf{0}, \Sigma_i) \rightarrow x \sim \mathcal{N}\left(0, \sum_i \mathbf{e}_i^\top \Sigma_i \mathbf{e}_i\right)$$

- Joint distribution for $x_1(t)$, $x_2(t)$, $x_3(t)$ and $f(t)$.
  Damping ratios:

ζ_1	ζ_2	ζ_3
0.125	2	1



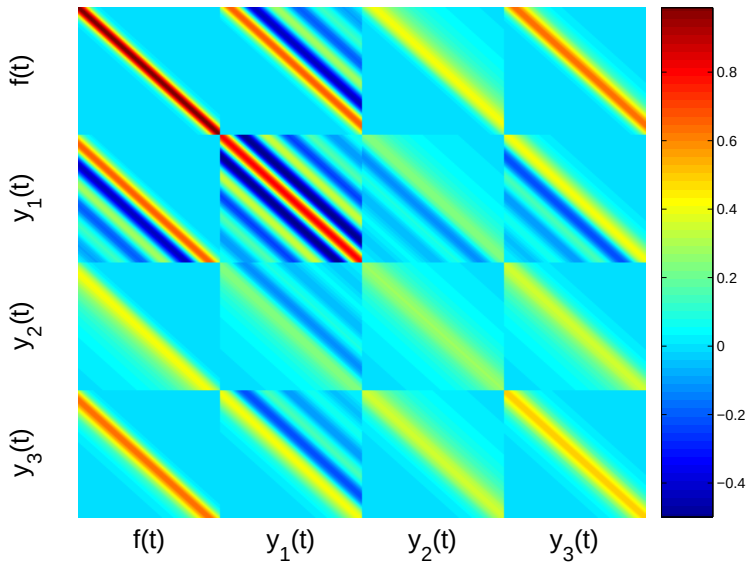
Covariance for ODE Model

- Exponentiated Quadratic Covariance function for $f(t)$

$$x_j(t) = \frac{1}{m_j \omega_j} \sum_{i=1}^{q} s_{ji} \exp(-\alpha_j t) \int_0^t f_i(\tau) \exp(\alpha_j \tau) \sin(\omega_j (t-\tau)) \mathrm{d}\tau$$

- Joint distribution for $x_1(t)$, $x_2(t)$, $x_3(t)$ and $f(t)$.
  Damping ratios:

ζ_1	ζ_2	ζ_3
0.125	2	1



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

- `lfmSample`

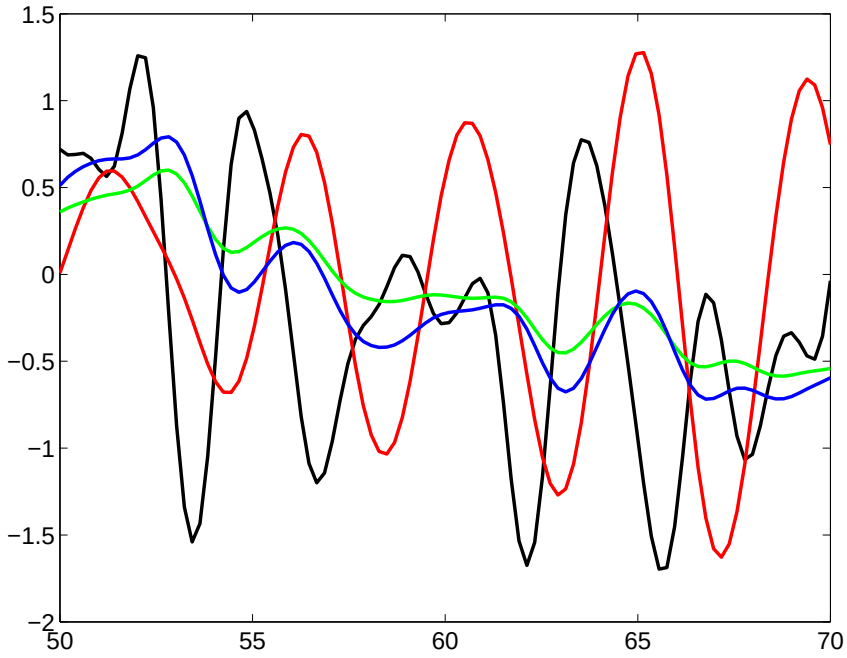


Figure : Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

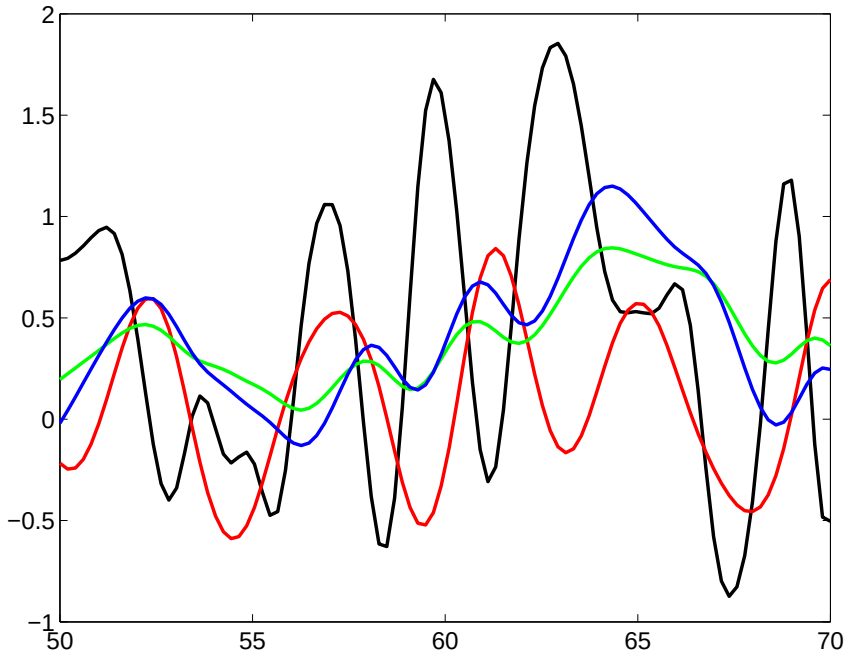


Figure : Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

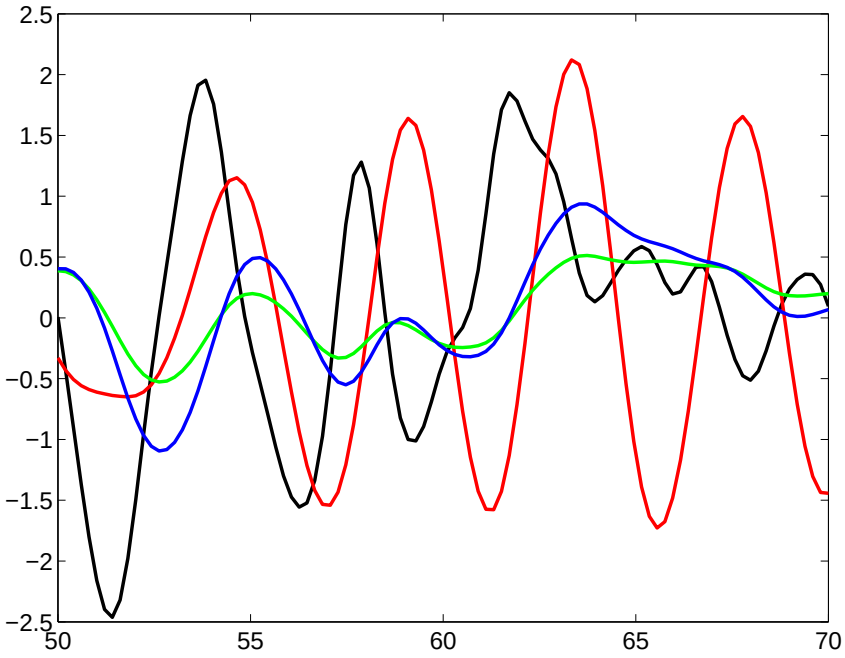


Figure : Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

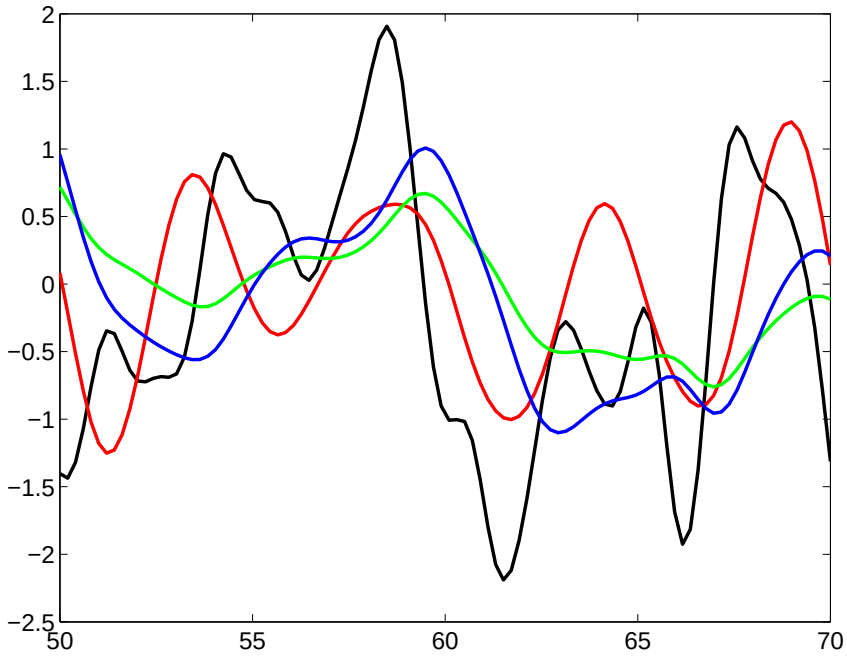


Figure : Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

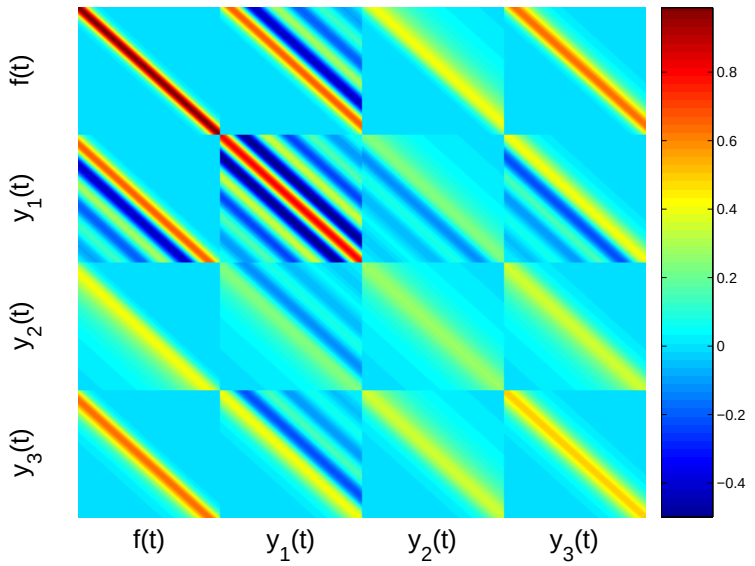
Covariance for ODE

- Exponentiated Quadratic Covariance function for $f(t)$

$$x_j(t) = \frac{1}{m_j \omega_j} \sum_{i=1}^{q} s_{ji} \exp(-\alpha_j t) \int_0^t f_i(\tau) \exp(\alpha_j \tau) \sin(\omega_j (t-\tau)) \mathrm{d}\tau$$

- Joint distribution for $x_1(t)$, $x_2(t)$, $x_3(t)$ and $f(t)$.
- Damping ratios:

ζ_1	ζ_2	ζ_3
0.125	2	1



Example: Motion Capture

Mauricio Alvarez and David Luengo (Álvarez et al., 2009, 2013)

- Motion capture data: used for animating human motion.
- Multivariate time series of angles representing joint positions.
- Objective: generalize from training data to realistic motions.
- Use 2nd Order Latent Force Model with mass/spring/damper (resistor inductor capacitor) at each joint.

Example: Motion Capture

Mauricio Alvarez and David Luengo (Álvarez et al., 2009, 2013)

- Motion capture data: used for animating human motion.
- Multivariate time series of angles representing joint positions.
- Objective: generalize from training data to realistic motions.
- Use 2nd Order Latent Force Model with mass/spring/damper (resistor inductor capacitor) at each joint.

Example: Motion Capture

Mauricio Alvarez and David Luengo (Álvarez et al., 2009, 2013)

- Motion capture data: used for animating human motion.
- Multivariate time series of angles representing joint positions.
- Objective: generalize from training data to realistic motions.
- Use 2nd Order Latent Force Model with mass/spring/damper (resistor inductor capacitor) at each joint.

Example: Motion Capture

Mauricio Alvarez and David Luengo (Álvarez et al., 2009, 2013)

- Motion capture data: used for animating human motion.
- Multivariate time series of angles representing joint positions.
- Objective: generalize from training data to realistic motions.
- Use 2nd Order Latent Force Model with mass/spring/damper (resistor inductor capacitor) at each joint.

Prediction of Test Motion

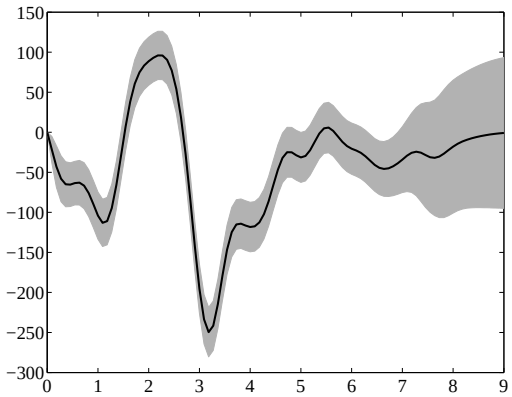
- Model left arm only.
- 3 balancing motions (18, 19, 20) from subject 49.
- 18 and 19 are similar, 20 contains more dramatic movements.
- Train on 18 and 19 and testing on 20
- Data was down-sampled by 32 (from 120 fps).
- Reconstruct motion of left arm for 20 given other movements.
- Compare with GP that predicts left arm angles given other body angles.

Mocap Results

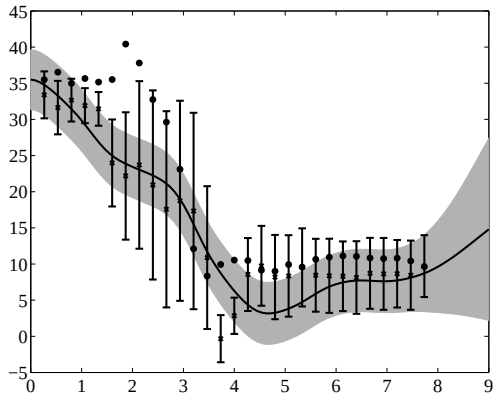
Table : Root mean squared (RMS) angle error for prediction of the left arm's configuration in the motion capture data. Prediction with the latent force model outperforms the prediction with regression for all apart from the radius's angle.

Angle	Latent Force Error	Regression Error
Radius	4.11	4.02
Wrist	6.55	6.65
Hand X rotation	1.82	3.21
Hand Z rotation	2.76	6.14
Thumb X rotation	1.77	3.10
Thumb Z rotation	2.73	6.09

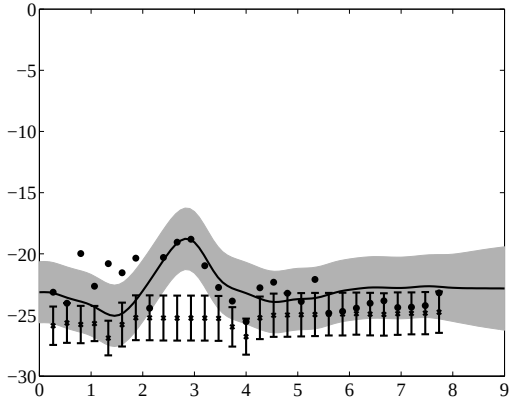
Mocap Results II



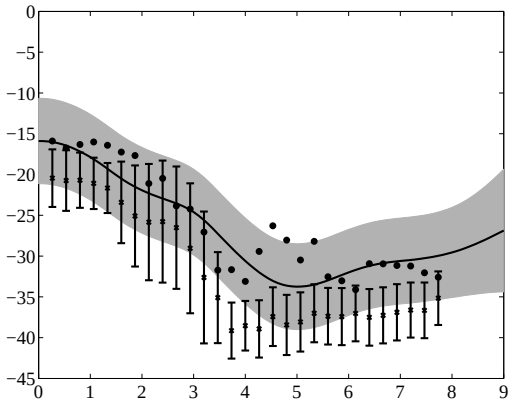
(a) Inferred Latent Force



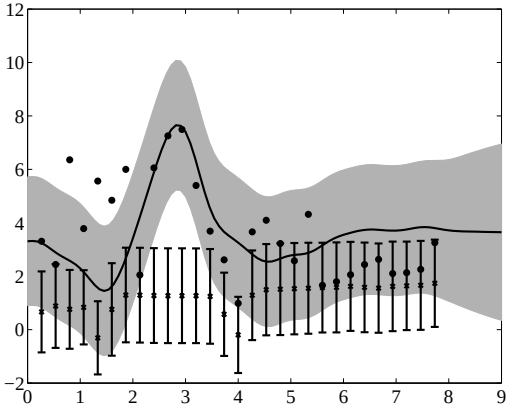
(b) Wrist



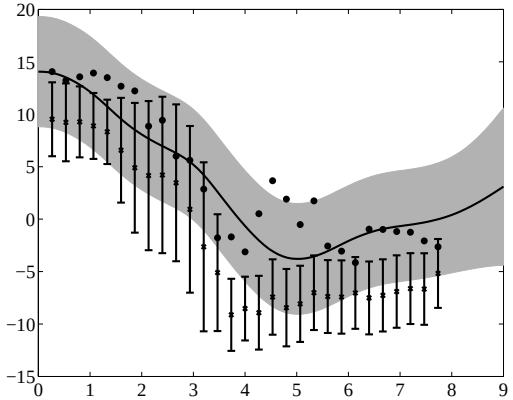
(c) Hand X Rotation



(d) Hand Z Rotation



(e) Thumb X Rotation



(f) Thumb Z Rotation

Figure : Predictions from LFM (solid line, grey error bars) and direct regression (crosses with stick error bars).

Motion Capture Experiments

- Data set is from the CMU motion capture data base[1].
- Two different types of movements: golf-swing and walking.
- Train on a subset of motions for each movement and test on a different subset.
- This assesses the model's ability to extrapolate.
- For testing: condition on three angles associated to the root nodes and first five and last five frames of the motion.
- Golf-swing use leave one out cross validation on four motions.
- For the walking train on 4 motions and validate on 8 motions.

Motion Capture Results

Table : RMSE and R^2 (explained variance) for golf swing and walking

Movement	**Method**	**RMSE**	**R^2 (%)**
Golf swing	IND GP	21.55 ± 2.35	30.99 ± 9.67
	MTGP	21.19 ± 2.18	45.59 ± 7.86
	SLFM	21.52 ± 1.93	49.32 ± 3.03
	LFM	**18.09 ± 1.30**	**72.25 ± 3.08**
Walking	IND GP	8.03 ± 2.55	30.55 ± 10.64
	MTGP	7.75 ± 2.05	37.77 ± 4.53
	SLFM	7.81 ± 2.00	36.84 ± 4.26
	LFM	**7.23 ± 2.18**	**48.15 ± 5.66**

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Example: Transcriptional Regulation

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

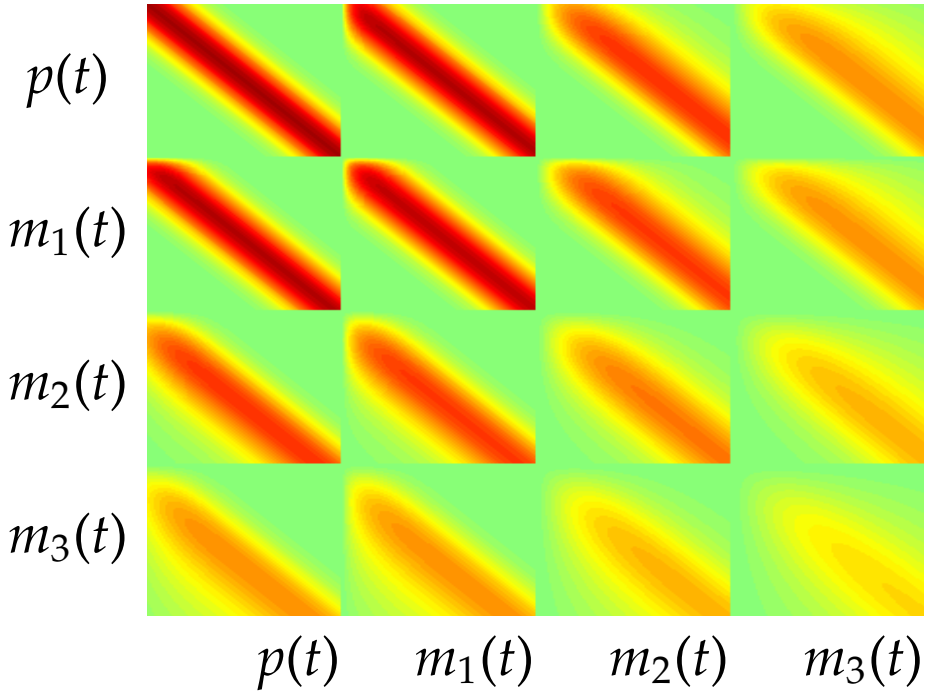
Covariance for Transcription Model

RBF covariance function for $p(t)$

$$m_i(t) = \frac{b_i}{d_i} + s_i \exp(-d_i t) \int_0^t p(u) \exp(d_i u)\, \mathrm{d}u.$$

- Joint distribution for $m_1(t)$, $m_2(t)$, $m_3(t)$, and $p(t)$.
- Here:

d_1	s_1	d_2	s_2	d_3	s_3
5	5	1	1	0.5	0.5



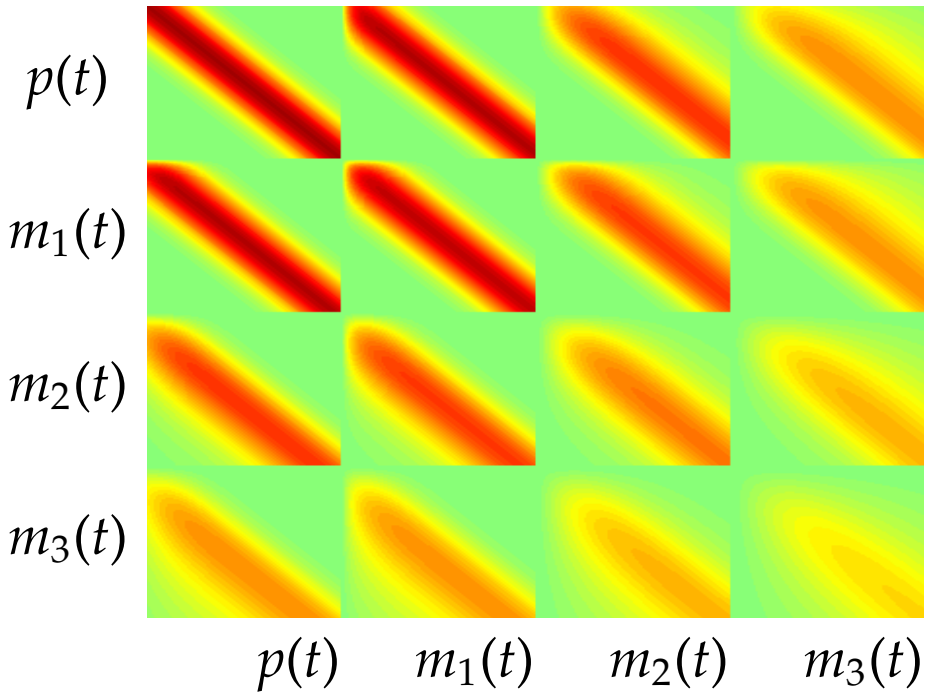
Covariance for Transcription Model

RBF covariance function for $p(t)$

$$m = b/d + \sum_i \mathbf{e}_i^\top \mathbf{p} \quad \mathbf{p} \sim \mathcal{N}(\mathbf{0}, \Sigma_i) \rightarrow m \sim \mathcal{N}\left(b/d, \sum_i \mathbf{e}_i^\top \Sigma_i \mathbf{e}_i\right)$$

- Joint distribution for $m_1(t)$, $m_2(t)$, $m_3(t)$, and $p(t)$.
- Here:

d_1	s_1	d_2	s_2	d_3	s_3
5	5	1	1	0.5	0.5



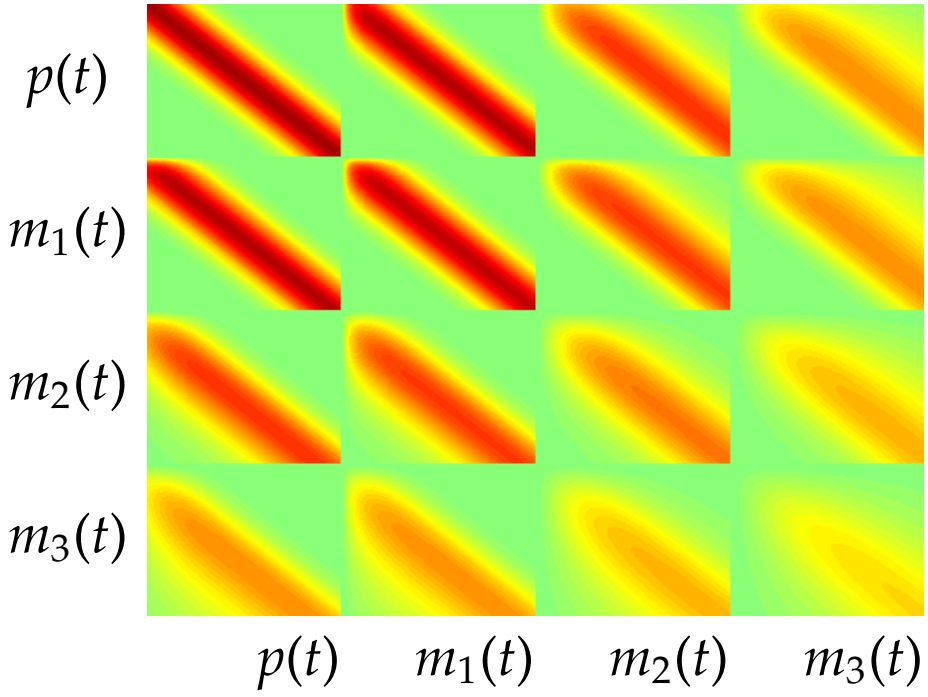
Covariance for Transcription Model

RBF covariance function for $p(t)$

$$m_i(t) = \frac{b_i}{d_i} + s_i \exp(-d_i t) \int_0^t p(u) \exp(d_i u)\, \mathrm{d}u.$$

- Joint distribution for $m_1(t)$, $m_2(t)$, $m_3(t)$, and $p(t)$.
- Here:

d_1	s_1	d_2	s_2	d_3	s_3
5	5	1	1	0.5	0.5



Joint Sampling of $p(t)$ and $m(t)$

- `simSample`

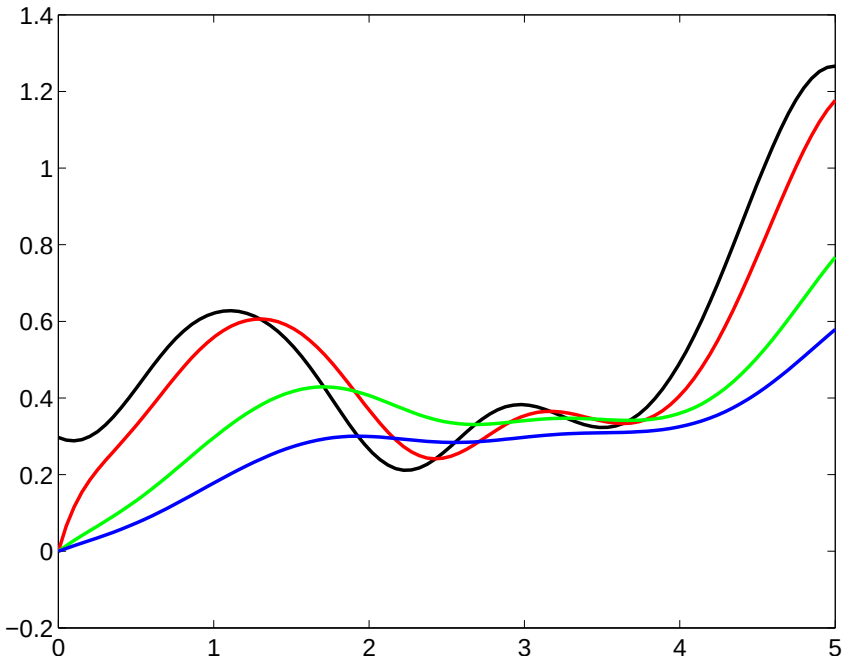


Figure : Joint samples from the ODE covariance, *black*: $p(t)$, *red*: $m_1(t)$ (high decay/sensitivity), *green*: $m_2(t)$ (medium decay/sensitivity) and *blue*: $m_3(t)$ (low decay/sensitivity).

Joint Sampling of $p(t)$ and $m(t)$

- `simSample`

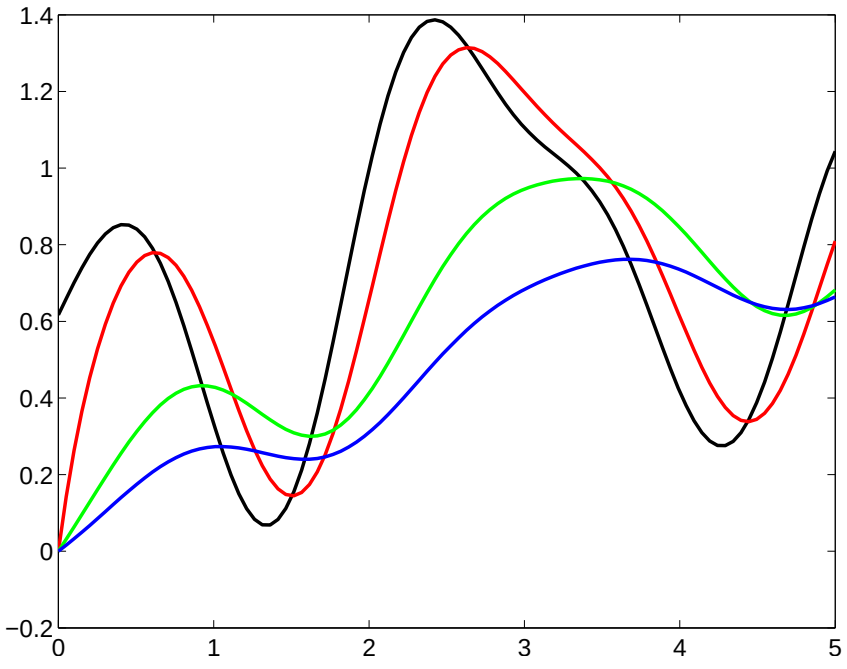


Figure : Joint samples from the ODE covariance, *black*: $p(t)$, *red*: $m_1(t)$ (high decay/sensitivity), *green*: $m_2(t)$ (medium decay/sensitivity) and *blue*: $m_3(t)$ (low decay/sensitivity).

Joint Sampling of $p(t)$ and $m(t)$

- `simSample`

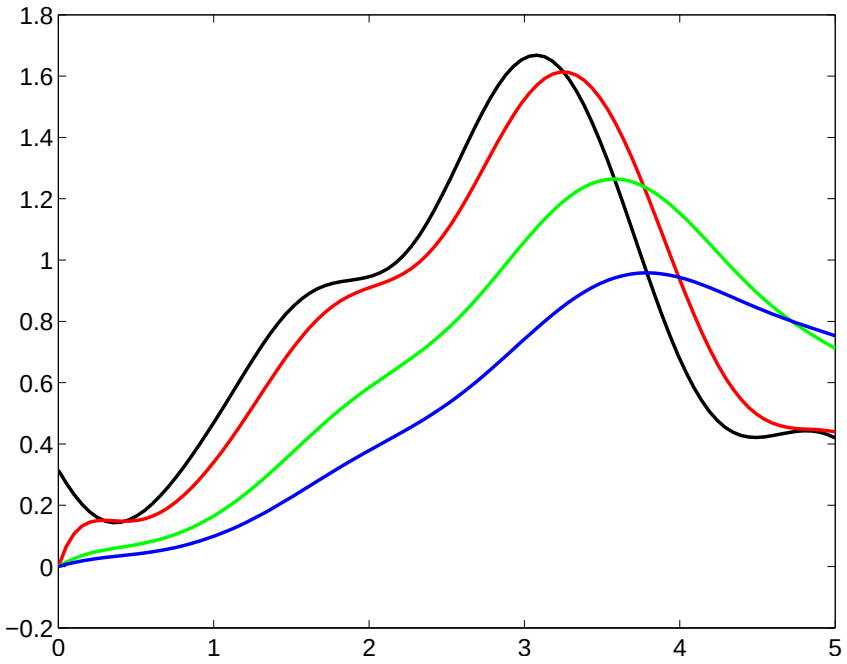


Figure : Joint samples from the ODE covariance, *black*: $p(t)$, *red*: $m_1(t)$ (high decay/sensitivity), *green*: $m_2(t)$ (medium decay/sensitivity) and *blue*: $m_3(t)$ (low decay/sensitivity).

Joint Sampling of $p(t)$ and $m(t)$

- `simSample`

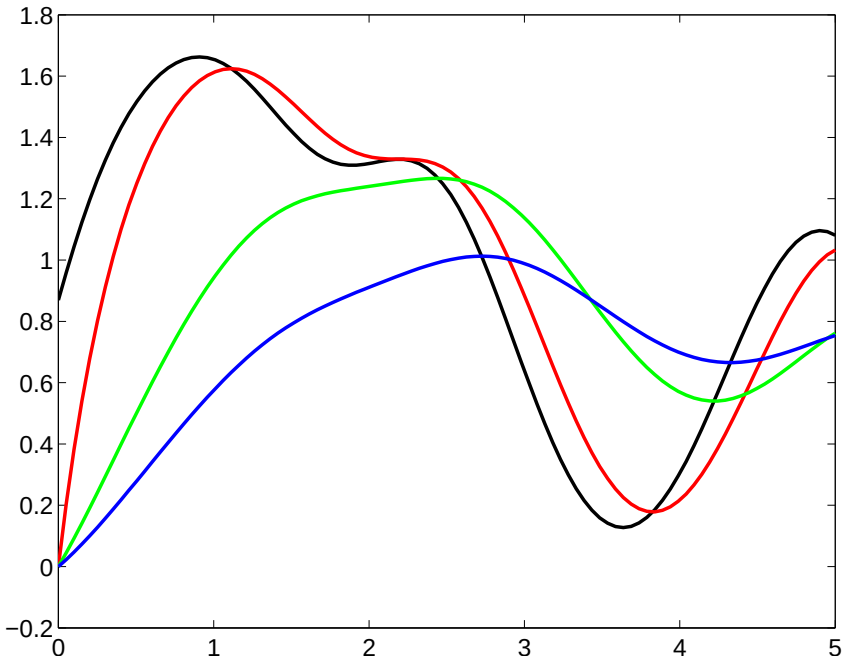
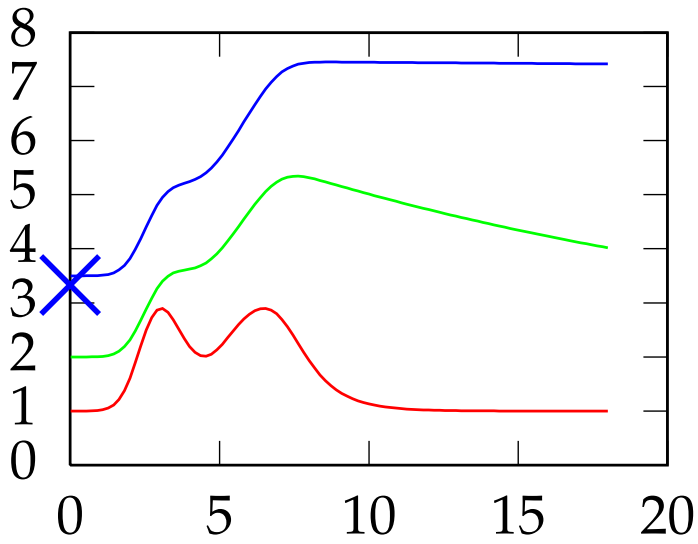
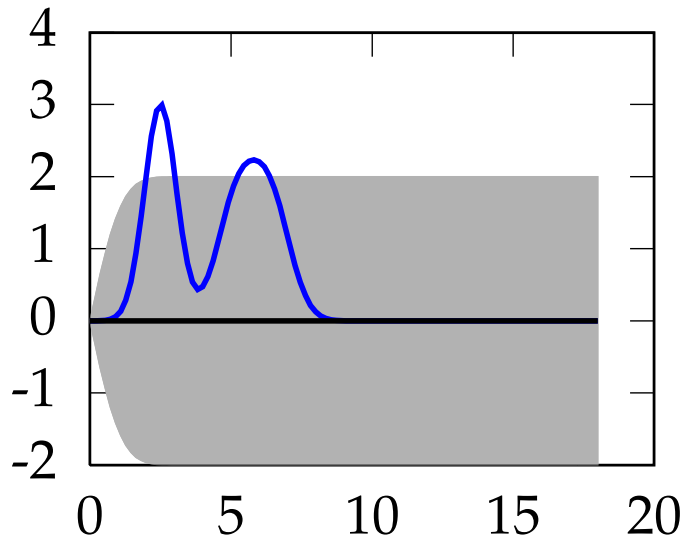
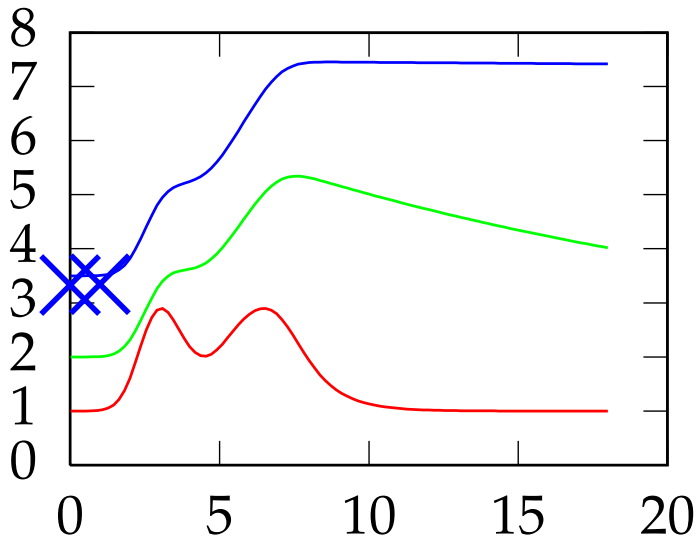
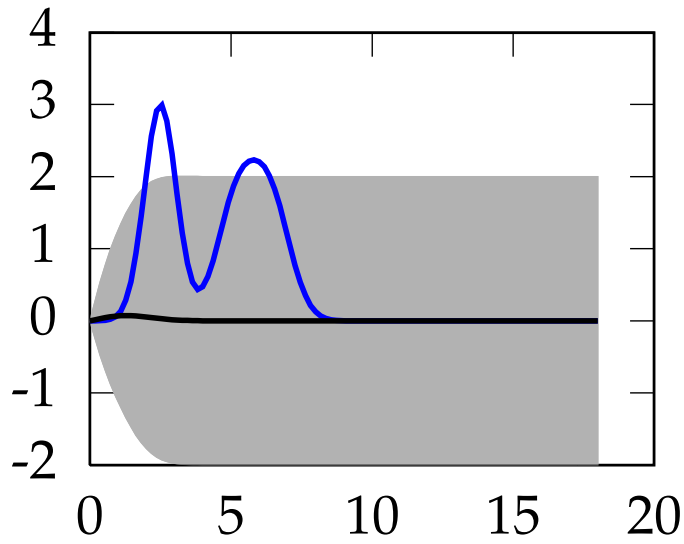


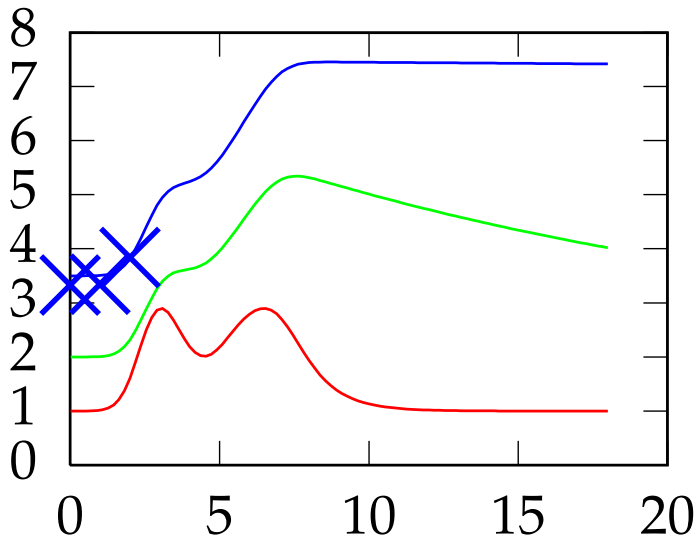
Figure : Joint samples from the ODE covariance, *black*: $p(t)$, *red*: $m_1(t)$ (high decay/sensitivity), *green*: $m_2(t)$ (medium decay/sensitivity) and *blue*: $m_3(t)$ (low decay/sensitivity).

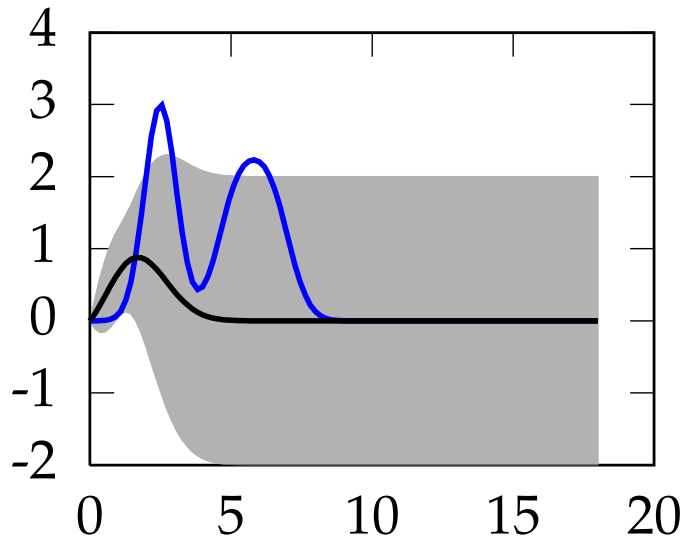

8
7
6
5
4
3
2
1
0
0
5
10
15
20

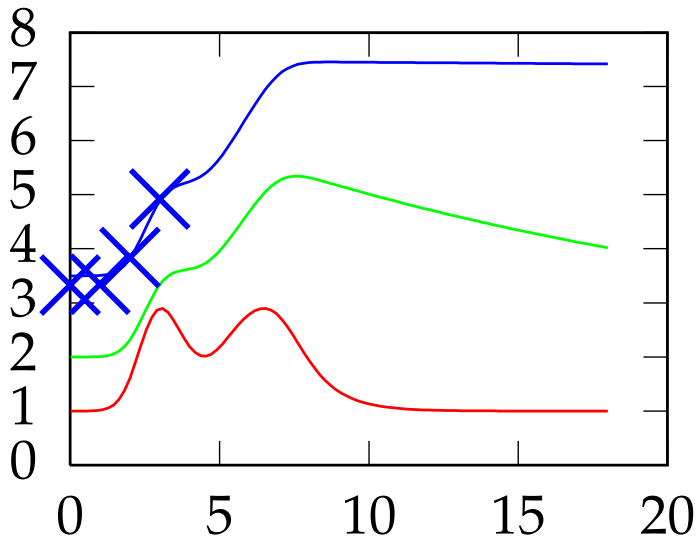

4
3
2
1
0
-1
-2
0
5
10
15
20

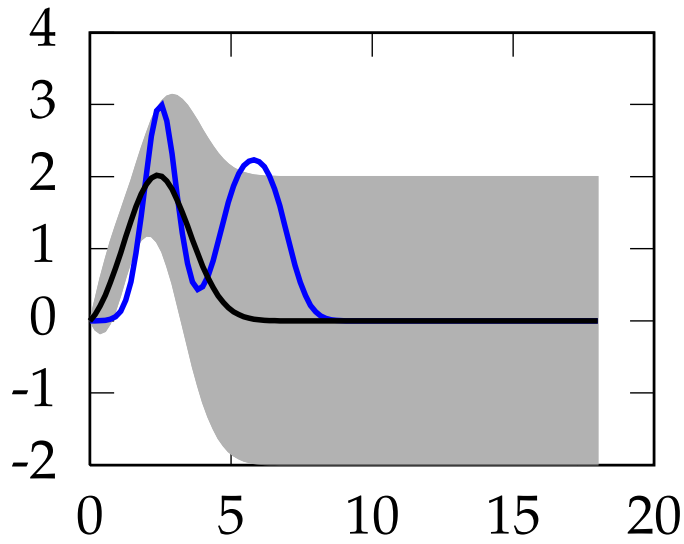

8
7
6
5
4
3
2
1
0
0
5
10
15
20

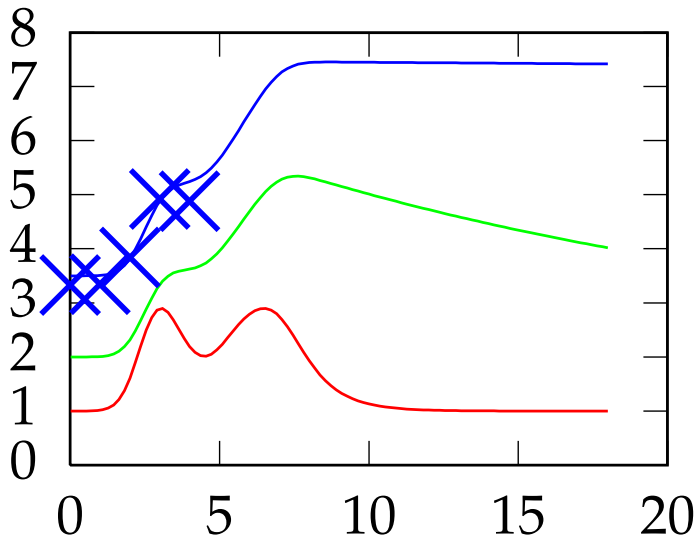

4
3
2
1
0
-1
-2
0
5
10
15
20

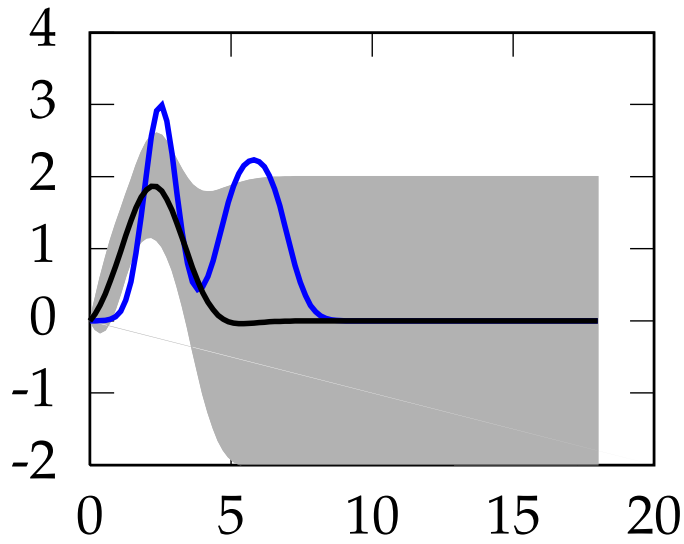

8
7
6
5
4
3
2
1
0
0
5
10
15
20

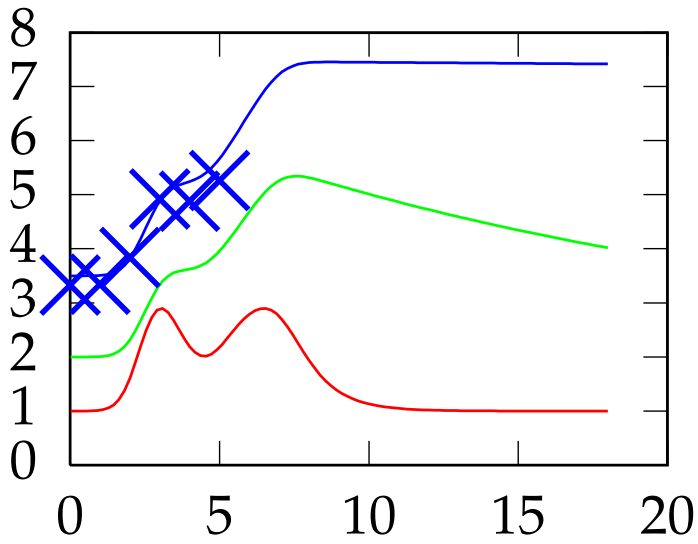

4
3
2
1
0
-1
-2
0
5
10
15
20

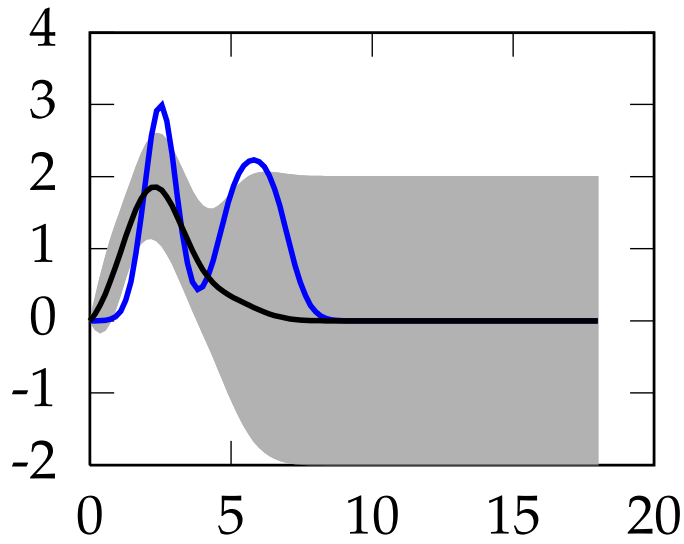

8
7
6
5
4
3
2
1
0
0
5
10
15
20

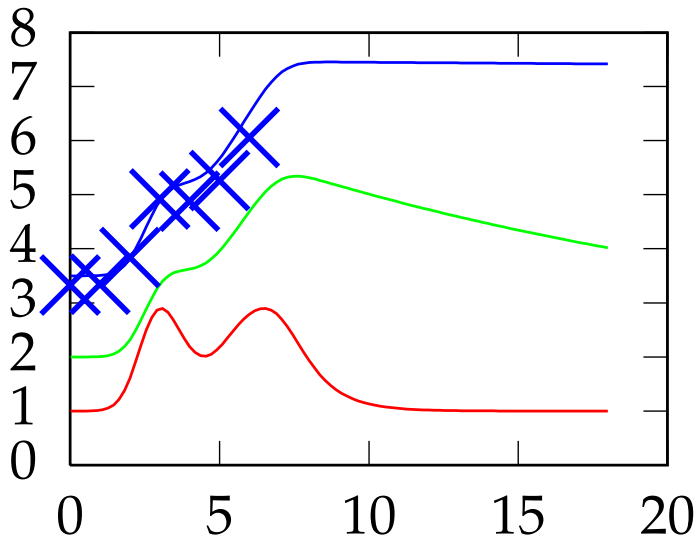

4
3
2
1
0
-1
-2
0
5
10
15
20

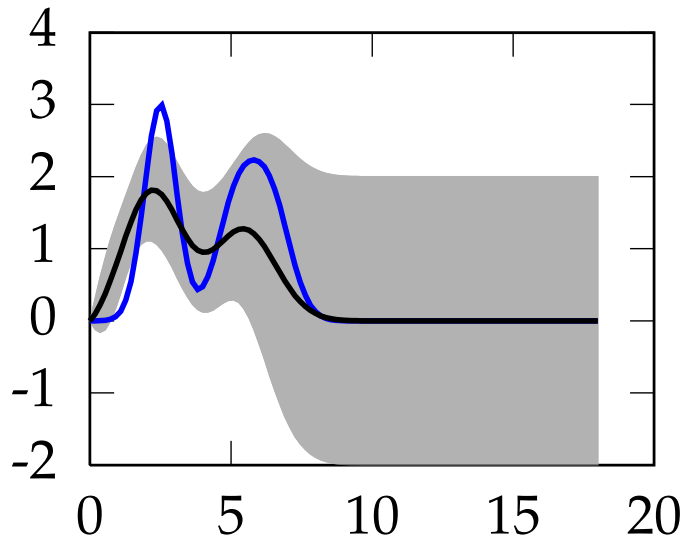

8
7
6
5
4
3
2
1
0
0
5
10
15
20

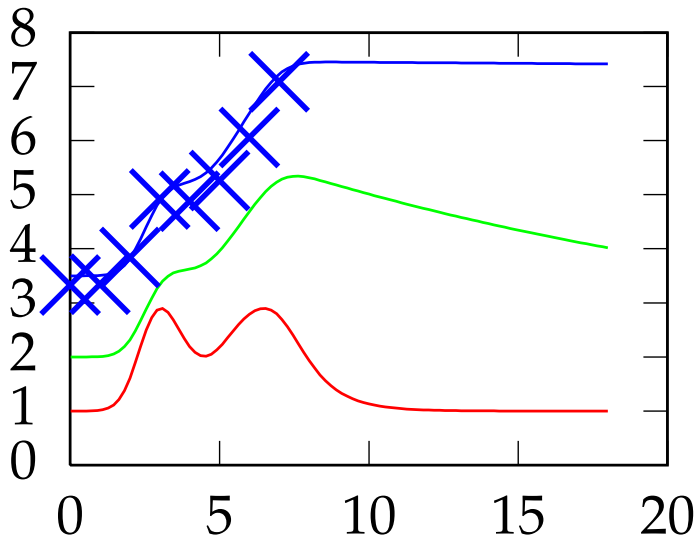

4
3
2
1
0
-1
-2
0
5
10
15
20

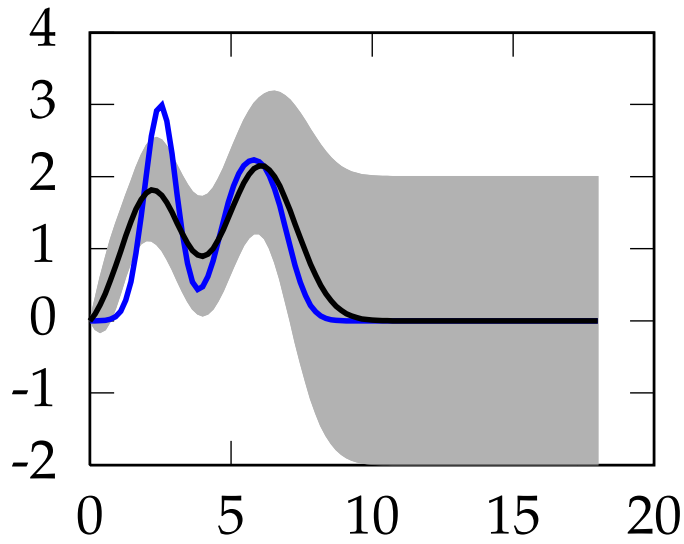

8
7
6
5
4
3
2
1
0
0
5
10
15
20

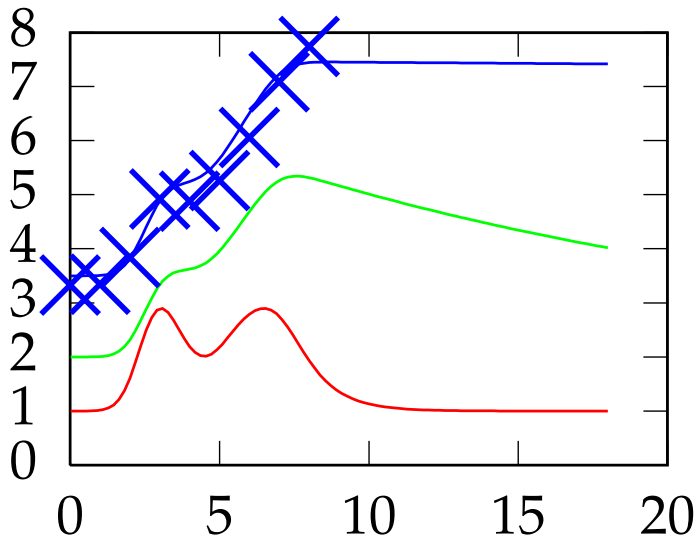

4
3
2
1
0
-1
-2
0
5
10
15
20

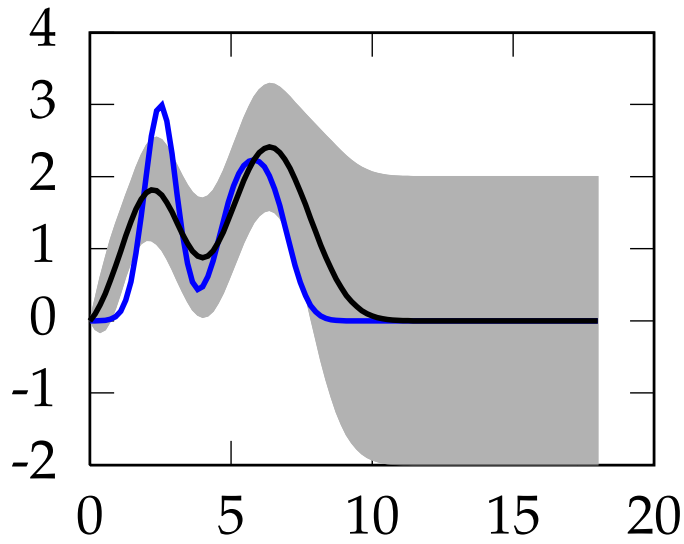

8
7
6
5
4
3
2
1
0
0
5
10
15
20

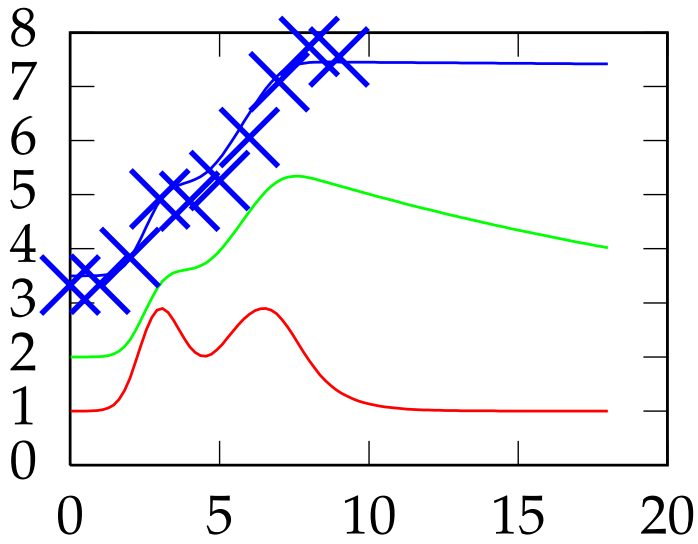

4
3
2
1
0
-1
-2
0
5
10
15
20

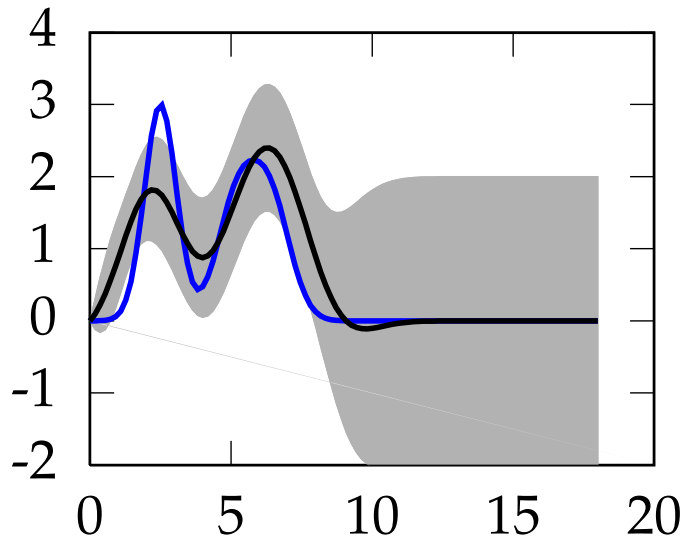

8
7
6
5
4
3
2
1
0
0
5
10
15
20

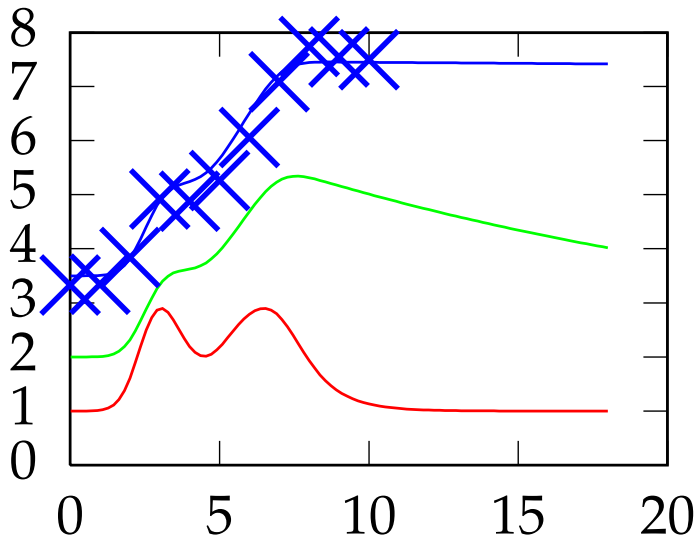

4
3
2
1
0
-1
-2
0
5
10
15
20

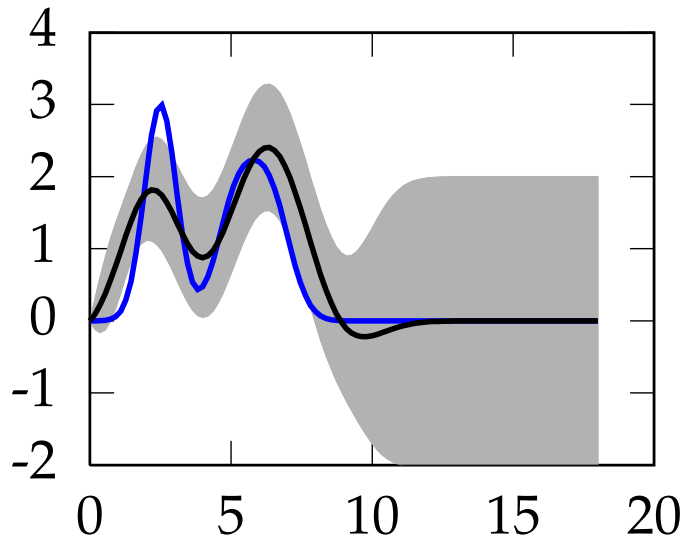

8
7
6
5
4
3
2
1
0
0
5
10
15
20

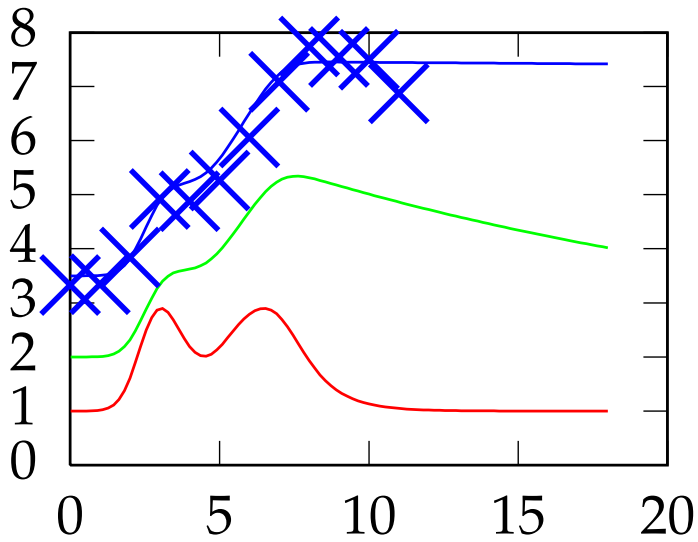

4
3
2
1
0
-1
-2
0
5
10
15
20

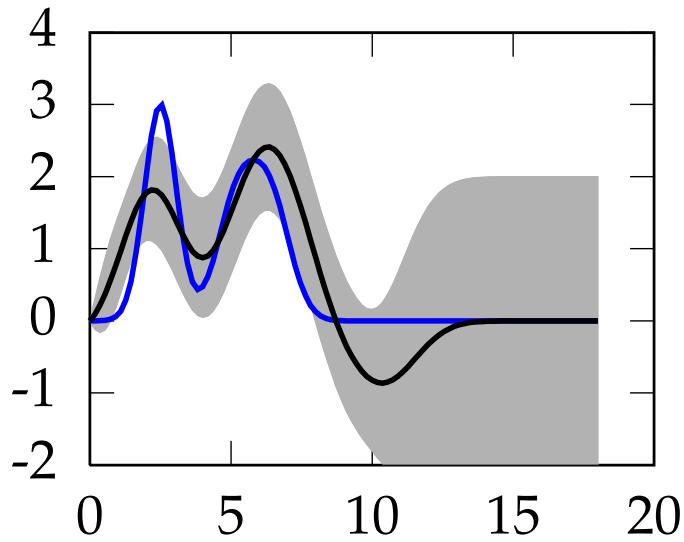

8
7
6
5
4
3
2
1
0
0
5
10
15
20

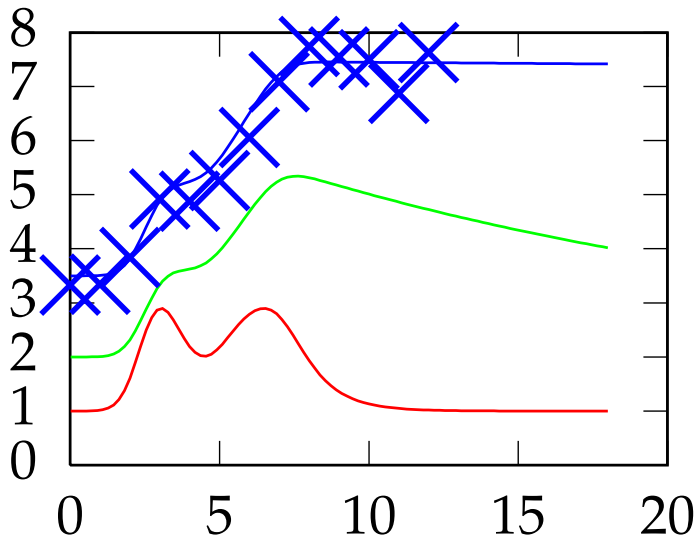

4
3
2
1
0
-1
-2
0
5
10
15
20

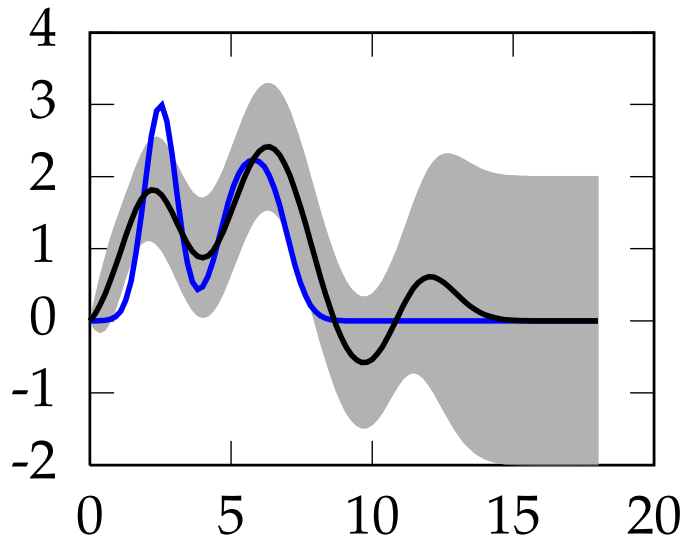

8
7
6
5
4
3
2
1
0
0
5
10
15
20

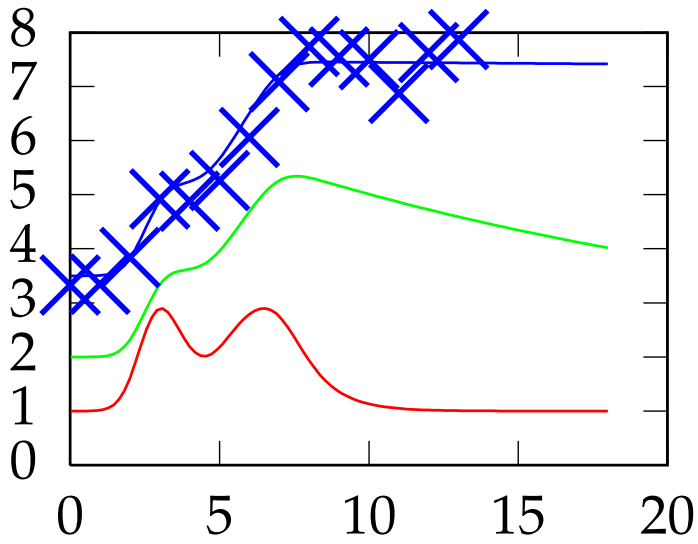

4
3
2
1
0
-1
-2
0
5
10
15
20

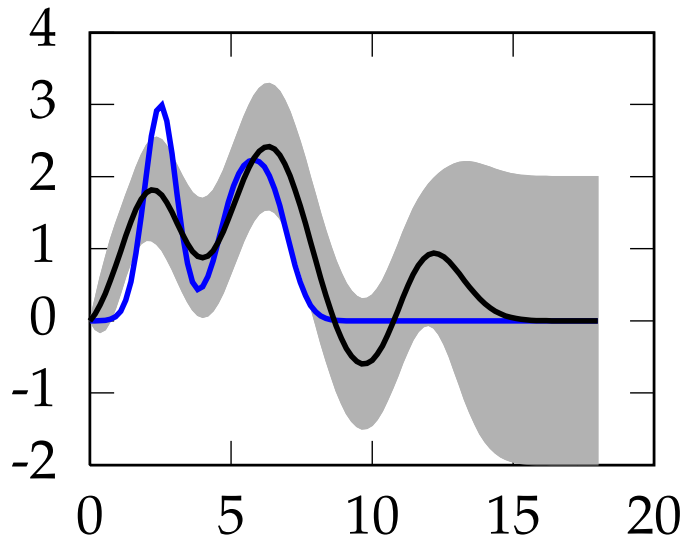

8
7
6
5
4
3
2
1
0
0
5
10
15
20

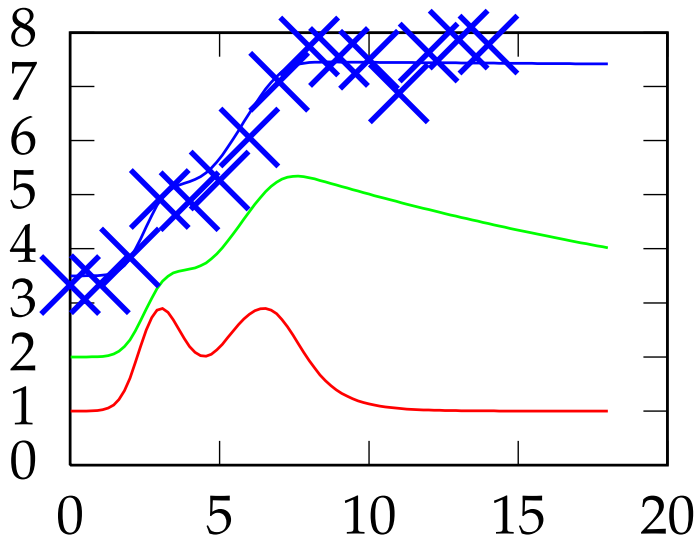

4
3
2
1
0
-1
-2
0
5
10
15
20

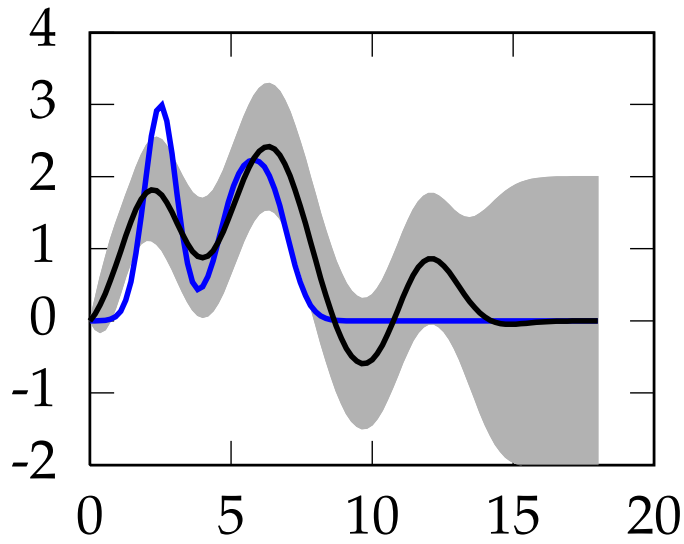

8
7
6
5
4
3
2
1
0
0
5
10
15
20

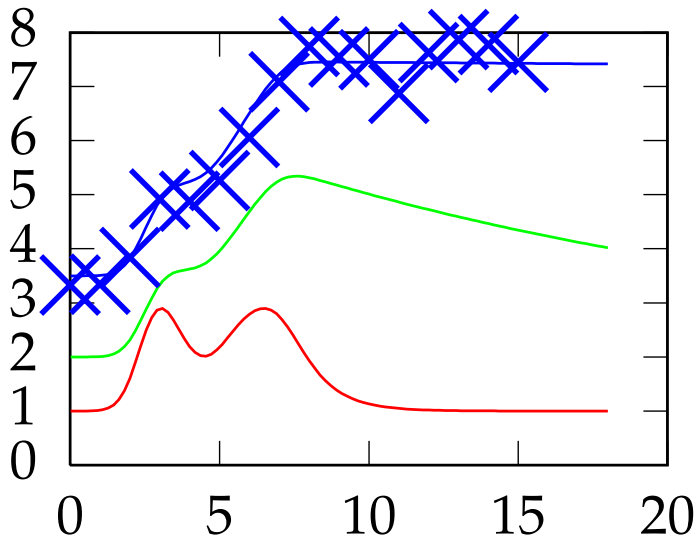

4
3
2
1
0
-1
-2
0
5
10
15
20

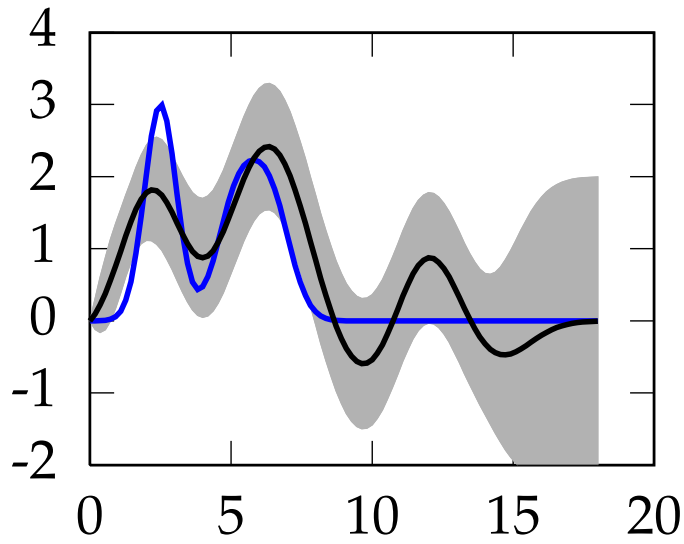

8
7
6
5
4
3
2
1
0
0
5
10
15
20

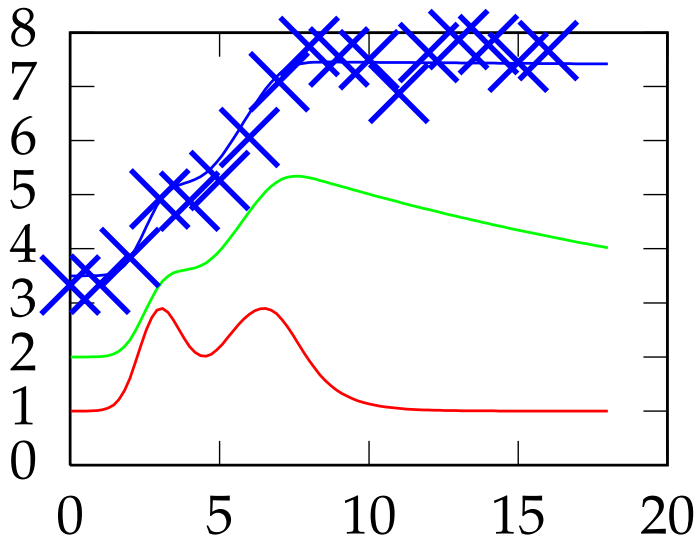

4
3
2
1
0
-1
-2
0
5
10
15
20

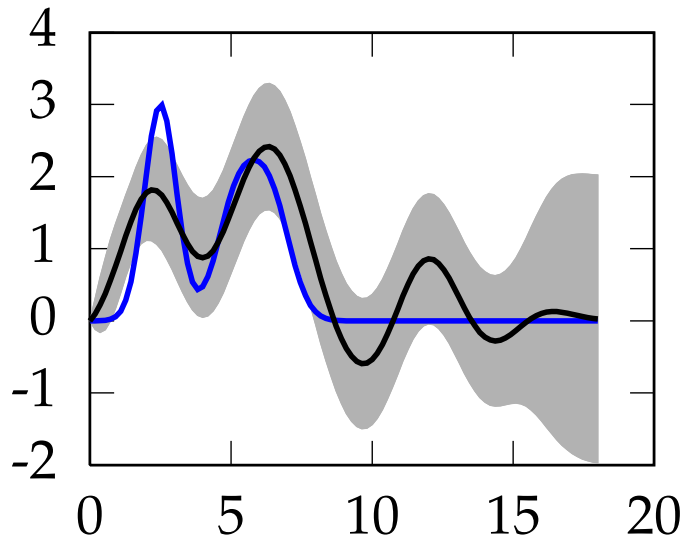

8
7
6
5
4
3
2
1
0
0
5
10
15
20

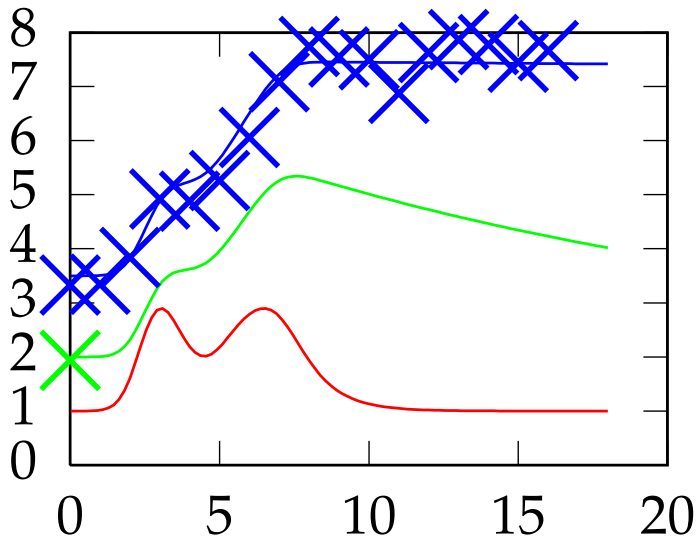

4
3
2
1
0
-1
-2
0
5
10
15
20

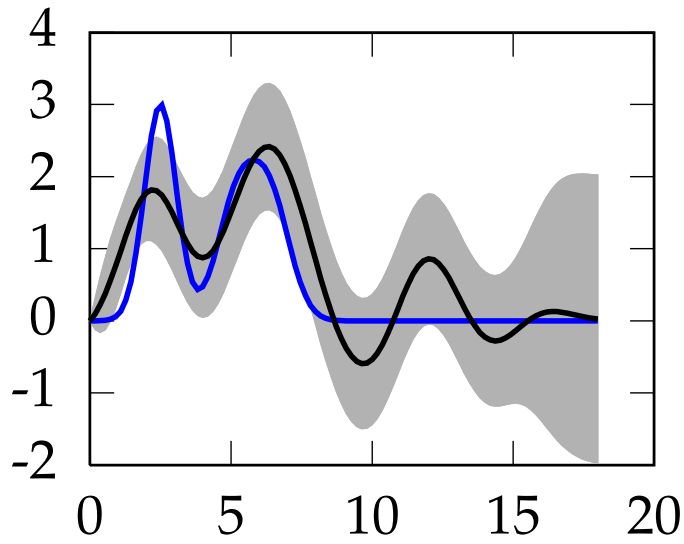

8
7
6
5
4
3
2
1
0
0
5
10
15
20

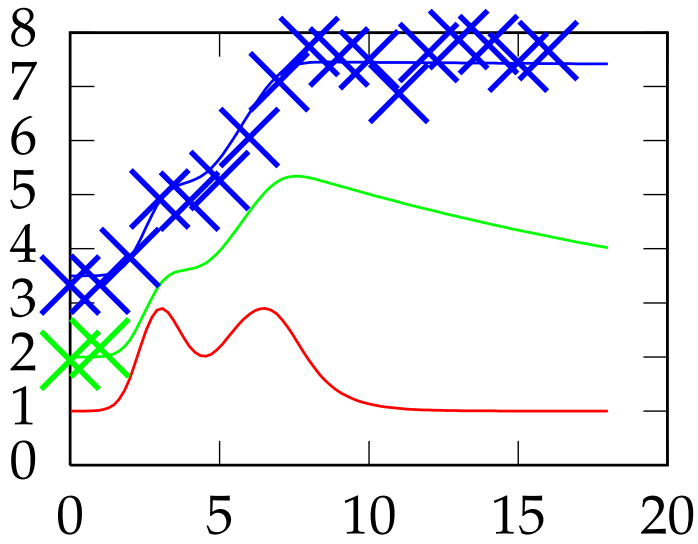

4
3
2
1
0
-1
-2
0
5
10
15
20

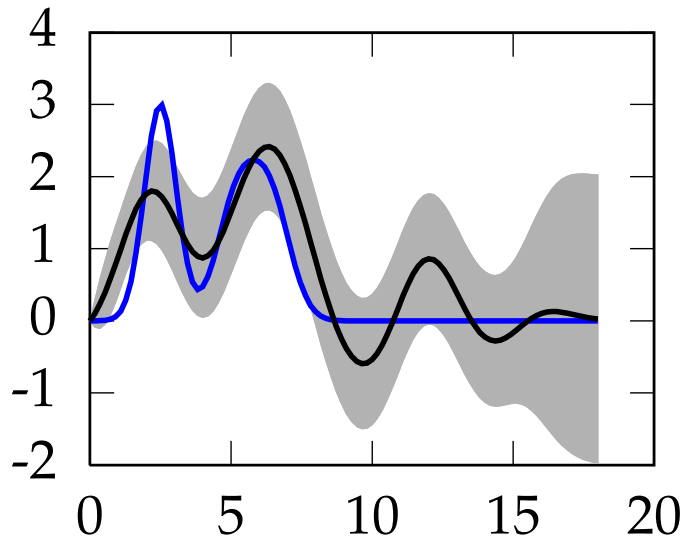

8
7
6
5
4
3
2
1
0
0
5
10
15
20

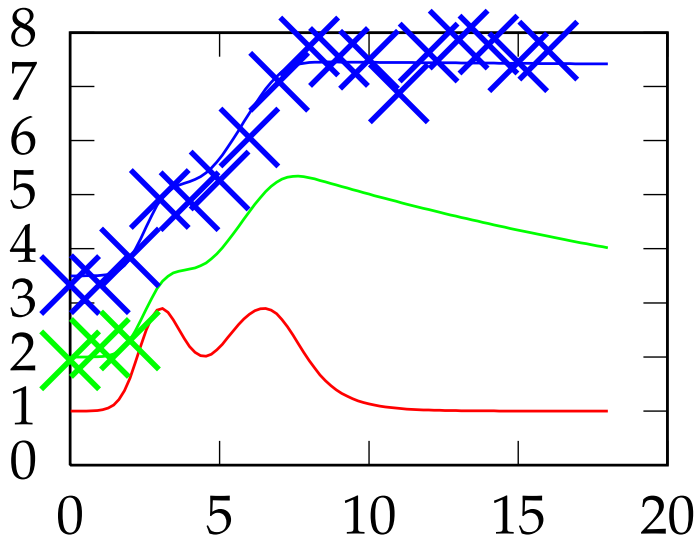

4
3
2
1
0
-1
-2
0
5
10
15
20

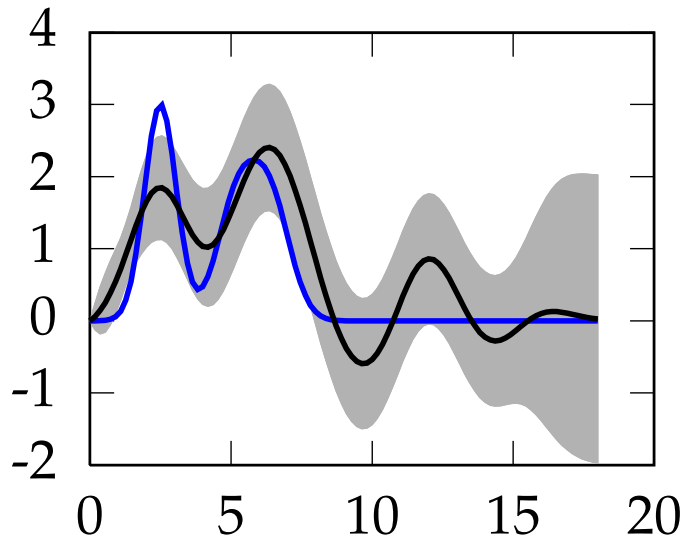

8
7
6
5
4
3
2
1
0
0
5
10
15
20

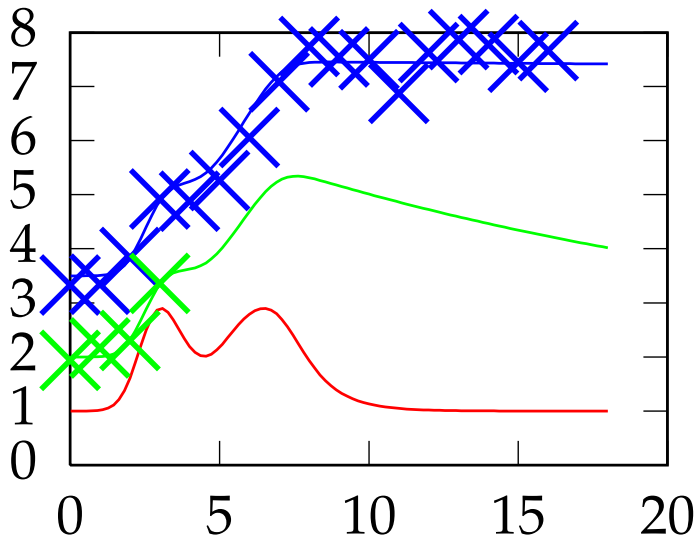

4
3
2
1
0
-1
-2
0
5
10
15
20

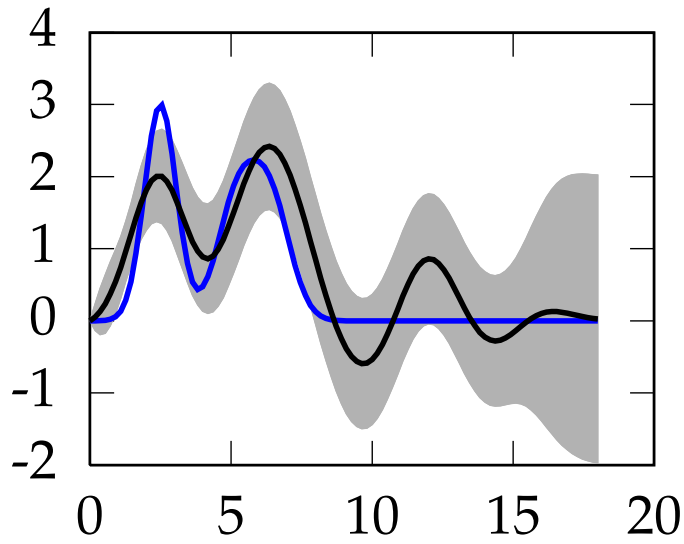

8
7
6
5
4
3
2
1
0
0
5
10
15
20

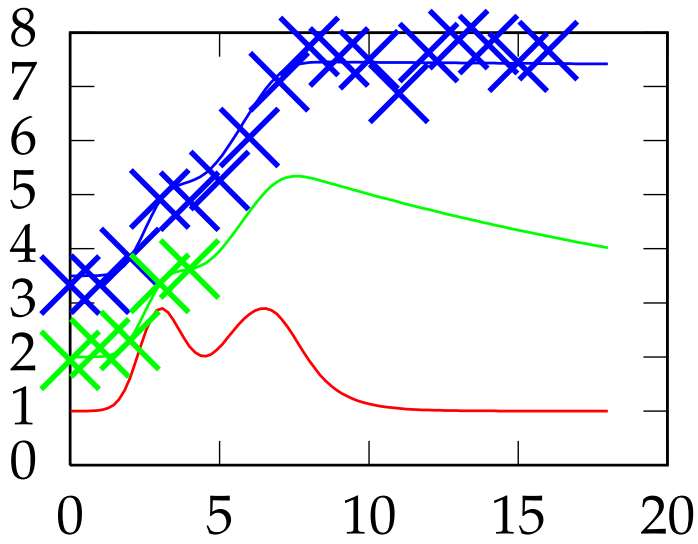

4
3
2
1
0
-1
-2
0
5
10
15
20

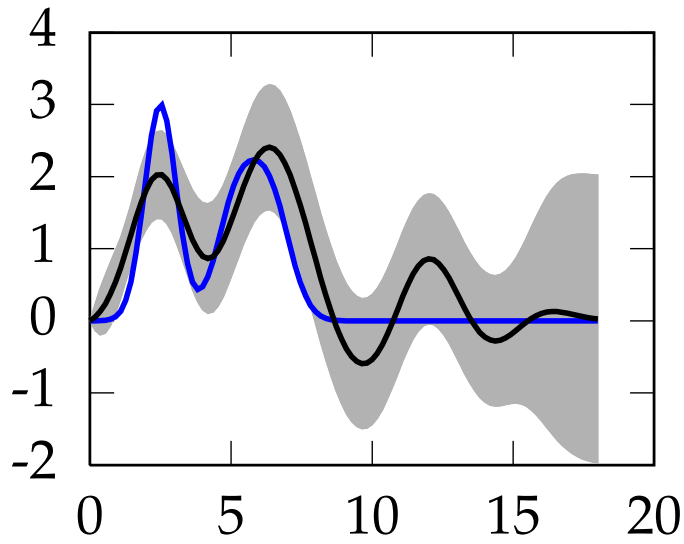

8
7
6
5
4
3
2
1
0
0
5
10
15
20

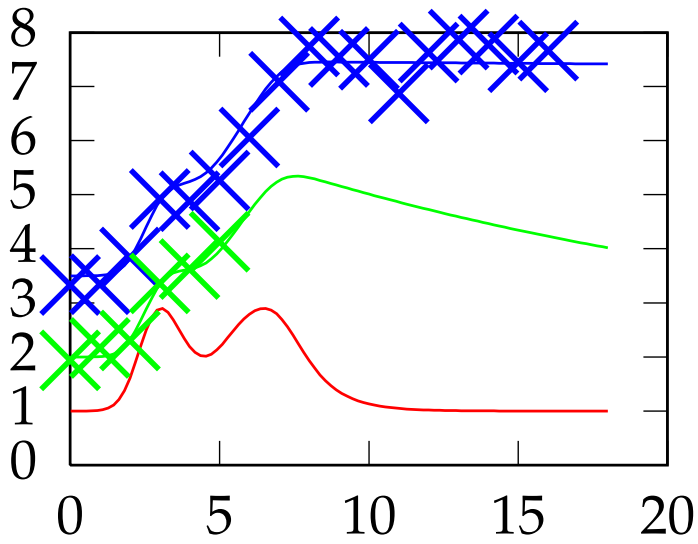

4
3
2
1
0
-1
-2
0
5
10
15
20

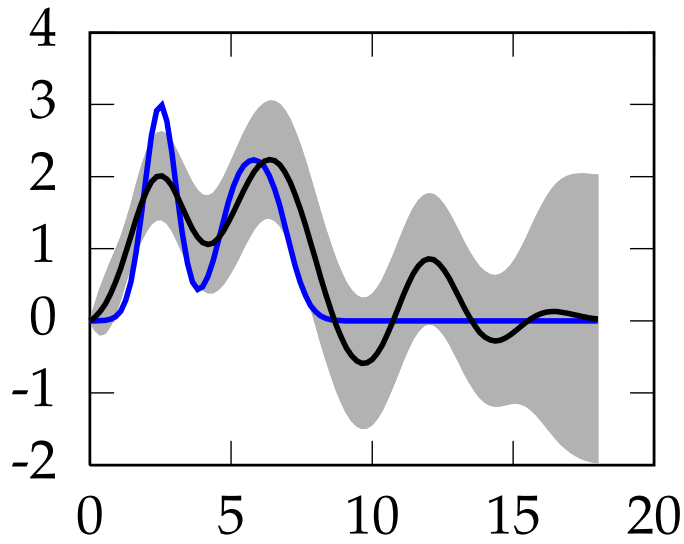

8
7
6
5
4
3
2
1
0
0
5
10
15
20

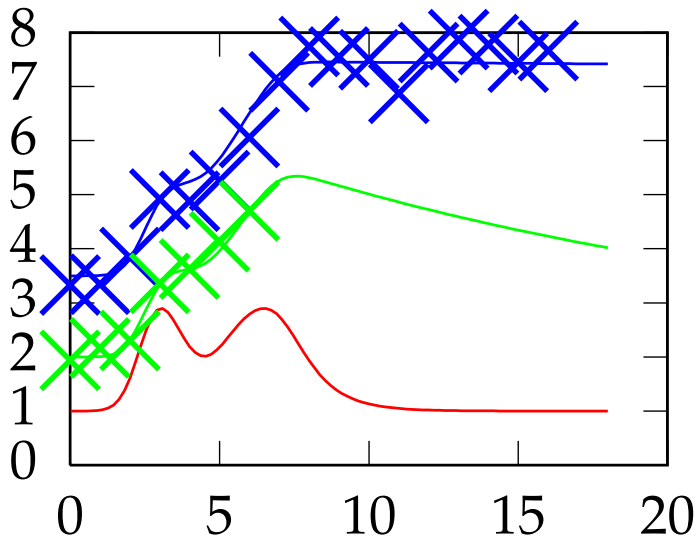

4
3
2
1
0
-1
-2
0
5
10
15
20

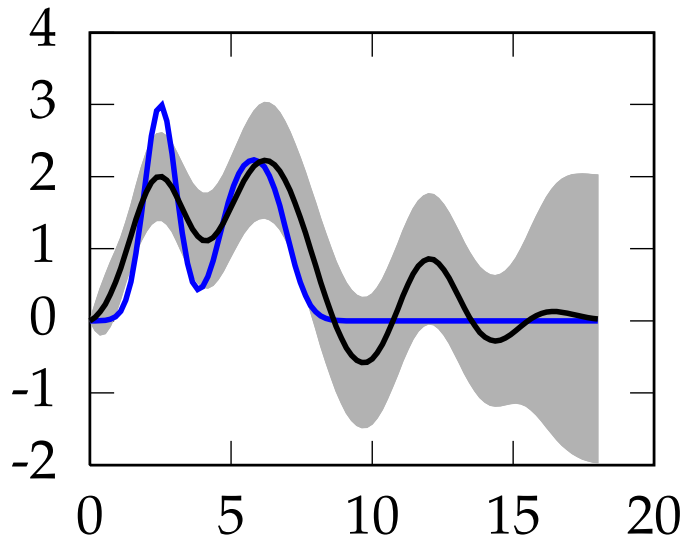

8
7
6
5
4
3
2
1
0
0
5
10
15
20

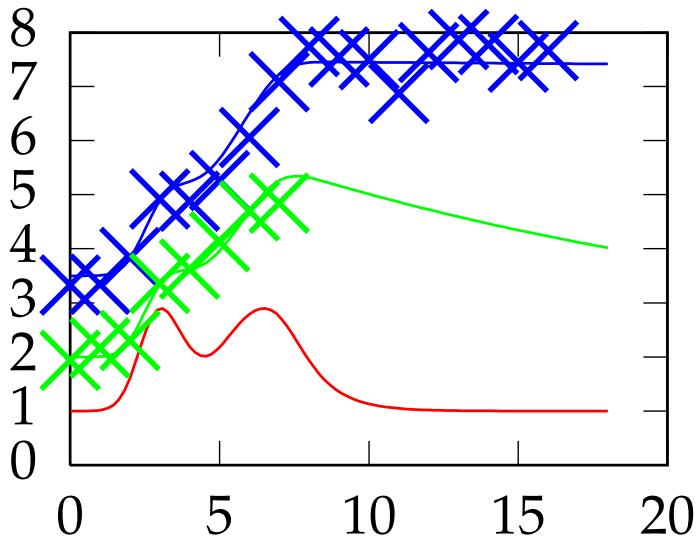

4
3
2
1
0
-1
-2
0
5
10
15
20

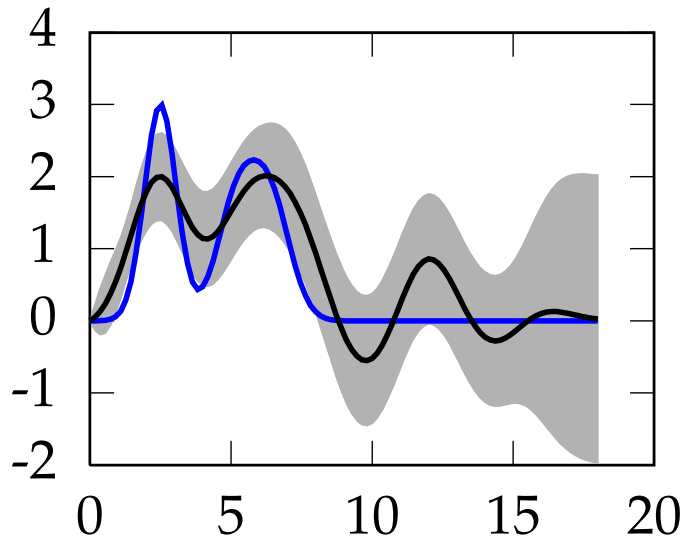

8
7
6
5
4
3
2
1
0
0
5
10
15
20

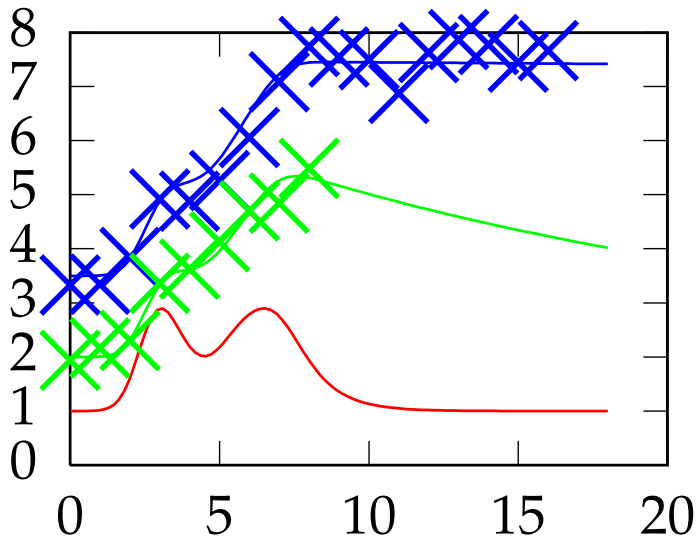

4
3
2
1
0
-1
-2
0
5
10
15
20

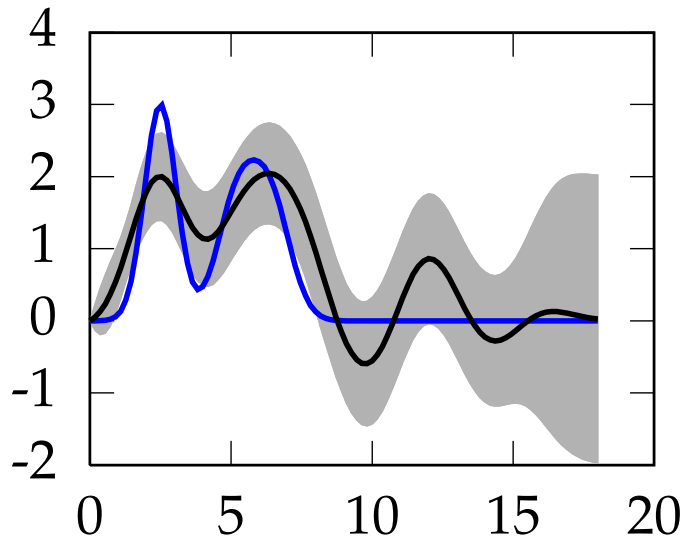

8
7
6
5
4
3
2
1
0
0
5
10
15
20

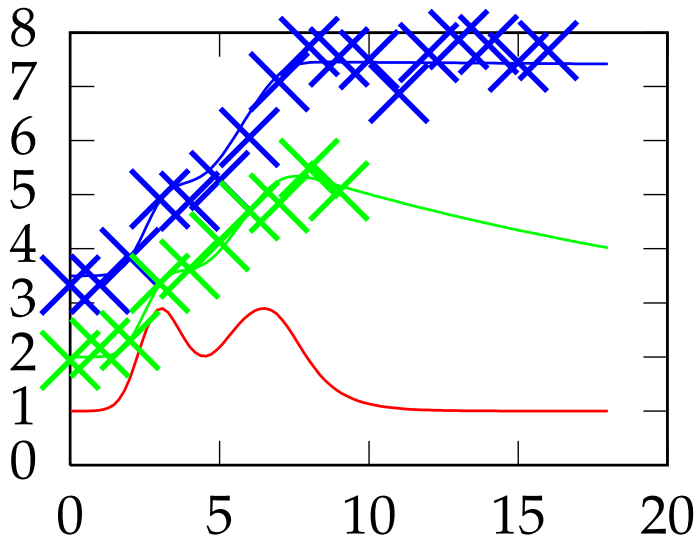

4
3
2
1
0
-1
-2
0
5
10
15
20

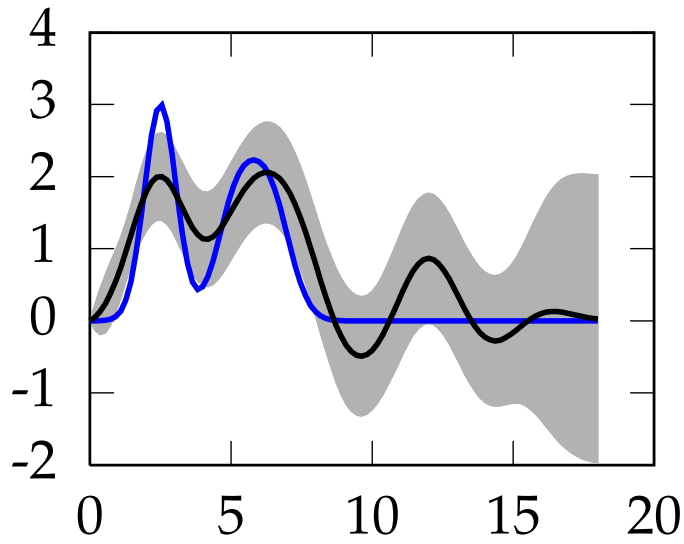

8
7
6
5
4
3
2
1
0
0
5
10
15
20

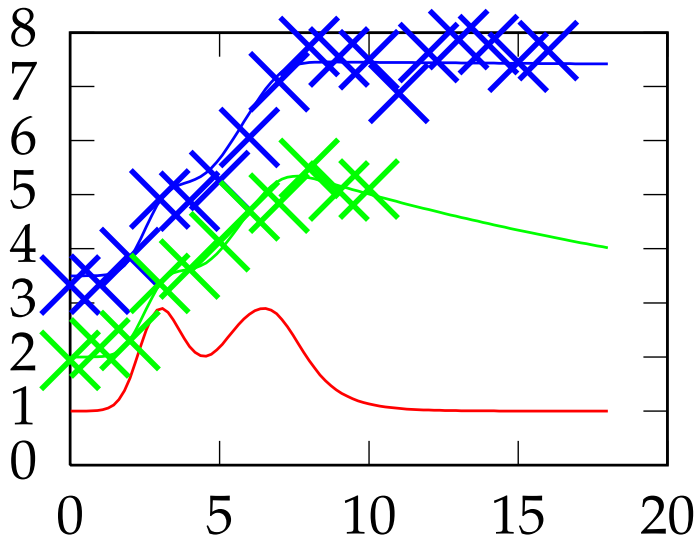

4
3
2
1
0
-1
-2
0
5
10
15
20

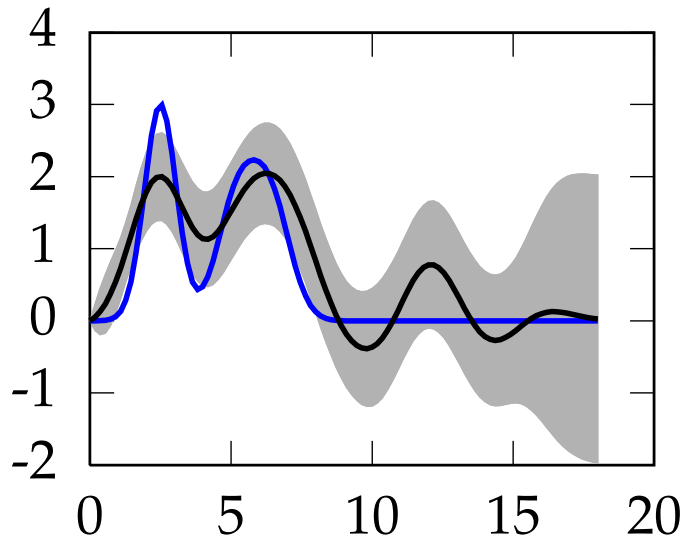

8
7
6
5
4
3
2
1
0
0
5
10
15
20

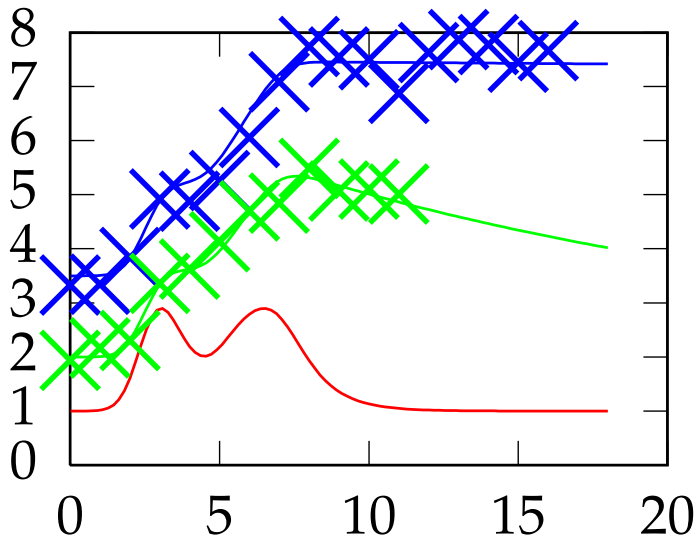

4
3
2
1
0
-1
-2
0
5
10
15
20

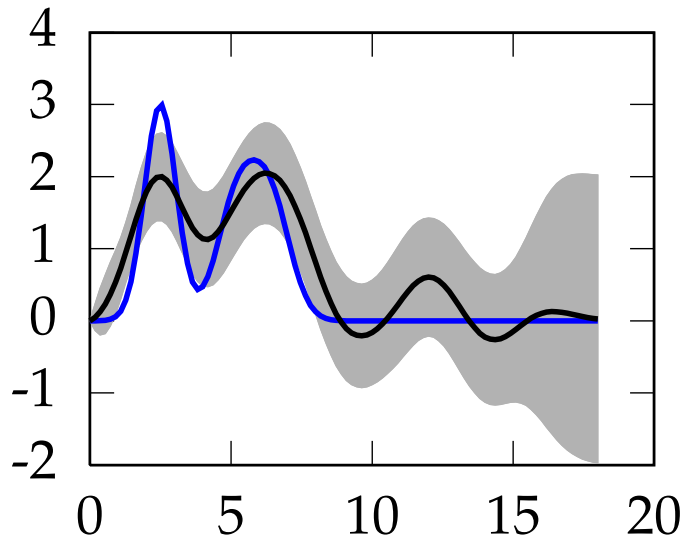

8
7
6
5
4
3
2
1
0
0
5
10
15
20

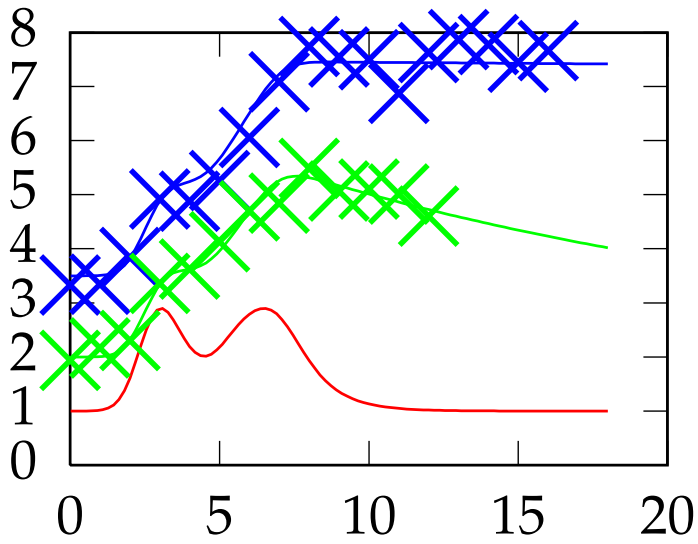

4
3
2
1
0
-1
-2
0
5
10
15
20

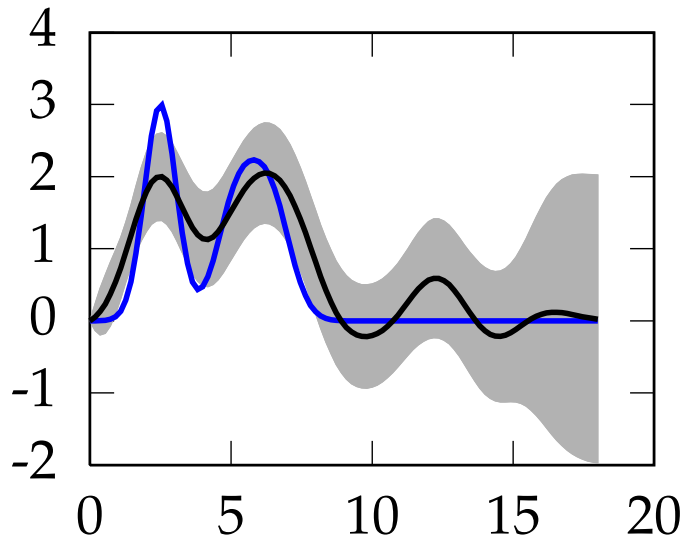

8
7
6
5
4
3
2
1
0
0
5
10
15
20

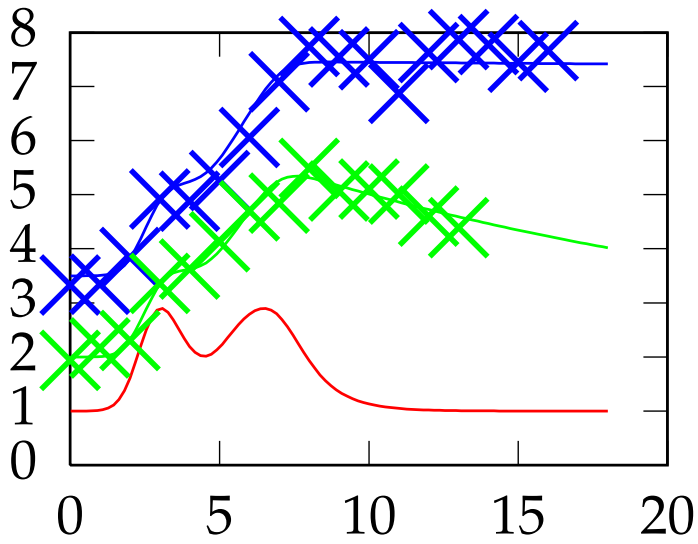

4
3
2
1
0
-1
-2
0
5
10
15
20

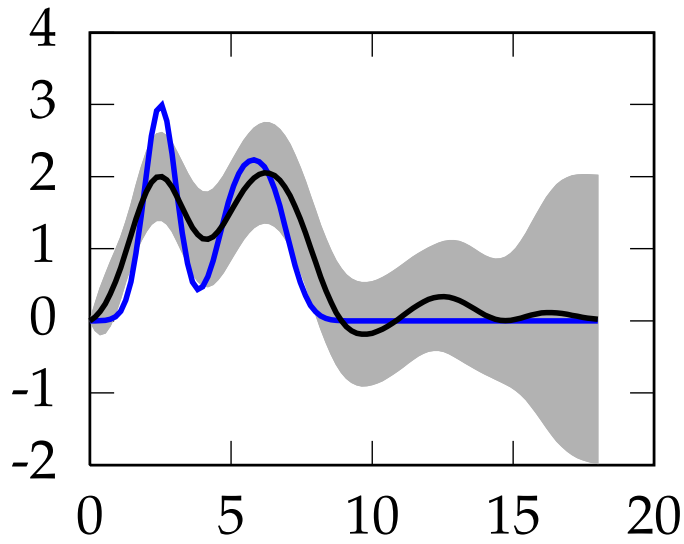

8
7
6
5
4
3
2
1
0
0
5
10
15
20

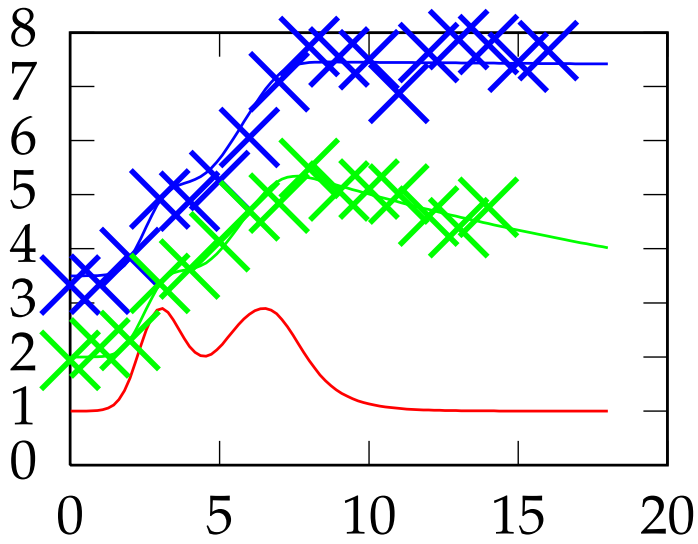

4
3
2
1
0
-1
-2
0
5
10
15
20

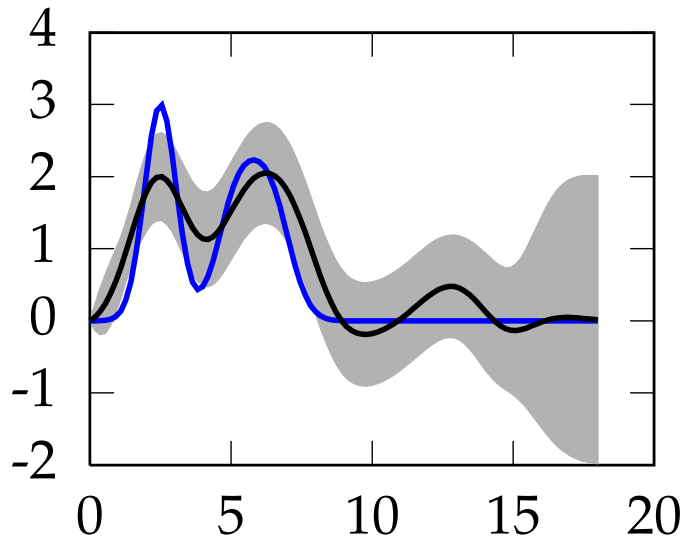

8
7
6
5
4
3
2
1
0
0
5
10
15
20

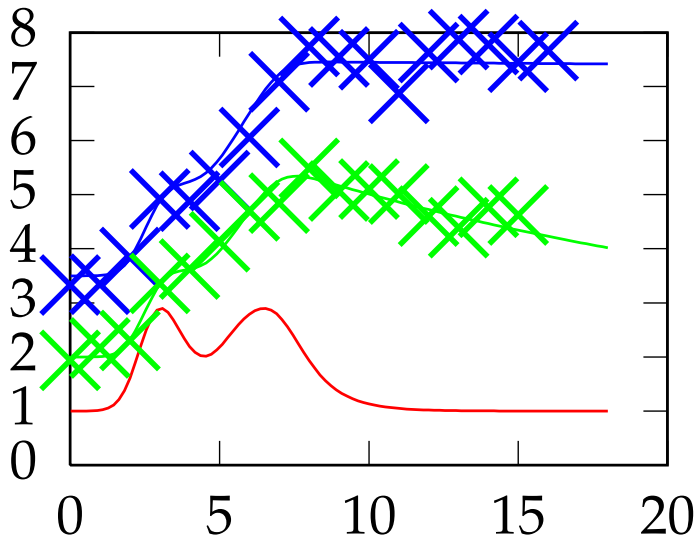

4
3
2
1
0
-1
-2
0
5
10
15
20

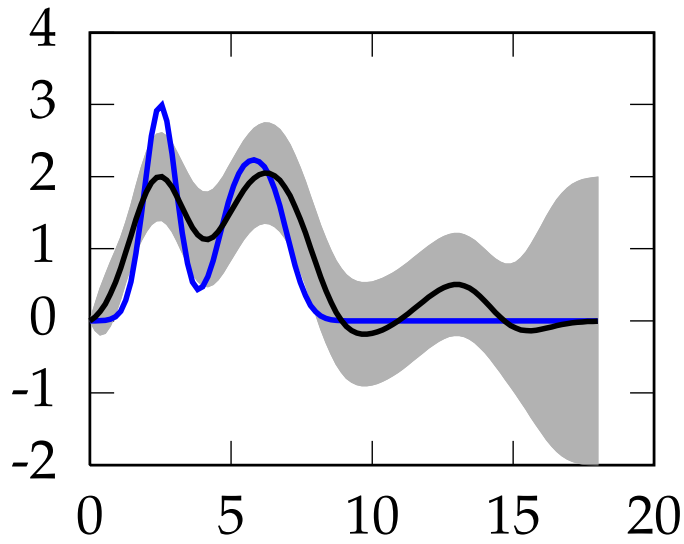

8
7
6
5
4
3
2
1
0
0
5
10
15
20

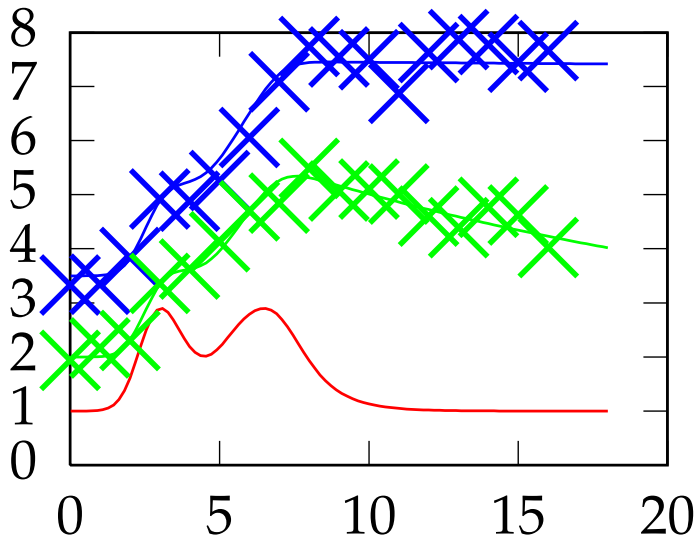

4
3
2
1
0
-1
-2
0
5
10
15
20

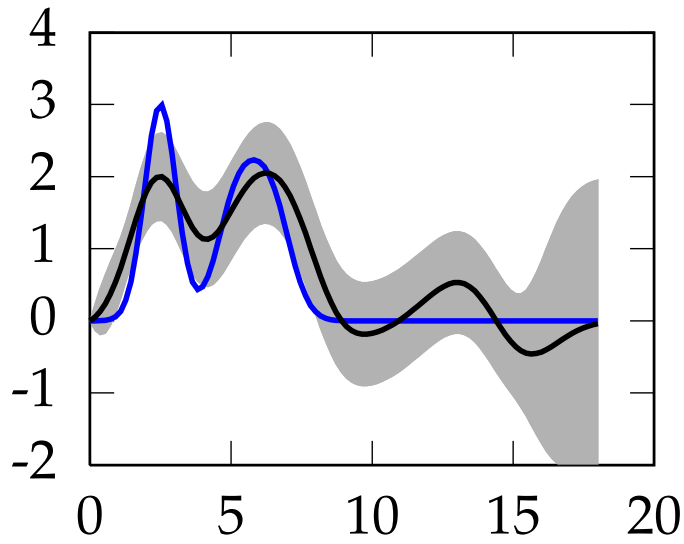

8
7
6
5
4
3
2
1
0
0
5
10
15
20

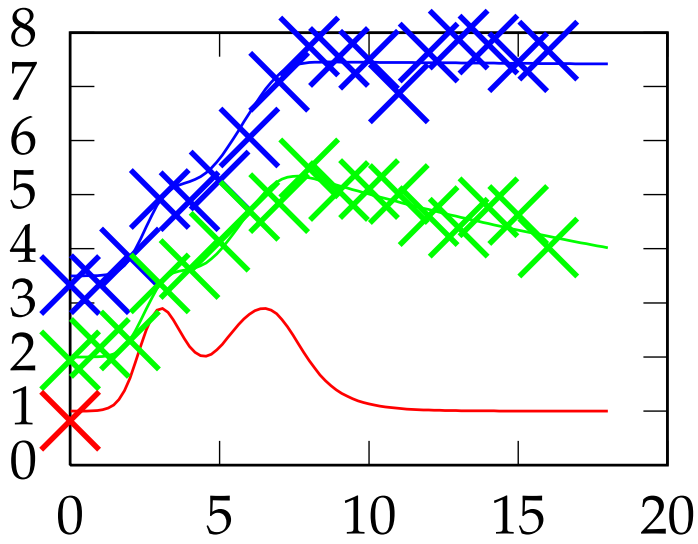

4
3
2
1
0
-1
-2
0
5
10
15
20

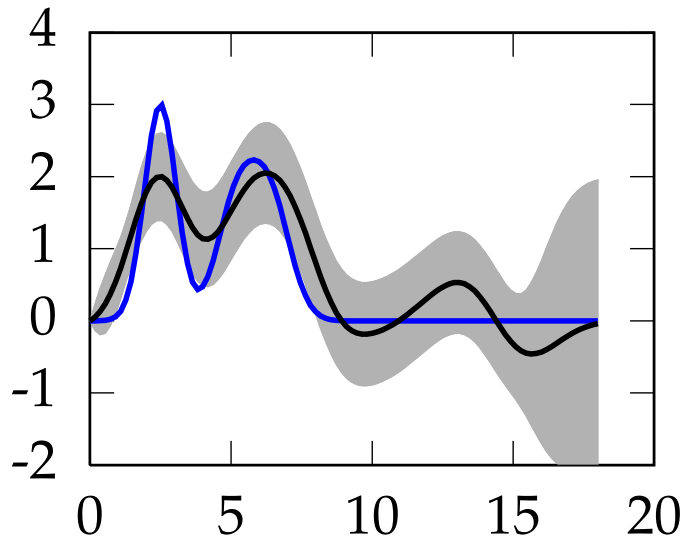

8
7
6
5
4
3
2
1
0
0
5
10
15
20

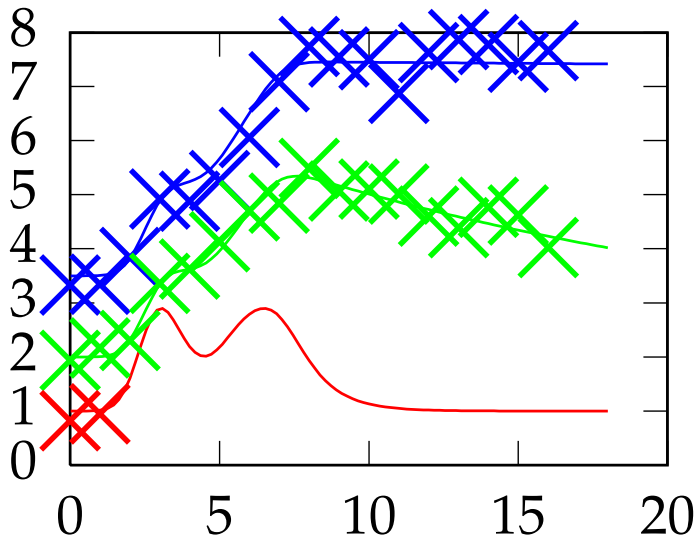

4
3
2
1
0
-1
-2
0
5
10
15
20

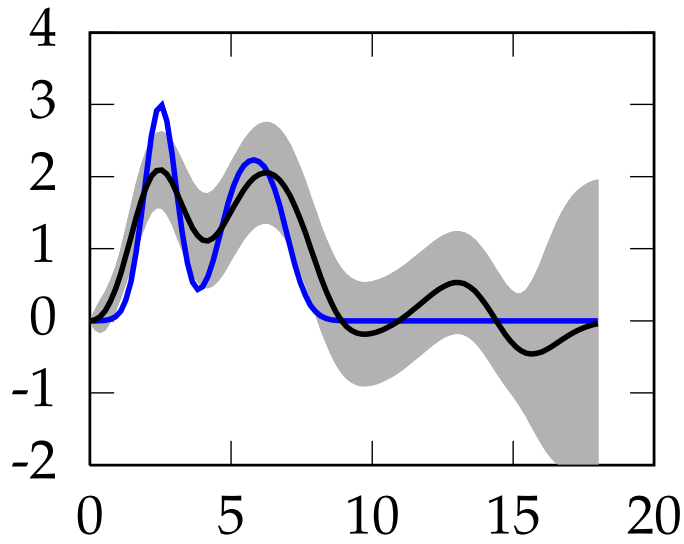

8
7
6
5
4
3
2
1
0
0
5
10
15
20

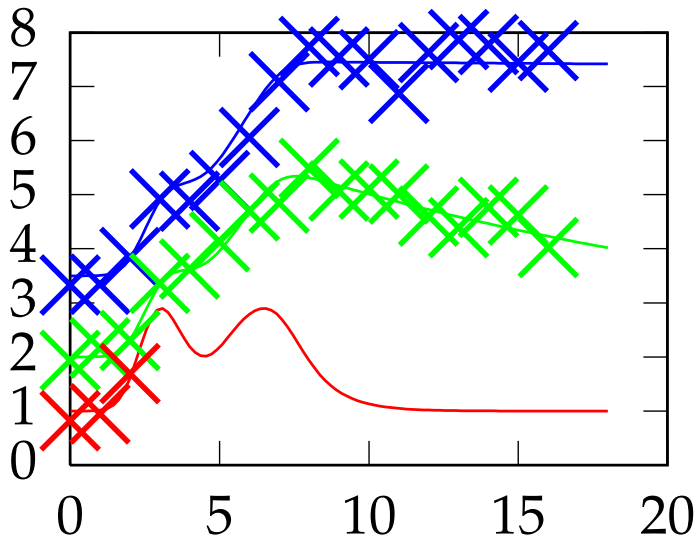

4
3
2
1
0
-1
-2
0
5
10
15
20

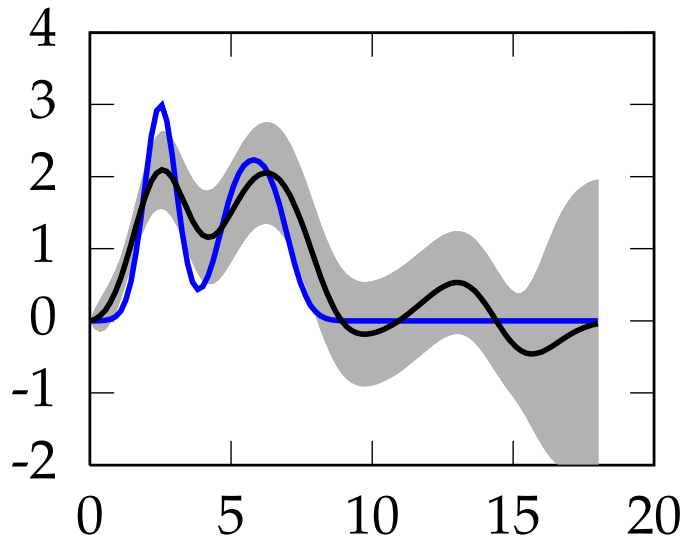

8
7
6
5
4
3
2
1
0
0
5
10
15
20

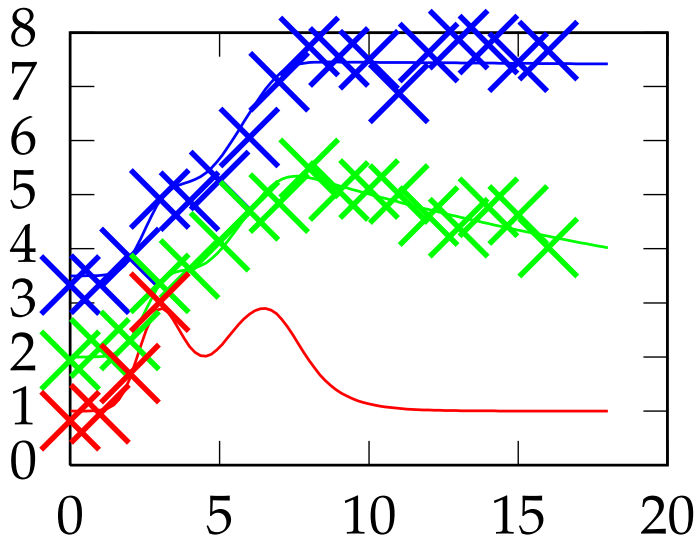

4
3
2
1
0
-1
-2
0
5
10
15
20

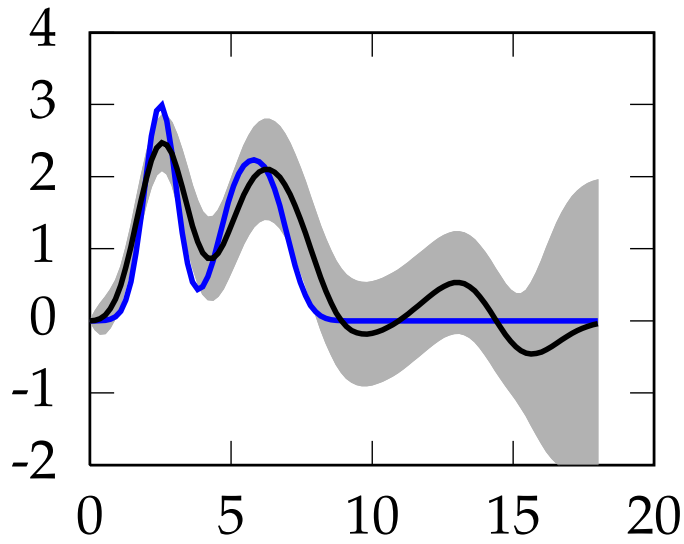

8
7
6
5
4
3
2
1
0
0
5
10
15
20

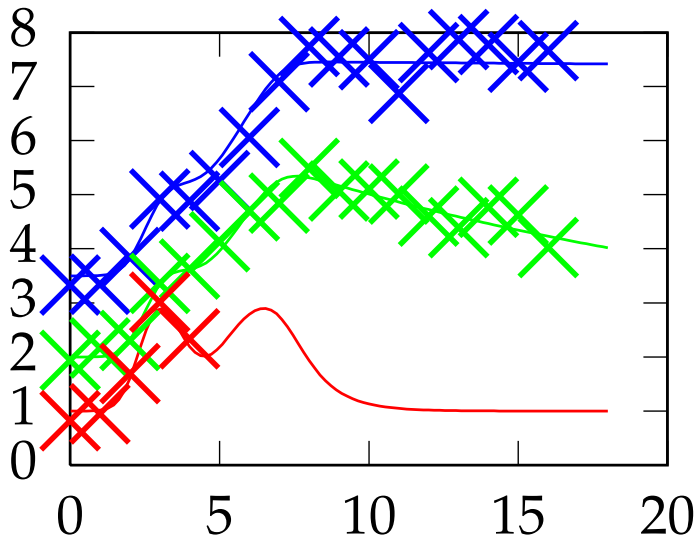

4
3
2
1
0
-1
-2
0
5
10
15
20

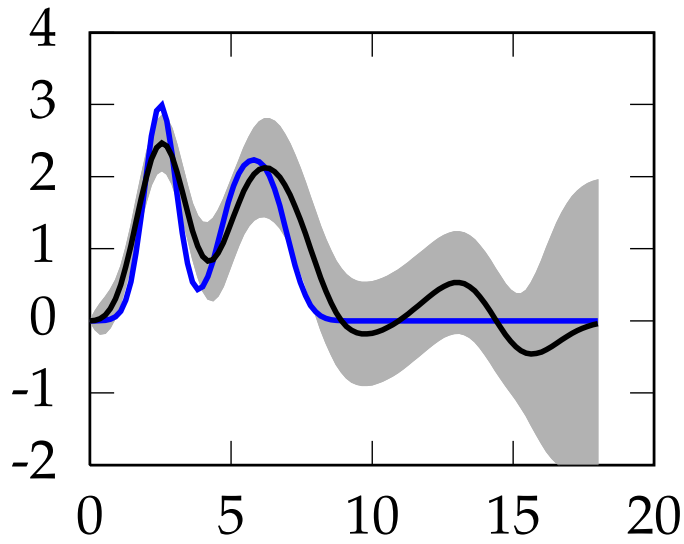

8
7
6
5
4
3
2
1
0
0
5
10
15
20

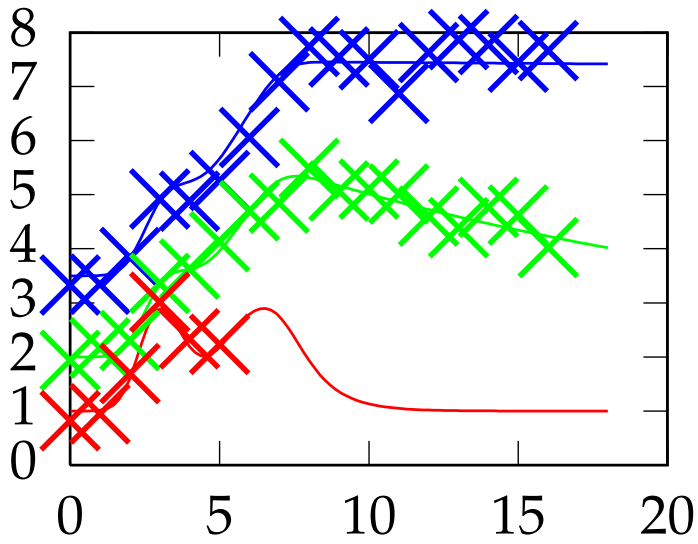

4
3
2
1
0
-1
-2
0
5
10
15
20

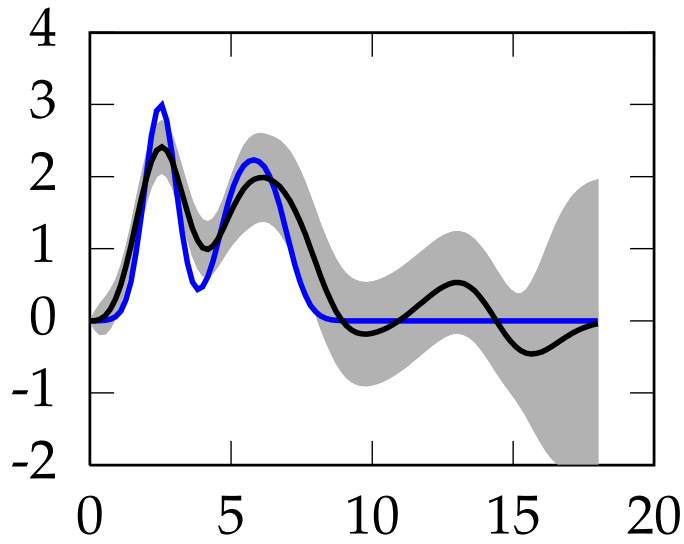

8
7
6
5
4
3
2
1
0
0
5
10
15
20

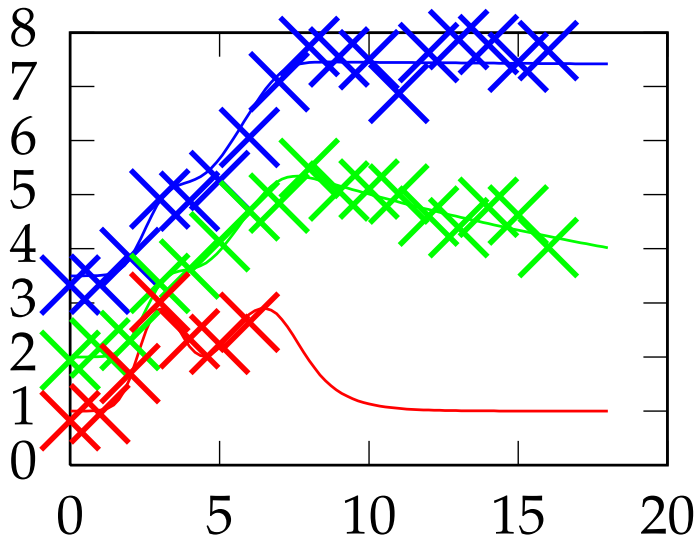

4
3
2
1
0
-1
-2
0
5
10
15
20

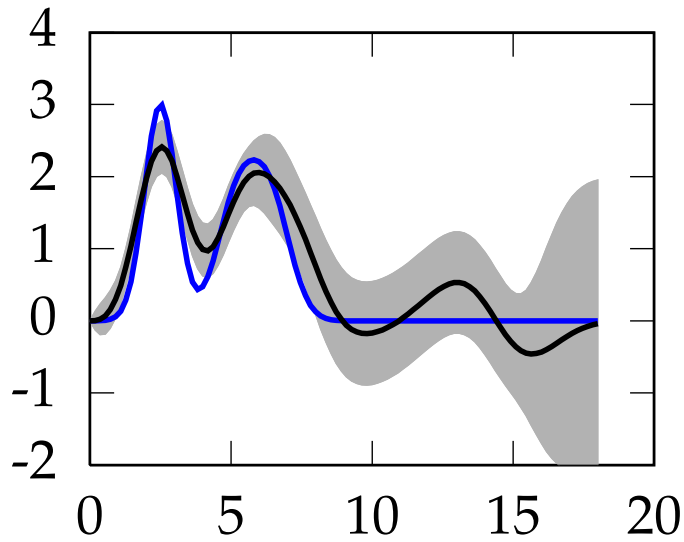

8
7
6
5
4
3
2
1
0
0
5
10
15
20

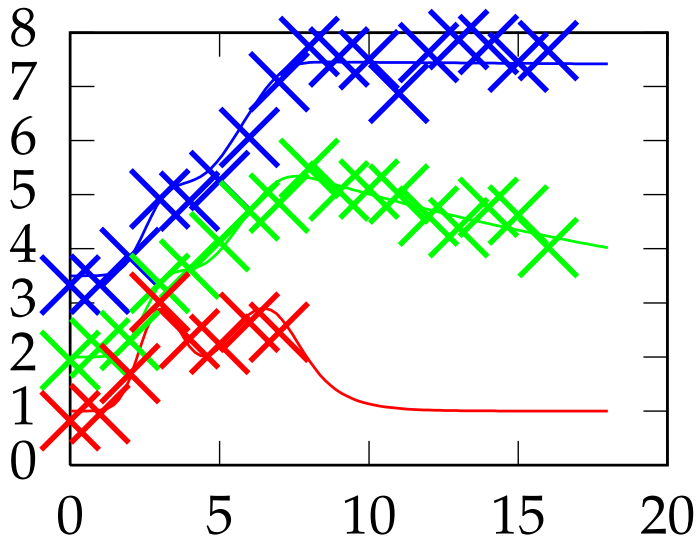

4
3
2
1
0
-1
-2
0
5
10
15
20

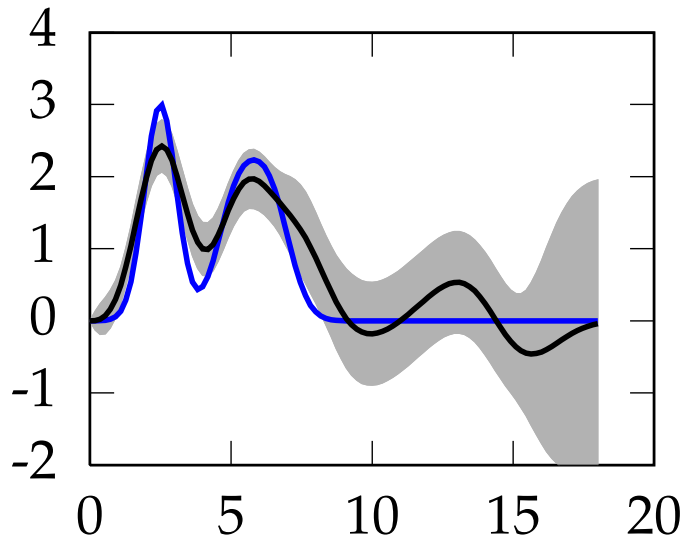

8
7
6
5
4
3
2
1
0
0
5
10
15
20

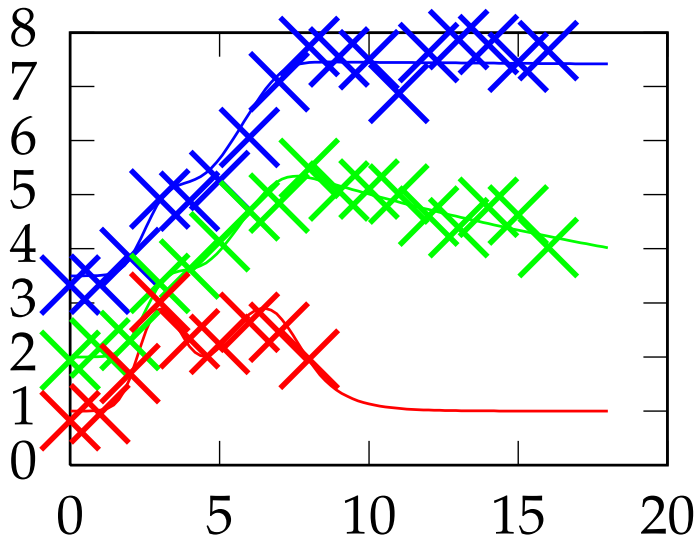

4
3
2
1
0
-1
-2
0
5
10
15
20

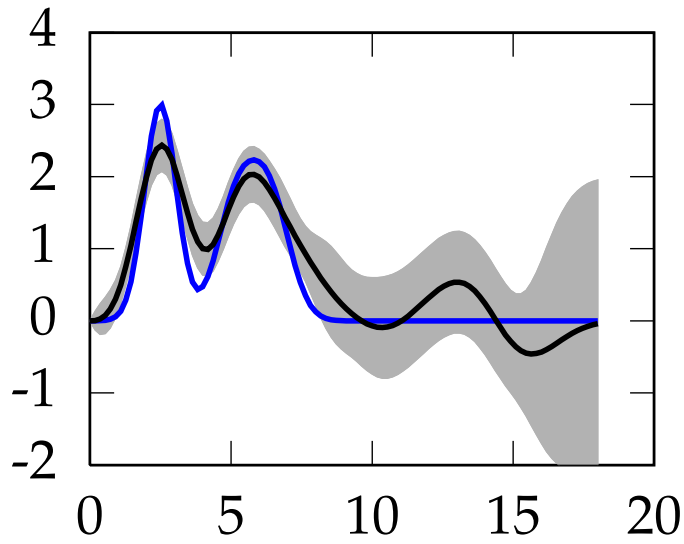

8
7
6
5
4
3
2
1
0
0
5
10
15
20

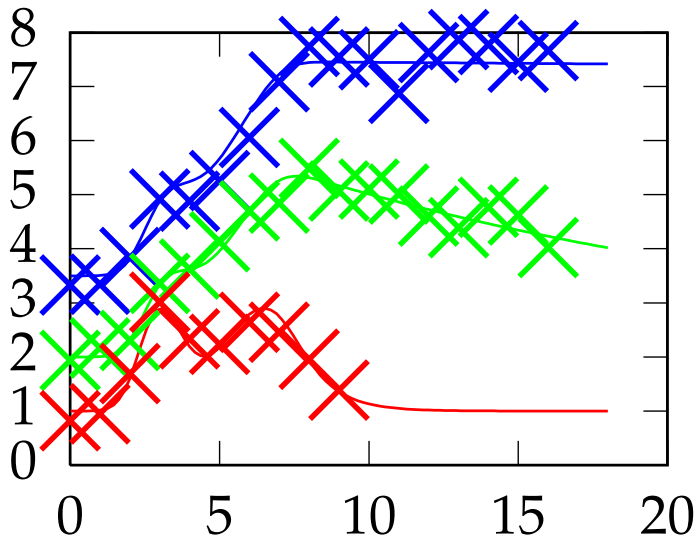

4
3
2
1
0
-1
-2
0
5
10
15
20

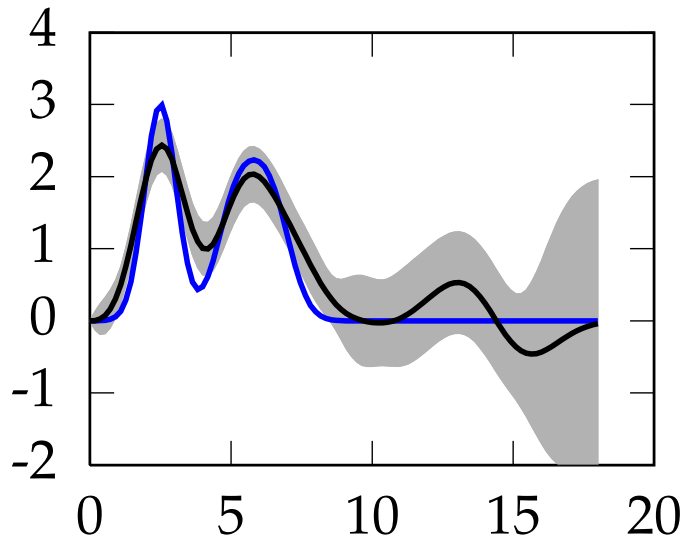

8
7
6
5
4
3
2
1
0
0
5
10
15
20

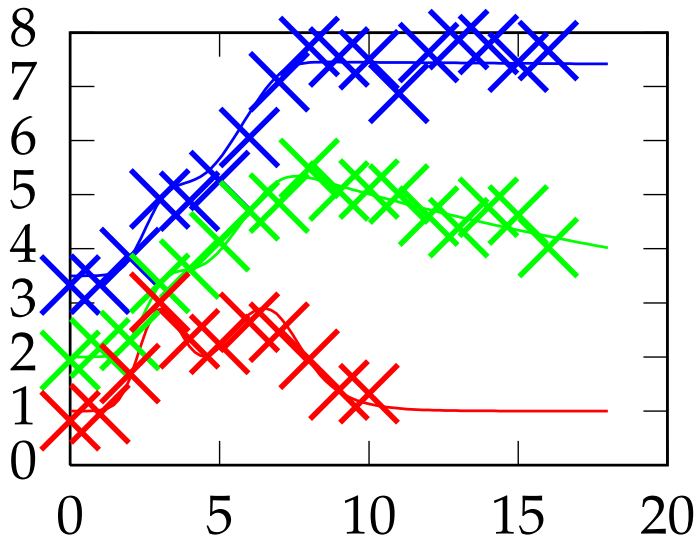

4
3
2
1
0
-1
-2
0
5
10
15
20

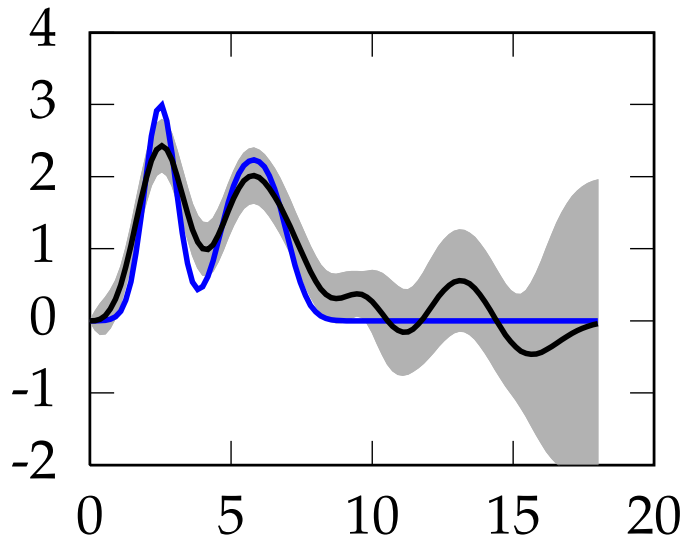

8
7
6
5
4
3
2
1
0
0
5
10
15
20

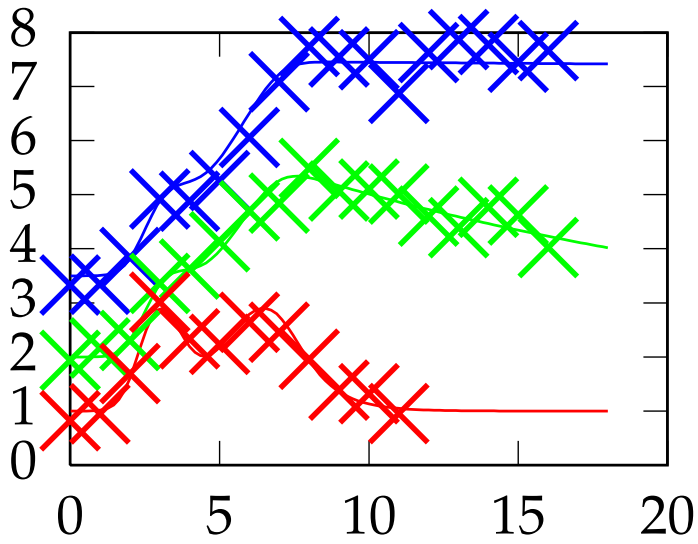

4
3
2
1
0
-1
-2
0
5
10
15
20

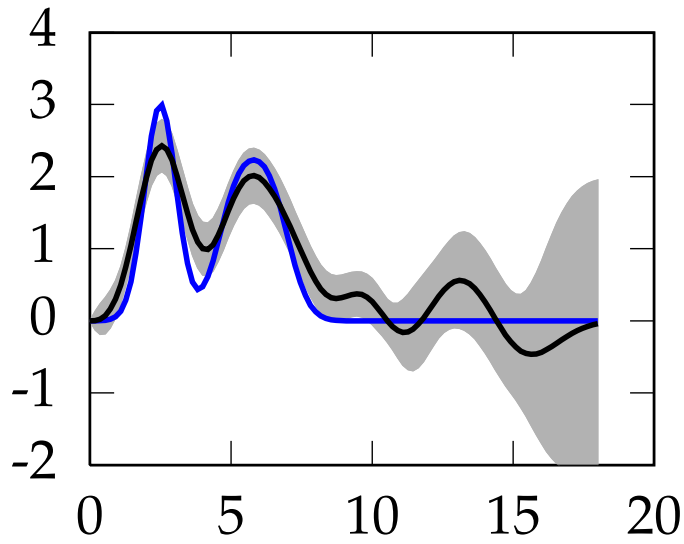

8
7
6
5
4
3
2
1
0
0
5
10
15
20

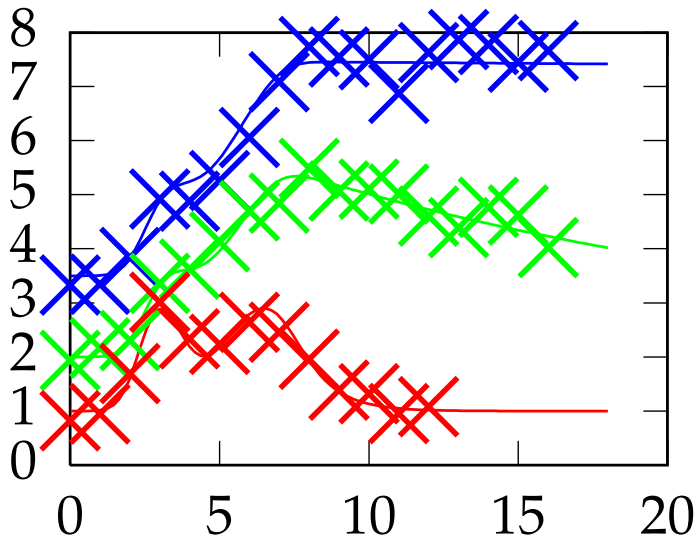

4
3
2
1
0
-1
-2
0
5
10
15
20

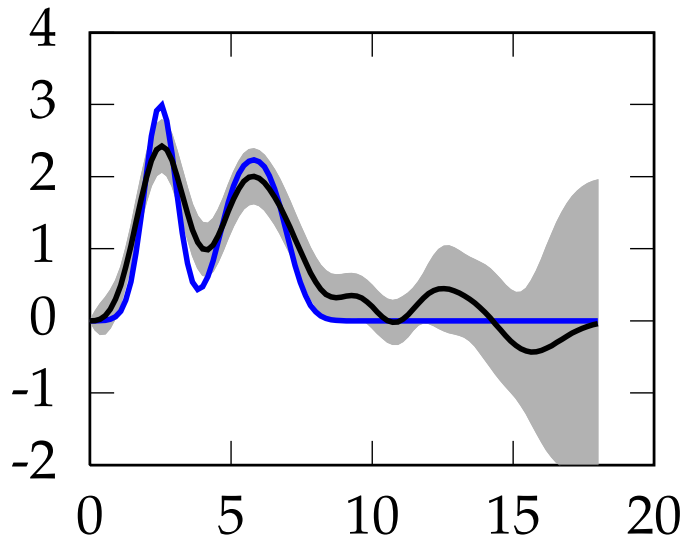

8
7
6
5
4
3
2
1
0
0
5
10
15
20

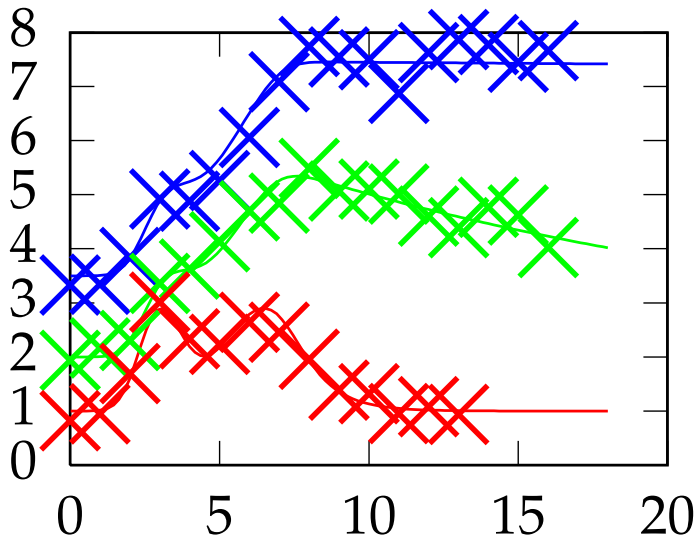

4
3
2
1
0
-1
-2
0
5
10
15
20

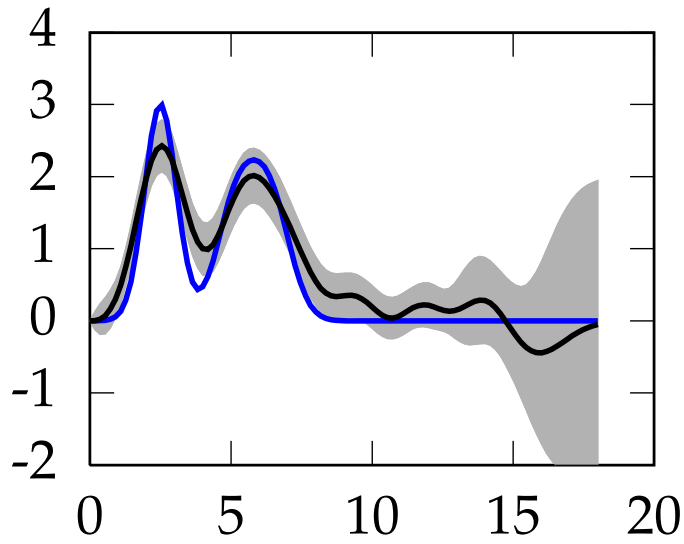

8
7
6
5
4
3
2
1
0
0
5
10
15
20

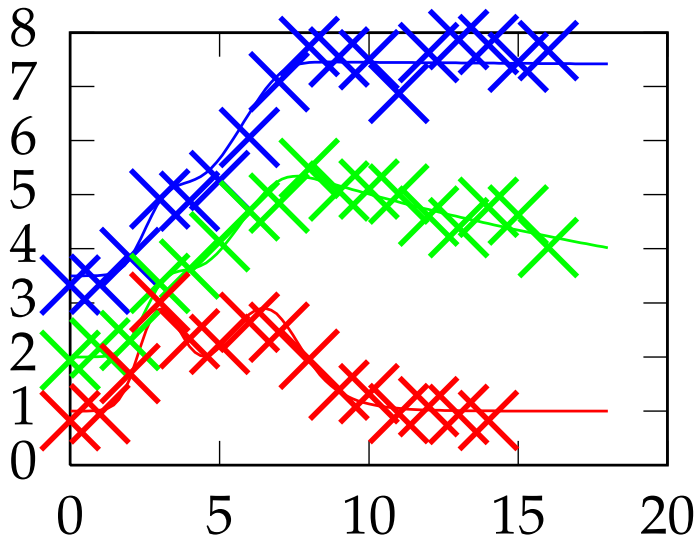

4
3
2
1
0
-1
-2
0
5
10
15
20

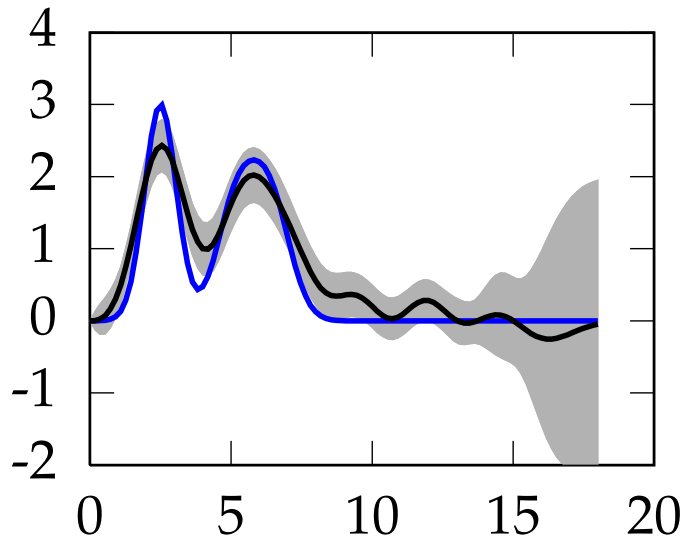

8
7
6
5
4
3
2
1
0
0
5
10
15
20

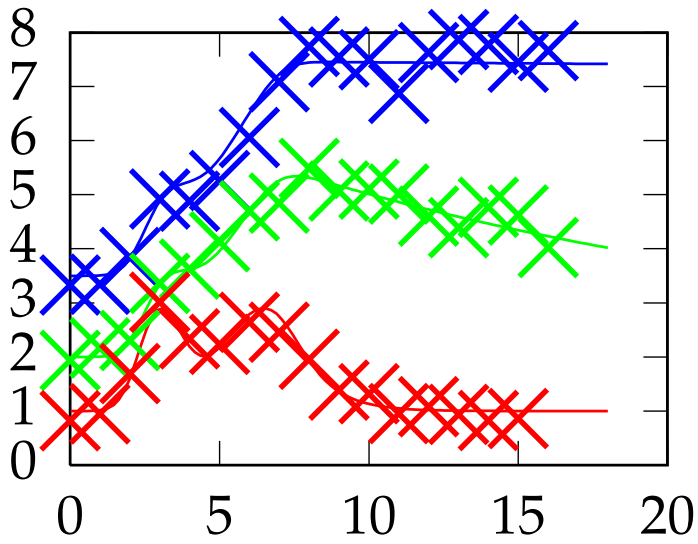

4
3
2
1
0
-1
-2
0
5
10
15
20

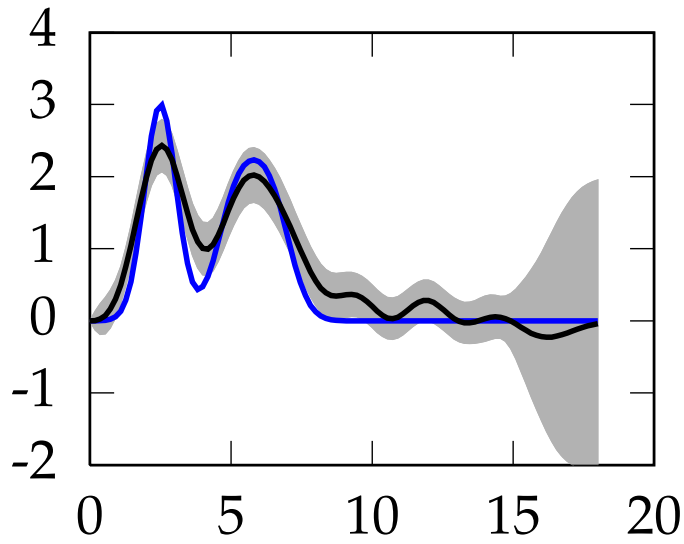

8
7
6
5
4
3
2
1
0
0
5
10
15
20

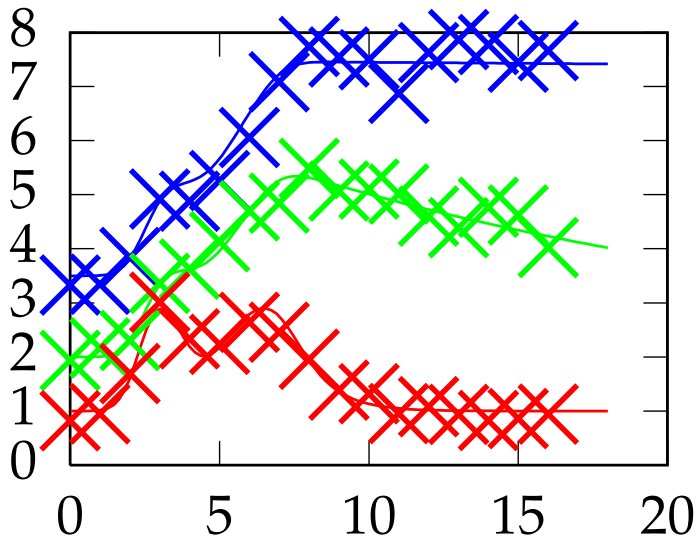

4
3
2
1
0
-1
-2
0
5
10
15
20

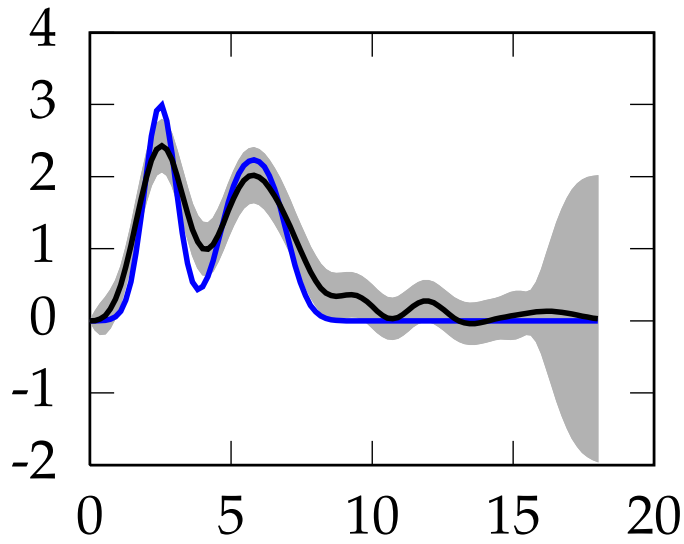

8
7
6
5
4
3
2
1
0
0
5
10
15
20


4
3
2
1
0
-1
-2
0
5
10
15
20


8
7
6
5
4
3
2
1
0
0
5
10
15
20


4
3
2
1
0
-1
-2
0
5
10
15
20


8
7
6
5
4
3
2
1
0
0
5
10
15
20


4
3
2
1
0
-1
-2
0
5
10
15
20

Radiation Damage in the Cell

- Radiation can damages molecules including DNA.
- Most DNA 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 G_1. In G_2 there are two copies of the chromosome. In G_1 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

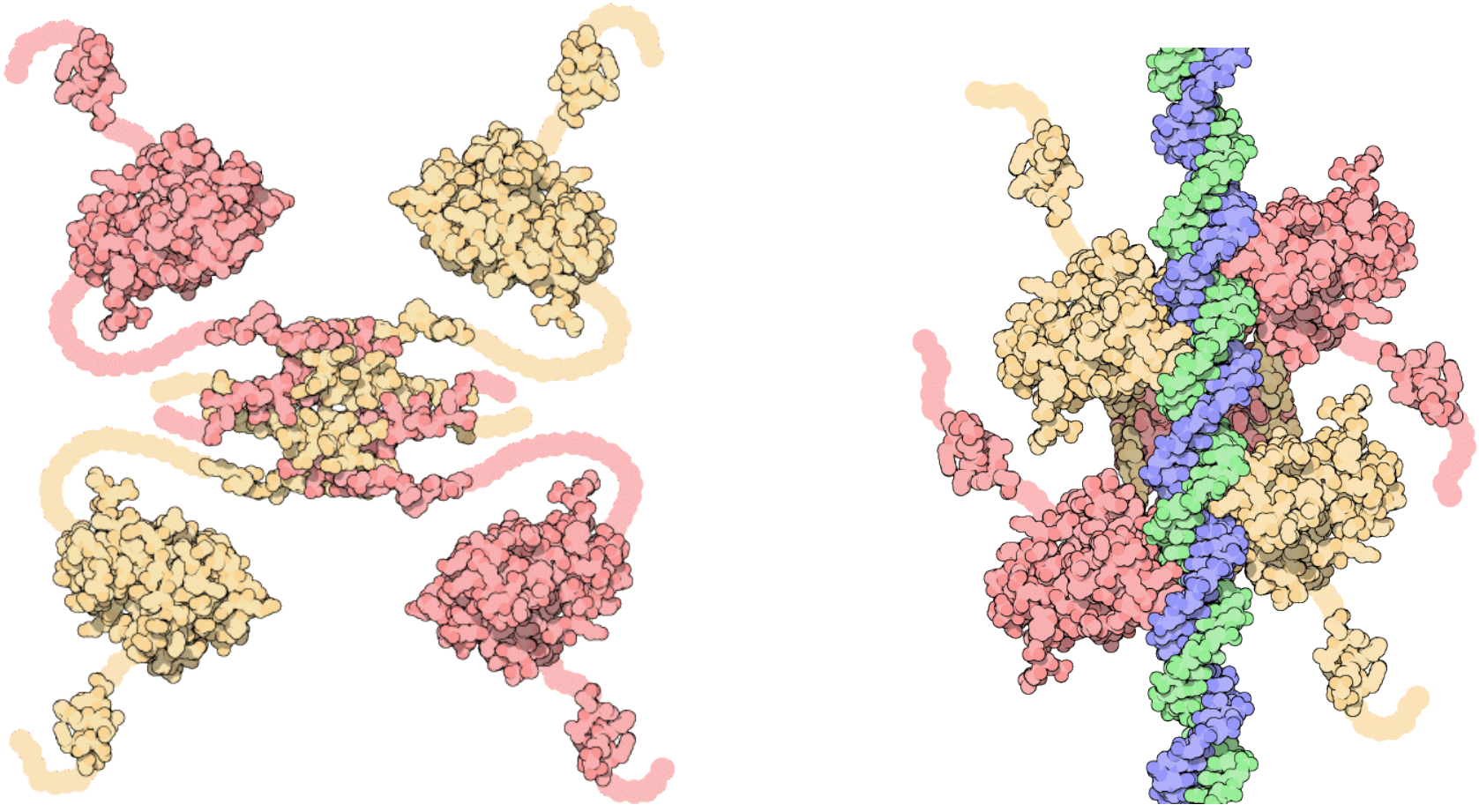


Figure : p53. *Left* unbound, *Right* bound to DNA. Images by David S. Goodsell from `http://www.rcsb.org/` (see the"Molecule of the Month" feature).

p53

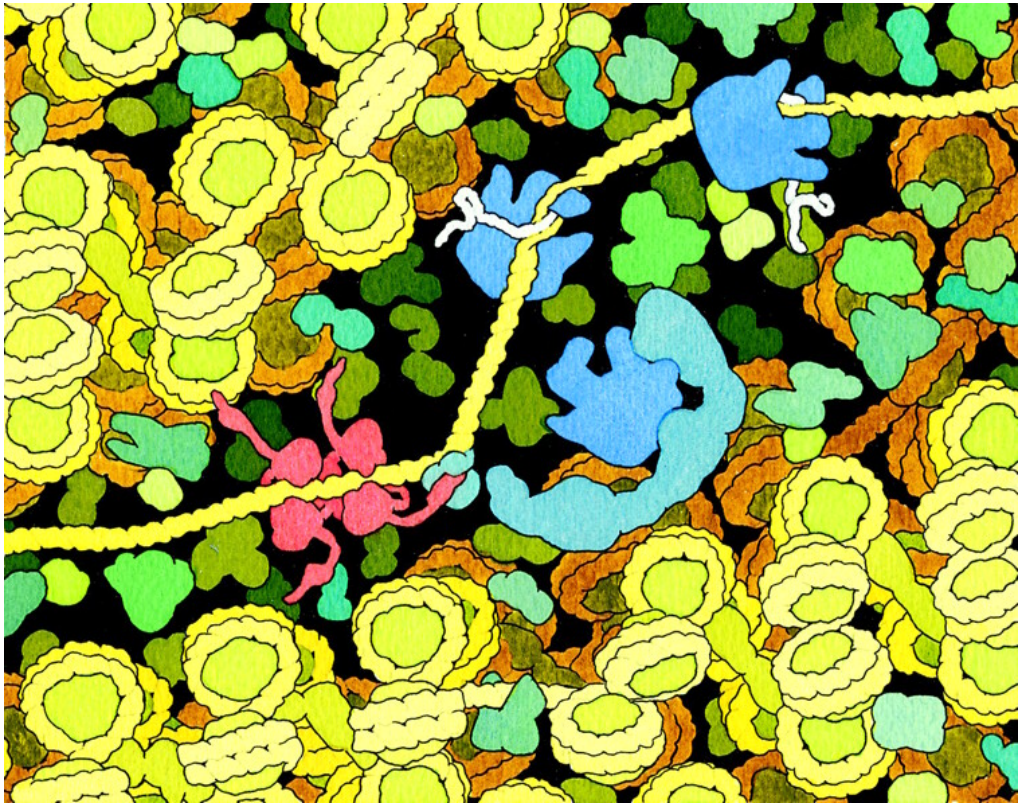


Figure : Repair of DNA damage by p53. Image from Goodsell (1999).

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 governed 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.

Modelling Assumption

- Assume p53 affects targets as a single input module network motif (SIM).

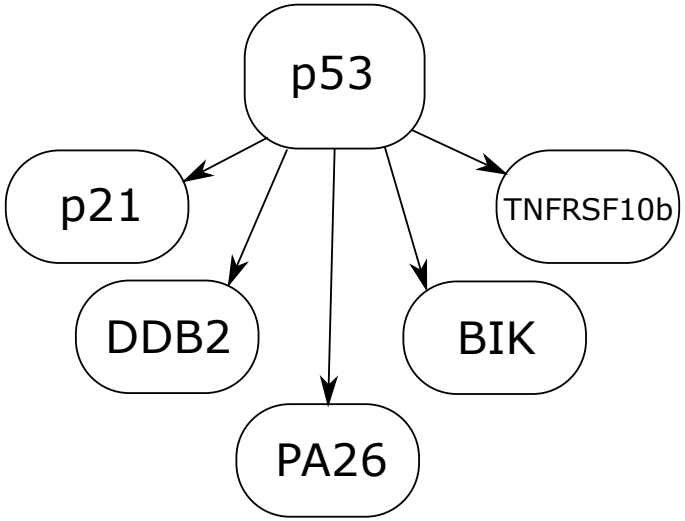


Figure : p53 SIM network motif as modelled by Barenco et al. 2006.

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

Ordinary Differential Equation Model

- First Order Differential Equation

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j p(t) - d_j m_j(t)$$

- Proposed by Barenco et al. (2006).
- $m_j(t)$ – concentration of gene j's mRNA
- $p(t)$ – 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 $p(t)$ is not observed?

p53 Results with GP

BIOINFORMATICS

Vol. 24 ECCB 2008, pages i70–i75
doi:10.1093/bioinformatics/btn278

Gaussian process modelling of latent chemical species: applications to inferring transcription factor activities

Pei Gao[1], Antti Honkela[2], Magnus Rattray[1] and Neil D. Lawrence[1,*]

[1]School of Computer Science, University of Manchester, Kilburn Building, Oxford Road, Manchester, M13 9PL and
[2]Adaptive Informatics Research Centre, Helsinki University of Technology, PO Box 5400, FI-02015 TKK, Finland

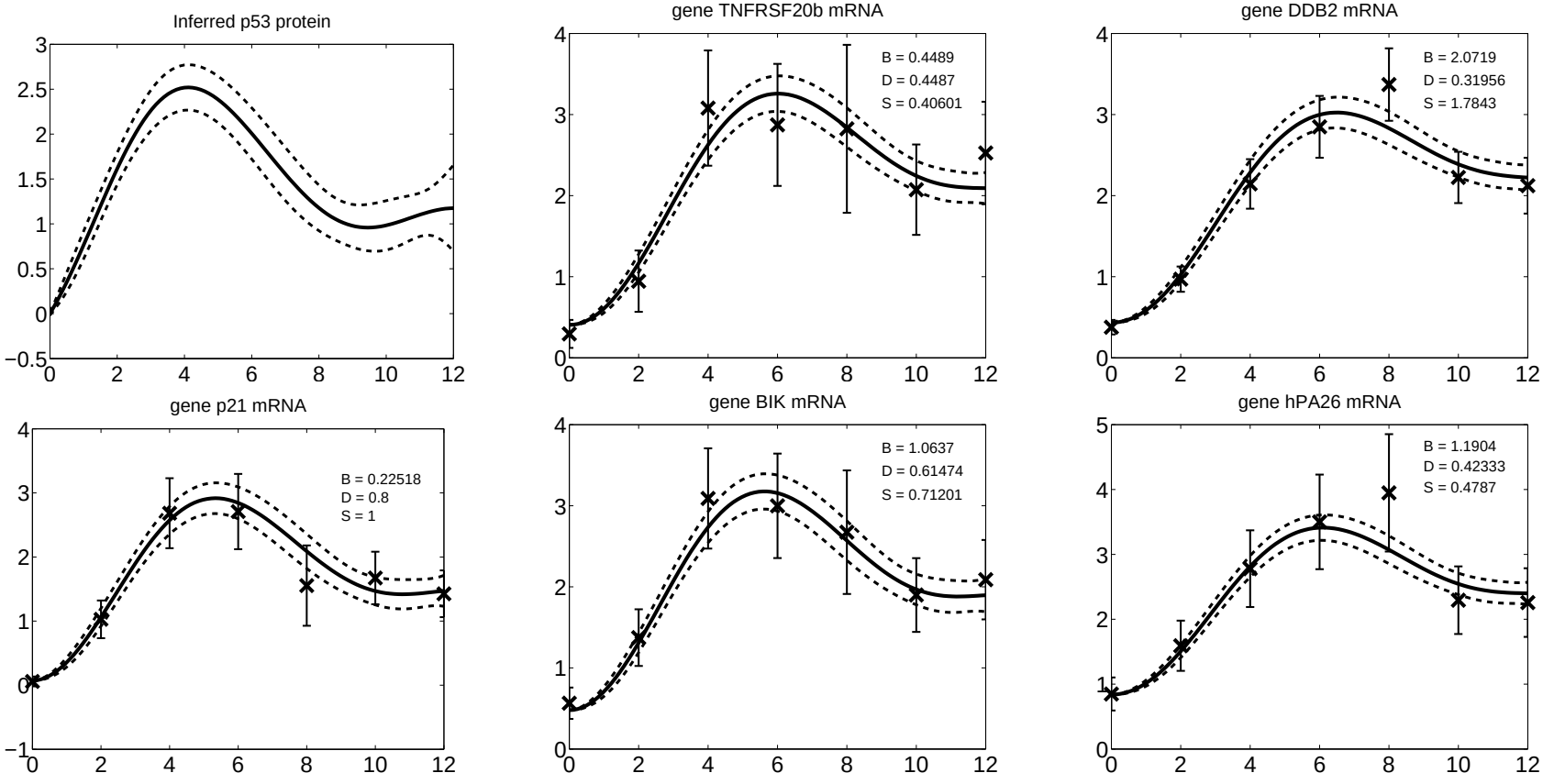
ABSTRACT

Motivation: Inference of *latent chemical species* in biochemical interaction networks is a key problem in estimation of the structure

A challenging problem for parameter estimation in ODE models occurs where one or more chemical species influencing the dynamics are controlled outside of the sub-system being modelled. For

p53 Results with GP

(Gao et al., 2008)

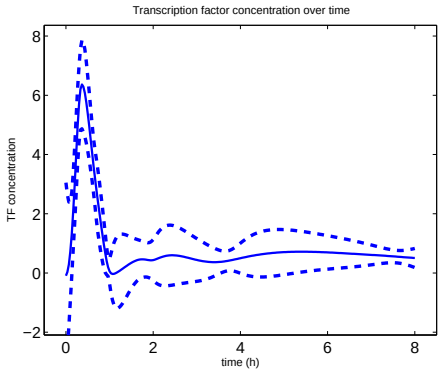


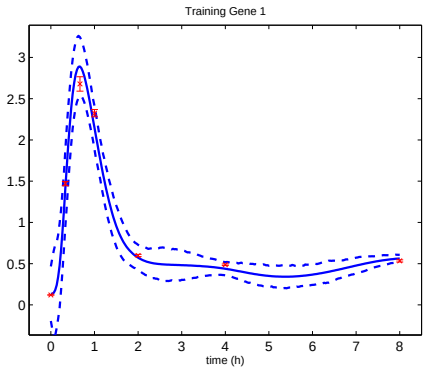
Ranking with ERK Signalling

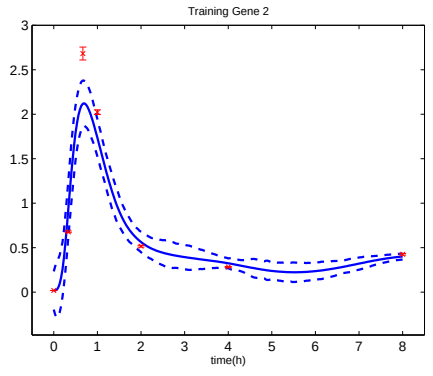
- Target Ranking for Elk-1.
- Elk-1 is phosphorylated by ERK from the EGF signalling pathway.
- Predict concentration of Elk-1 from known targets.
- Rank other targets of Elk-1.

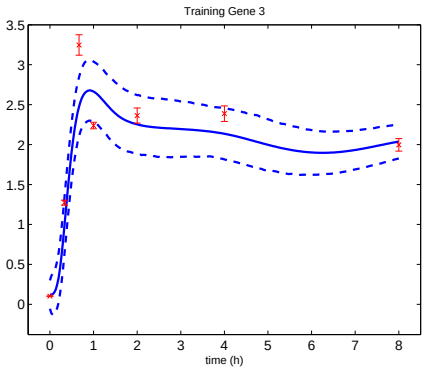
Elk-1 (MLP covariance)

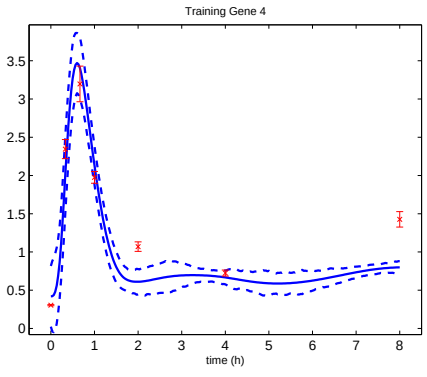
Jennifer Withers

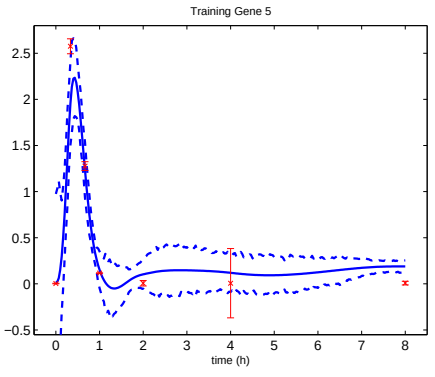






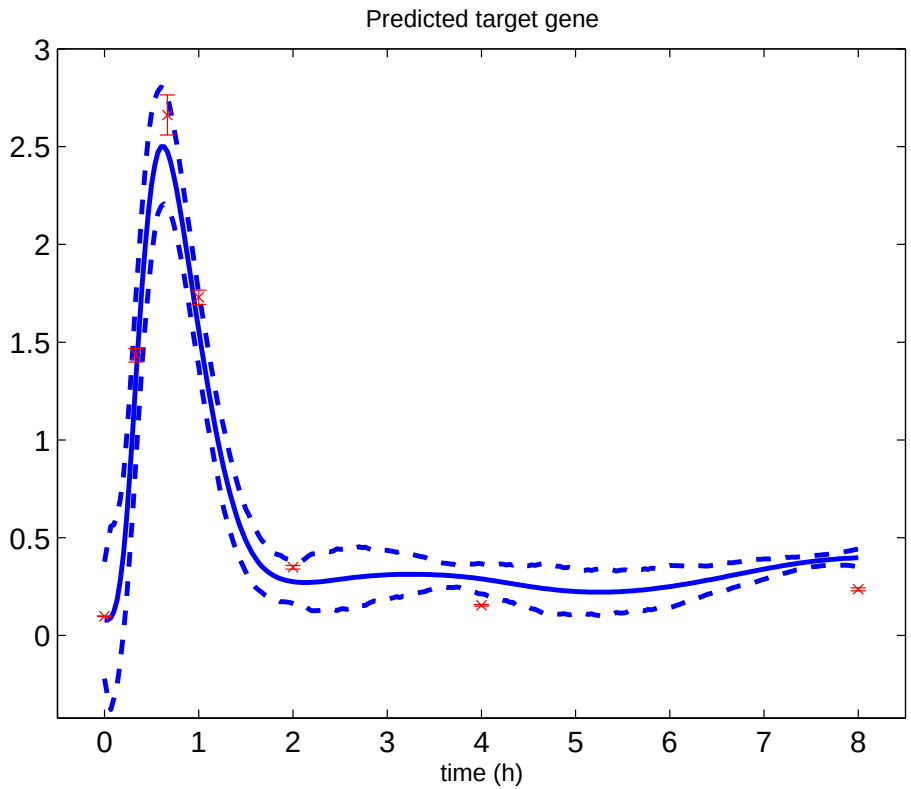


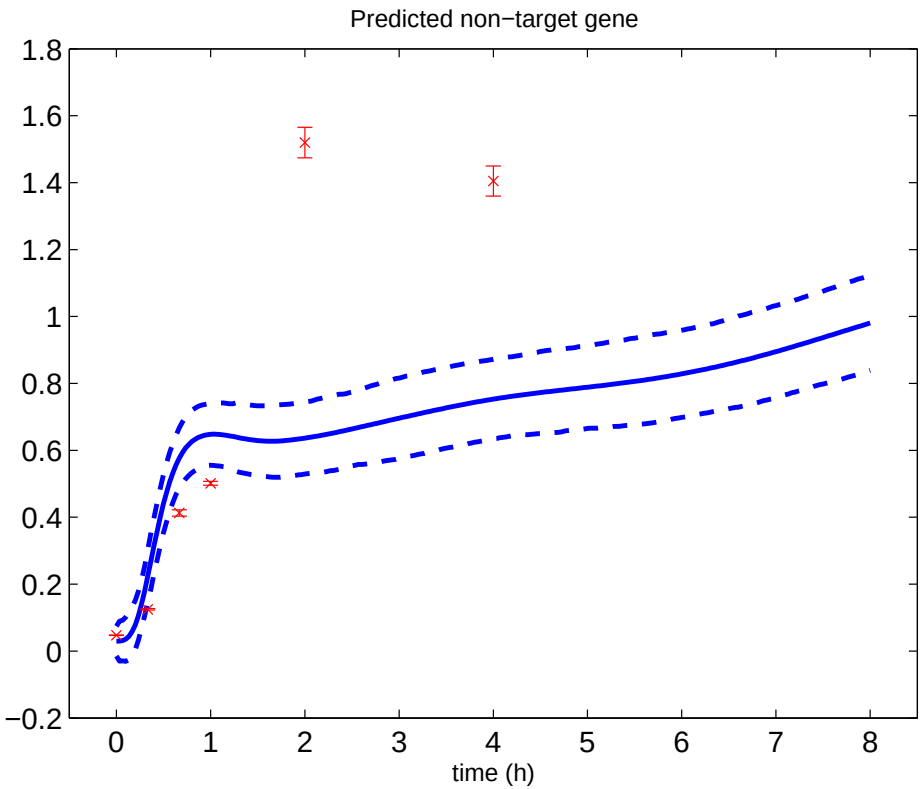




Elk-1 target selection

Fitted model used to rank potential targets of Elk-1





PNAS

Model-based method for transcription factor target identification with limited data

Antti Honkela[a,1], Charles Girardot[b], E. Hilary Gustafson[b], Ya-Hsin Liu[b], Eileen E. M. Furlong[b], Neil D. Lawrence[c,1], and Magnus Rattray[c,1]

[a]Department of Information and Computer Science, Aalto University School of Science and Technology, Helsinki, Finland; [b]Genome Biology U
European Molecular Biology Laboratory, Heidelberg, Germany; and [c]School of Computer Science, University of Manchester, Manchester, Unite

Edited by David Baker, University of Washington, Seattle, WA, and approved March 3, 2010 (received for review December 10, 2009)

We present a computational method for identifying potential targets of a transcription factor (TF) using wild-type gene expression time series data. For each putative target gene we fit a simple differential equation model of transcriptional regulation, and the

used for genome-wide scoring of putative target gen
is required to apply our method is wild-type time seri
lected over a period where TF activity is changing. Ou
allows for complementary evidence from expression

Cascaded Differential Equations

(Honkela et al., 2010)

- 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.
- Build a simple cascaded differential equation to model this.

Covariance for Translation/Transcription Model

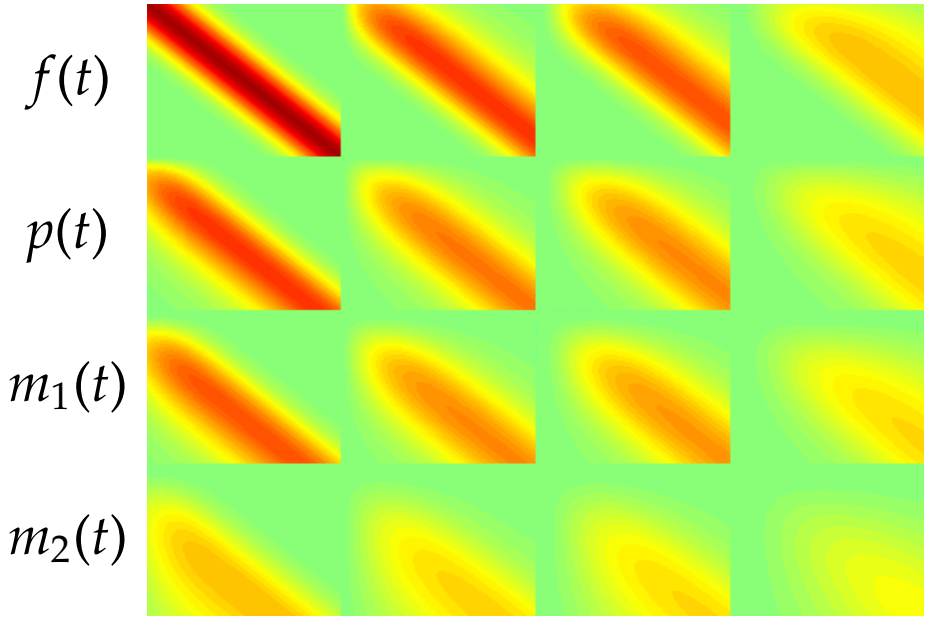
RBF covariance function for $f(t)$

$$p(t) = \sigma \exp(-\delta t) \int_0^t f(u) \exp(\delta u)\, \mathrm{d}u$$

$$m_i(t) = \frac{b_i}{d_i} + s_i \exp(-d_i t) \int_0^t p(u) \exp(d_i u)\, \mathrm{d}u.$$

- Joint distribution for $m_1(t)$, $m_2(t)$, $p(t)$ and $f(t)$.
- Here:

δ	d_1	s_1	d_2	s_2
1	5	5	0.5	0.5



Twist Results

- Use mRNA of Twist as driving input.
- For each gene build a cascade model that forces Twist to be the only TF.
- Compare fit of this model to a baseline (*e.g.* similar model but sensitivity zero).
- Rank according to the likelihood above the baseline.
- Compare with correlation, knockouts and time series network identification (TSNI) (Della Gatta et al., 2008).

Results for Twi using the Cascade model

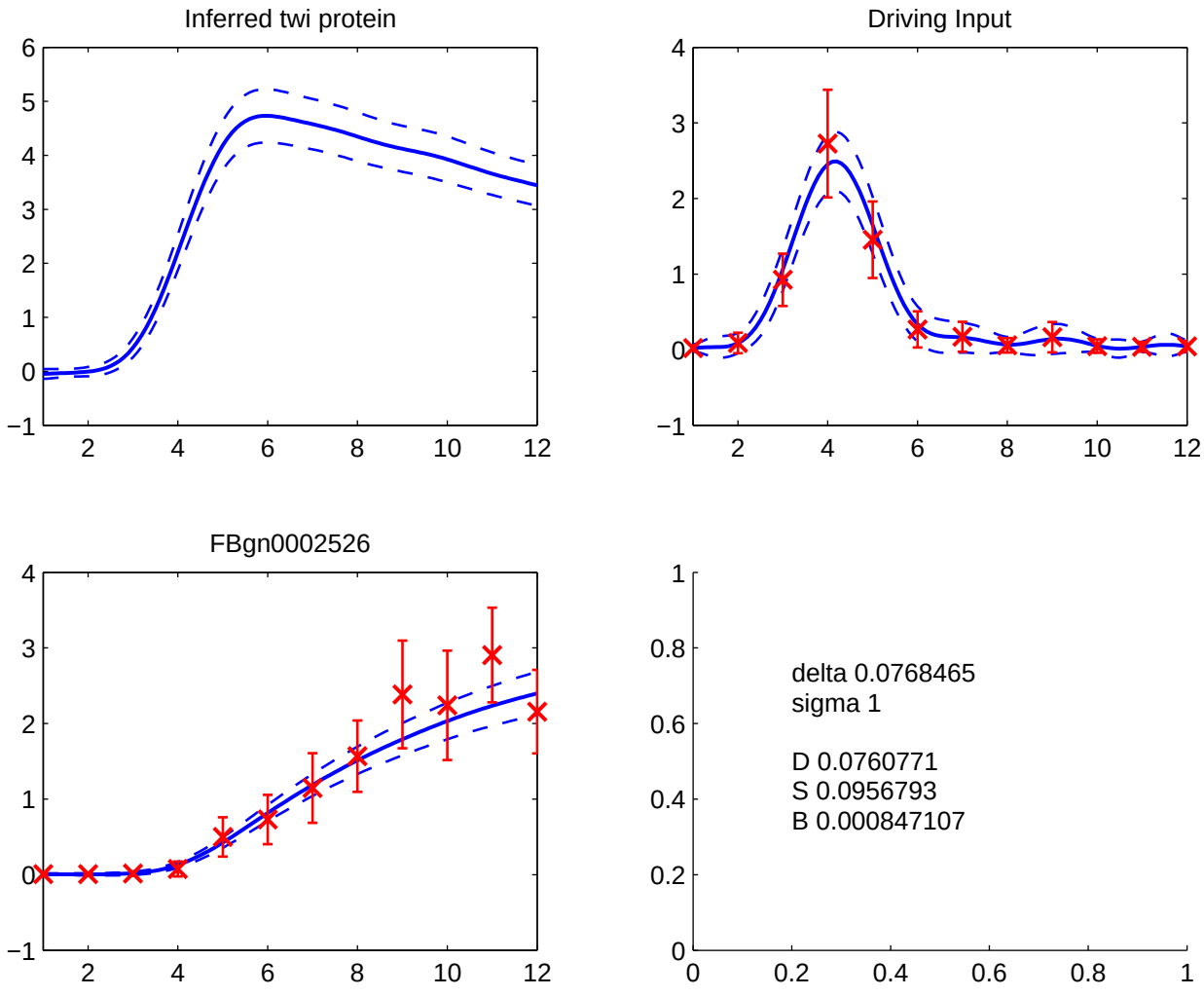


Figure : Model for flybase gene identity FBgn0002526.

Results for Twi using the Cascade model

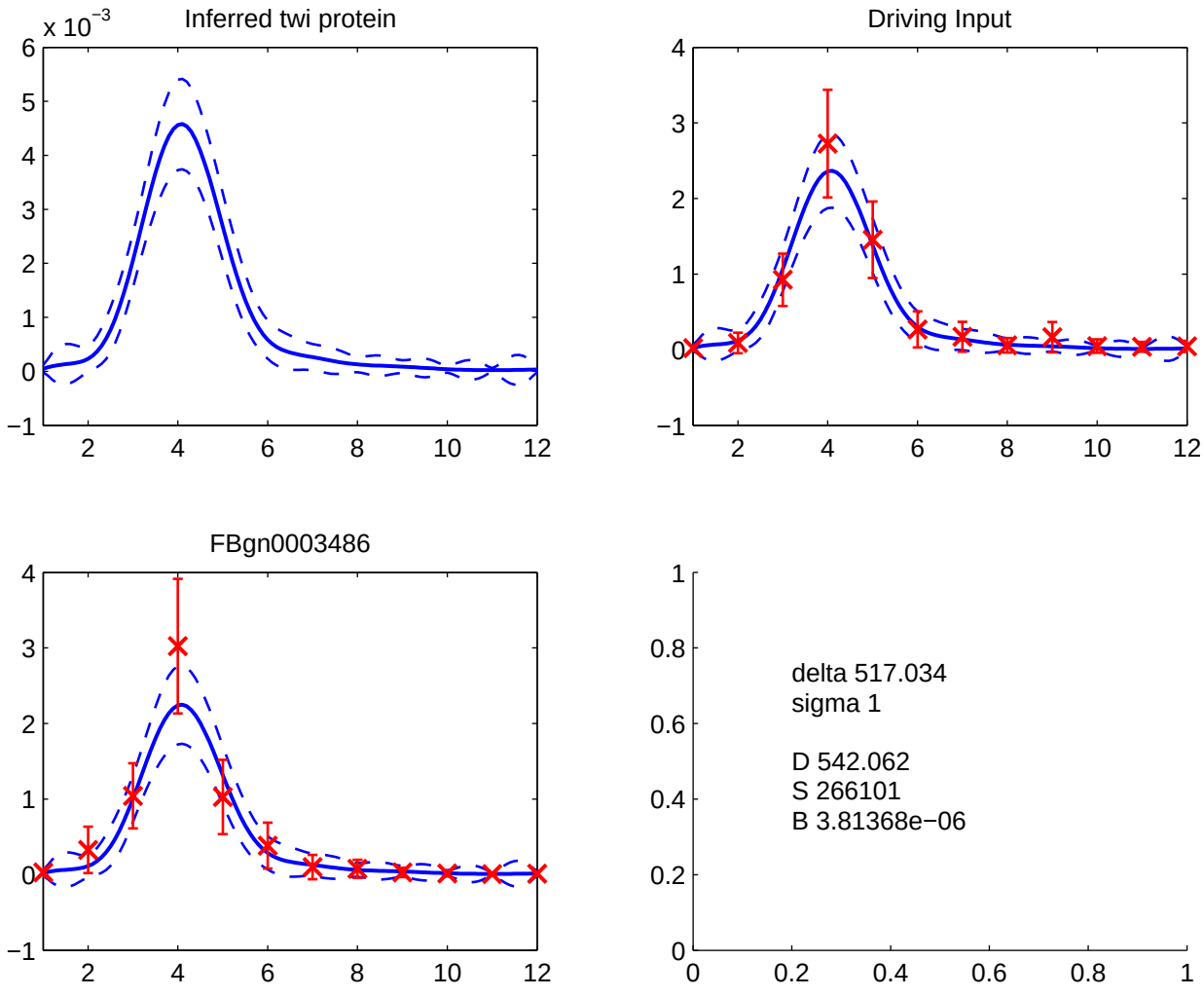


Figure : Model for flybase gene identity FBgn0003486.

Results for Twi using the Cascade model

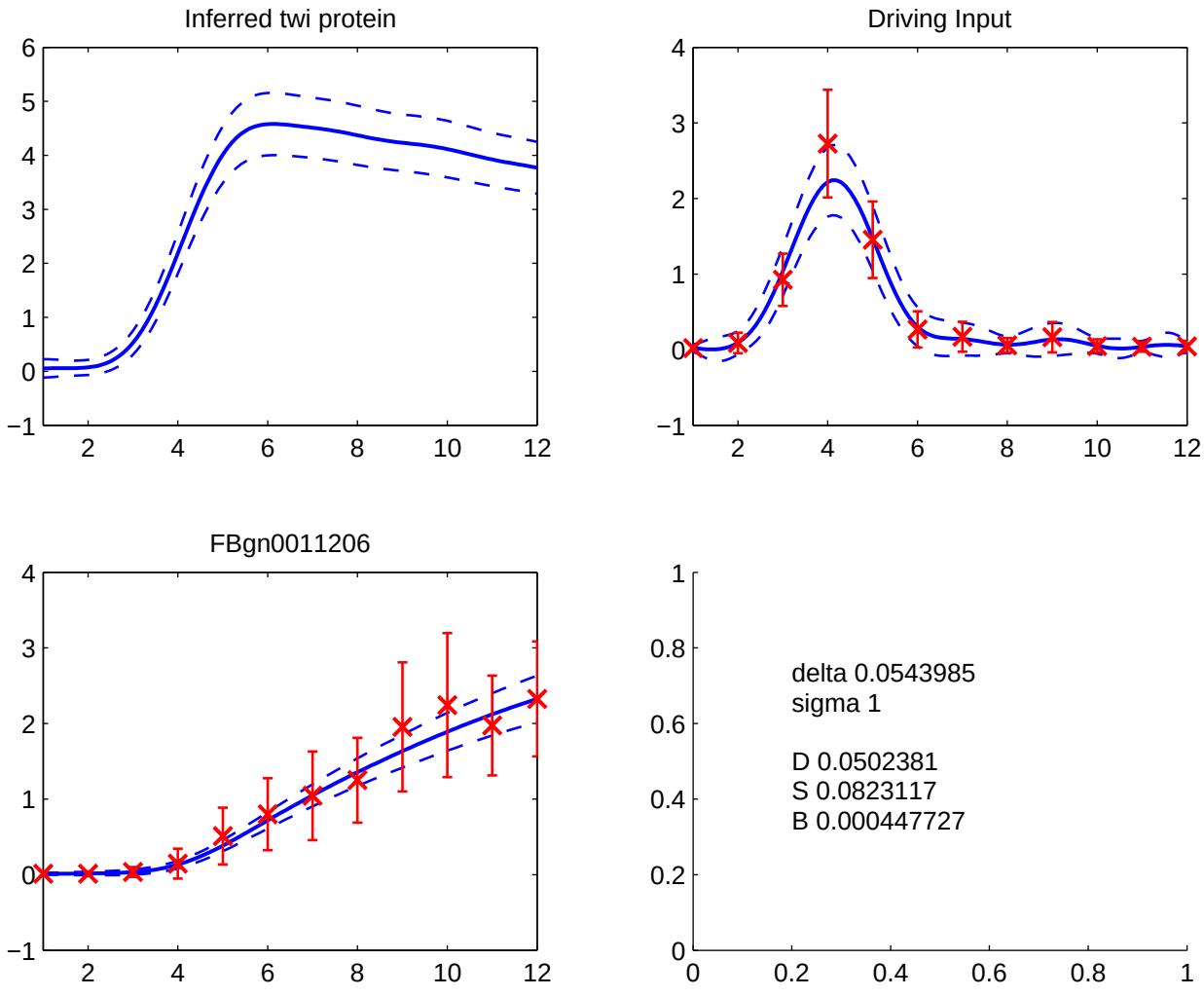


Figure : Model for flybase gene identity FBgn0011206.

Results for Twi using the Cascade model

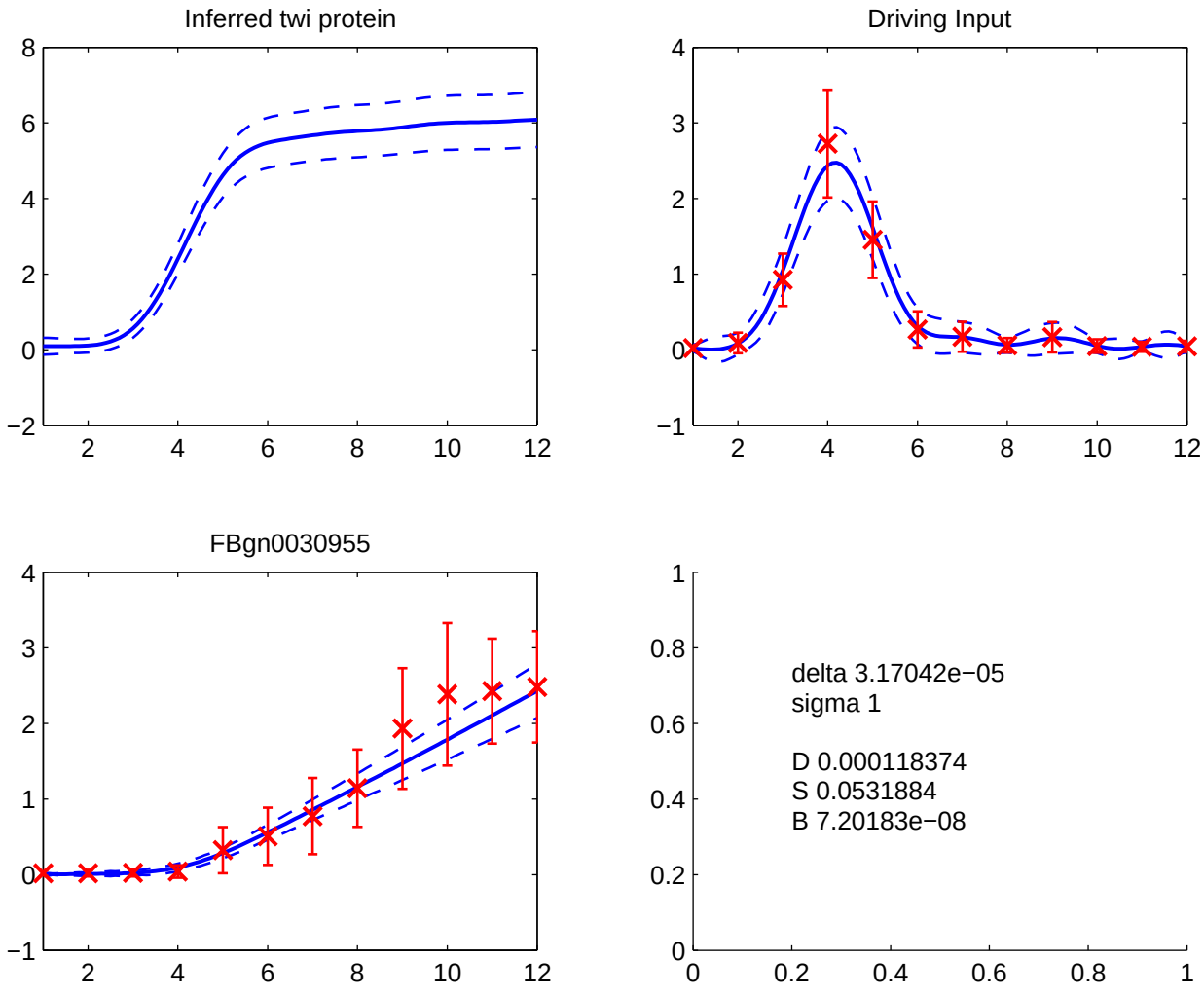


Figure : Model for flybase gene identity FBgn00309055.

Results for Twi using the Cascade model

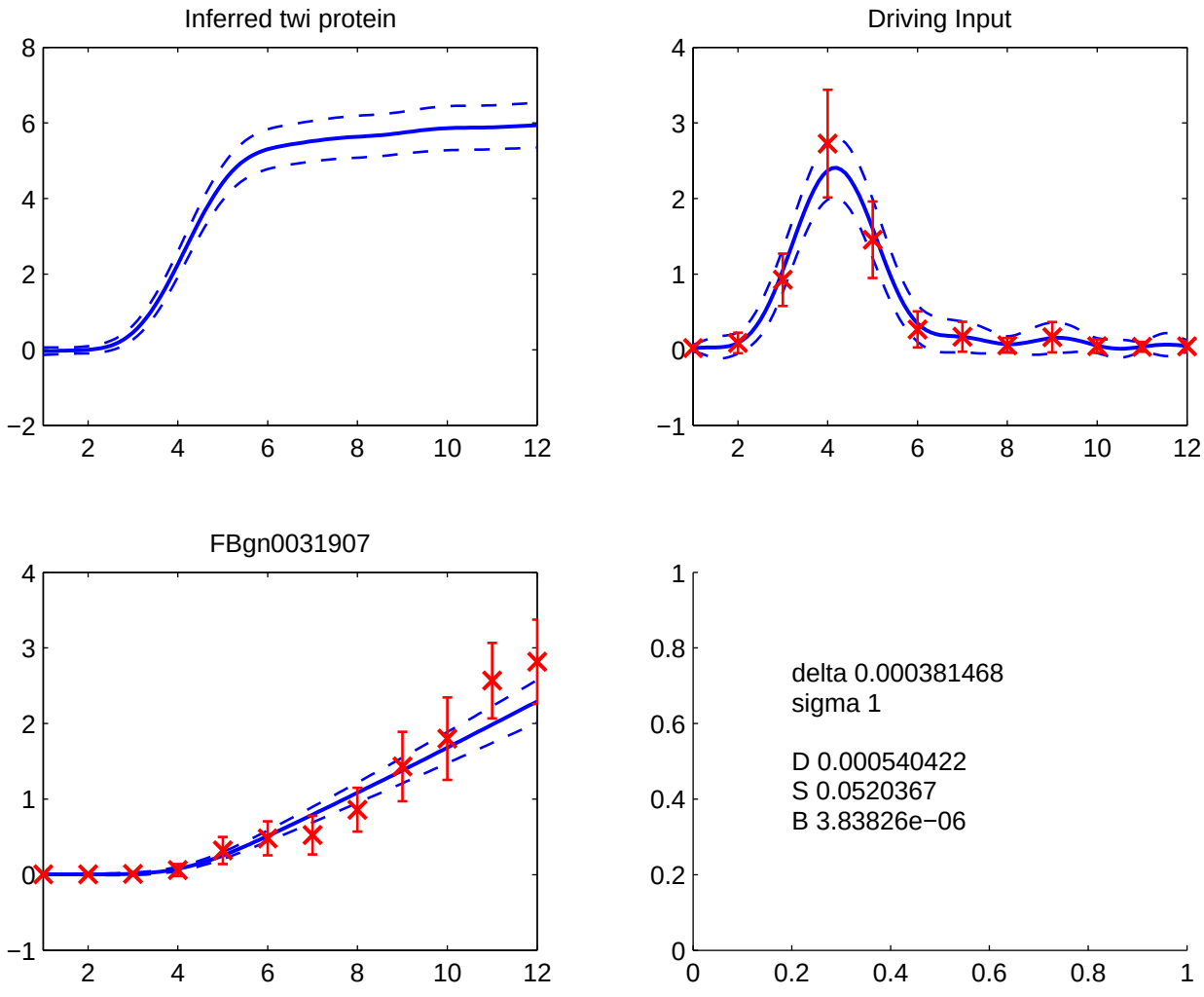


Figure : Model for flybase gene identity FBgn0031907.

Results for Twi using the Cascade model

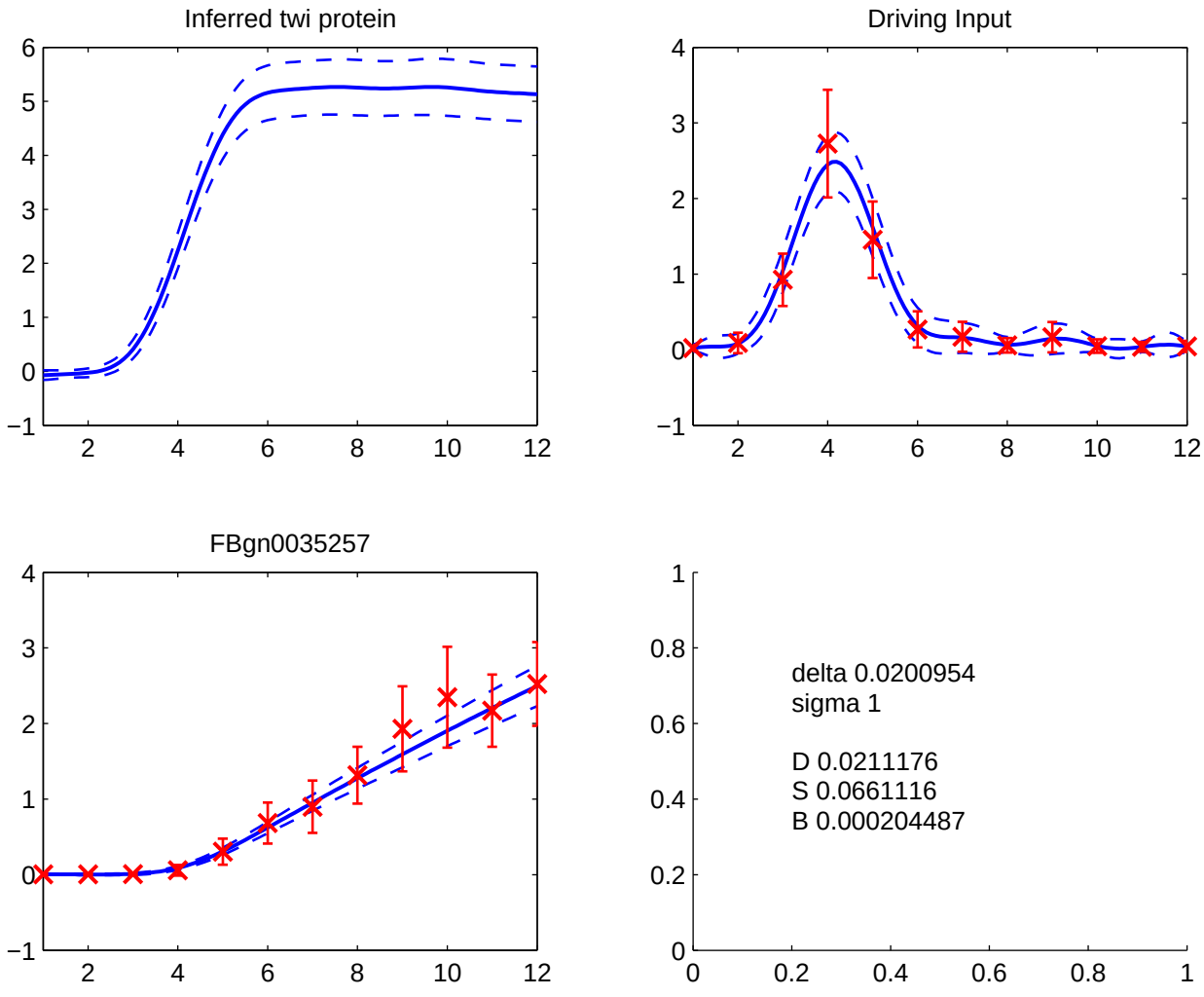


Figure : Model for flybase gene identity FBgn0035257.

Results for Twi using the Cascade model

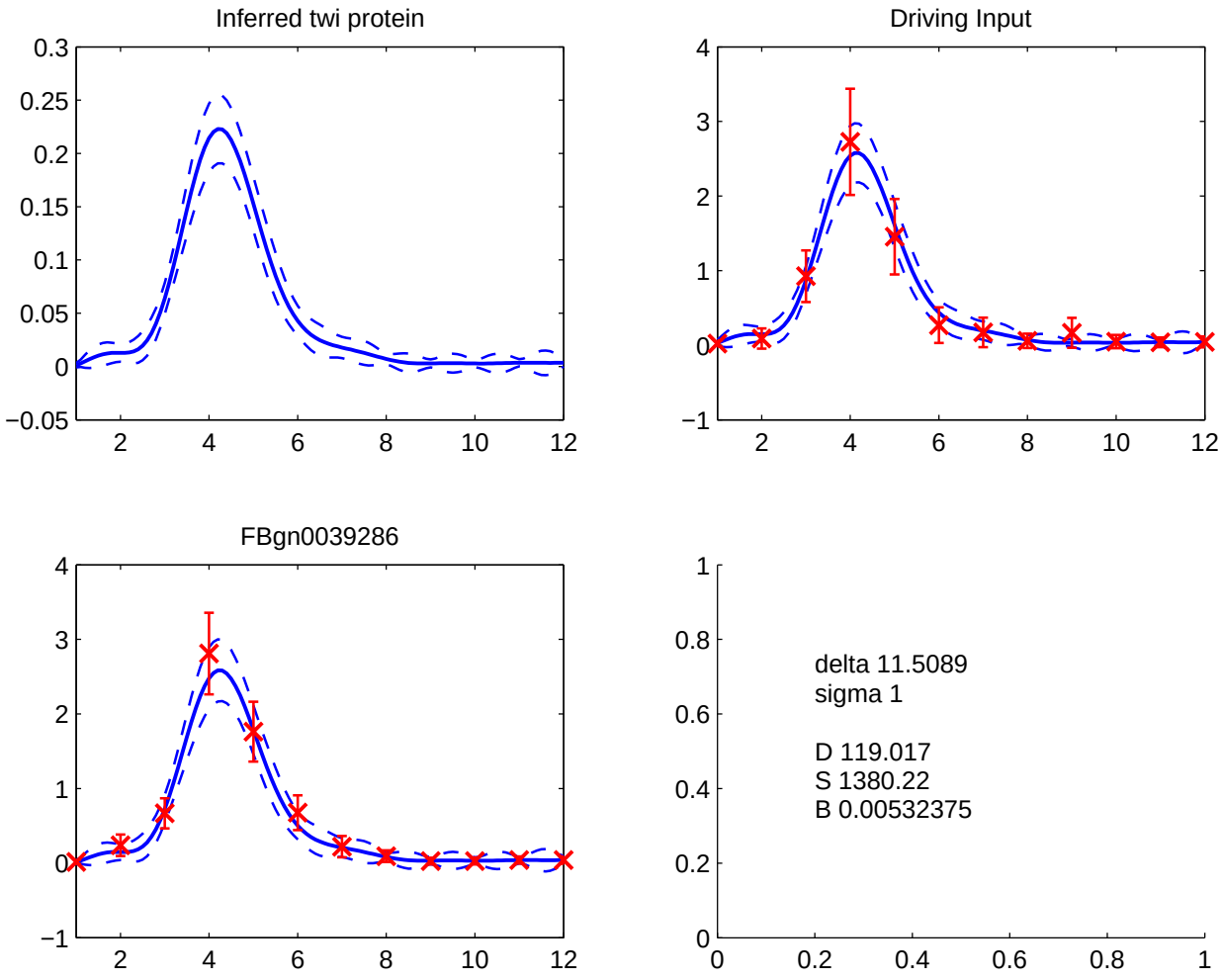
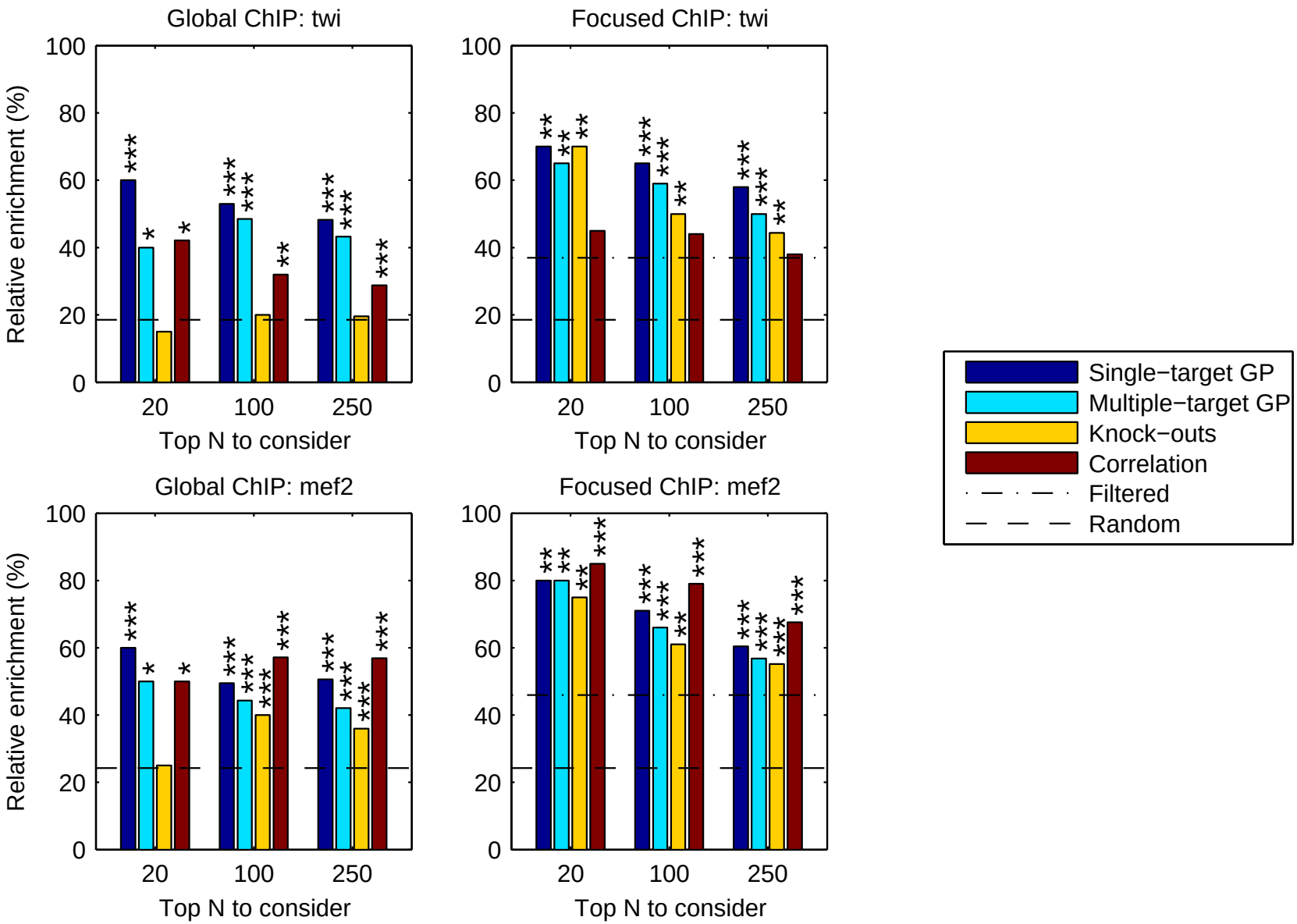


Figure : Model for flybase gene identity FBgn0039286.

Evaluation methods

- Evaluate the ranking methods by taking a number of top-ranked targets and record the number of "positives" (Zinzen et al., 2009):
  - targets with ChIP-chip binding sites within 2 kb of gene
  - (targets differentially expressed in TF knock-outs)
- Compare against
  - Ranking by correlation of expression profiles
  - Ranking by q-value of differential expression in knock-outs
- Optionally focus on genes with annotated expression in tissues of interest

Results



'***': $p < 0.001$, '**': $p < 0.01$, '*': $p < 0.05$

Summary

- Cascade models allow genomewide analysis of potential targets given only expression data.
- Once a set of potential candidate targets have been identified, they can be modelled in a more complex manner.
- We don't have ground truth, but evidence indicates that the approach *can* perform as well as knockouts.

Partial Differential Equations and Latent Forces

Mauricio Alvarez

- Can extend the concept to latent functions in PDEs.
- Jura data: concentrations of heavy metal pollutants from the Swiss Jura.
- Consider a latent function that represents how the pollutants were originally laid down (initial condition).
- Assume pollutants diffuse at different rates resulting in the concentrations observed in the data set.

$$\frac{\partial x_q(\mathbf{x},t)}{\partial t} = \sum_{j=1}^{d} \kappa_q \frac{\partial^2 x_q(\mathbf{x},t)}{\partial x_j^2},$$

- Latent function $f_r(\mathbf{x})$ represents the concentration of pollutants at time zero (i.e. the system's initial condition).

Solution to the PDE

Mauricio Alvarez

- The solution to the system (Polyanin, 2002) is then given by

$$x_q(\mathbf{x}, t) = \sum_{r=1}^{R} S_{rq} \int_{\mathbb{R}^d} f_r(\mathbf{x}') G_q(\mathbf{x}, \mathbf{x}', t) \mathrm{d}\mathbf{x}'$$

where $G_q(\mathbf{x}, \mathbf{x}', t)$ is the Green's function given as

$$G_q(\mathbf{x}, \mathbf{x}', t) = \frac{1}{2^d \pi^{d/2} T_q^{d/2}} \exp\left[-\sum_{j=1}^{d} \frac{(x_j - x_j')^2}{4T_q} \right],$$

with $T_q = \kappa_q t$.

Covariance Function

Mauricio Alvarez

- For latent function given by a GP with the RBF covariance function this is tractable.

$$k_{x_p x_q}(\mathbf{x}, \mathbf{x}', t) = \sum_{r=1}^{R} \frac{S_{rp} S_{rq} |\mathbf{L}_r|^{1/2}}{|\mathbf{L}_{rp} + \mathbf{L}_{rq} + \mathbf{L}_r|^{1/2}}$$
$$\times \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \left(\mathbf{L}_{rp} + \mathbf{L}_{rq} + \mathbf{L}_r\right)^{-1} (\mathbf{x} - \mathbf{x}')\right],$$

where $\mathbf{L}_{rp}, \mathbf{L}_{rq}$ and $\mathbf{L}_r$ are diagonal isotropic matrices with entries $2\kappa_p t$, $2\kappa_q t$ and $1/\ell_r^2$ respectively. The covariance function between the output and latent functions is given by

$$k_{x_q f_r}(\mathbf{x}, \mathbf{x}', t) = \frac{S_{rq} |\mathbf{L}_r|^{1/2}}{|\mathbf{L}_{rq} + \mathbf{L}_r|^{1/2}}$$
$$\times \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \left(\mathbf{L}_{rq} + \mathbf{L}_r\right)^{-1} (\mathbf{x} - \mathbf{x}')\right].$$

Prediction of Metal Concentrations

Mauricio Alvarez

- Replicate experiments in (Goovaerts, 1997, pp. 248,249):
  - *Primary variable* (Cd, Cu, Pb, Co) predicted in conjunction with *secondary variables* (Ni and Zn for Cd; Pb, Ni, and Zn for Cu; Cu, Ni, and Zn for Pb; Ni and Zn for Co).[2]
- Condition on the secondary variables to improve prediction for primary variables.
- Compare results for the diffusion kernel with independent GPs and "ordinary co-kriging" (Goovaerts, 1997, pp. 248,249).

Jura Results

Table : Mean absolute error and standard deviation for ten repetitions of the experiment for the Jura dataset. IGPs stands for independent GPs, GPDK stands for GP diffusion kernel, OCK for ordinary co-kriging. For the Gaussian process with diffusion kernel, we learn the diffusion coefficients and the length-scale of the covariance of the latent function.

Metals	IGPs	GPDK	OCK
Cd	0.5823±0.0133	0.4505±0.0126	0.5
Cu	15.9357±0.0907	7.1677±0.2266	7.8
Pb	22.9141±0.6076	10.1097±0.2842	10.7
Co	2.0735±0.1070	1.7546±0.0895	1.5

Convolutions and Computational Complexity

Mauricio Alvarez

- Solutions to these differential equations is normally as a convolution.

$$x_i(t) = \int f(u)\, k_i(u-t)\, \mathrm{d}u + h_i(t)$$

$$x_i(t) = \int_0^t f(u)\, g_i(u)\, \mathrm{d}u + h_i(t)$$

- Convolution Processes (Higdon, 2002; Boyle and Frean, 2005).
- Convolutions lead to $N \times d$ size covariance matrices $O\left(N^3 d^3\right)$ complexity, $O\left(N^2 d^2\right)$ storage.
- Model is conditionally independent over $\{x_i(t)\}_{i=1}^d$ given $f(t)$.

Independence Assumption

Mauricio Alvarez

- Can assume conditional independence given given $\{f(t_i)\}_{i=1}^{k}$. (Álvarez and Lawrence, 2009)
  - Result is very similar to PITC approximation (Quiñonero Candela and Rasmussen, 2005).
  - Reduces to $O\left(N^3dk^2\right)$ complexity, $O\left(N^2dk\right)$ storage.
  - Can also do a FITC style approximation (Snelson and Ghahramani, 2006).
  - Reduces to $O\left(Ndk^2\right)$ complexity, $O(Ndk)$ storage.

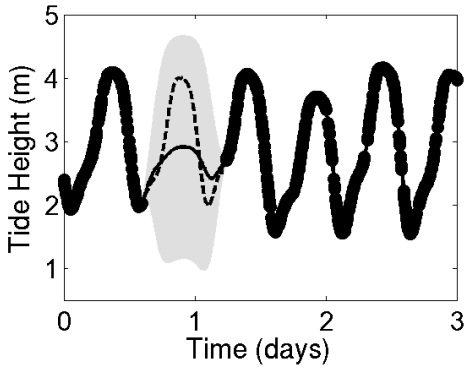
Tide Sensor Network

Mauricio Alvarez

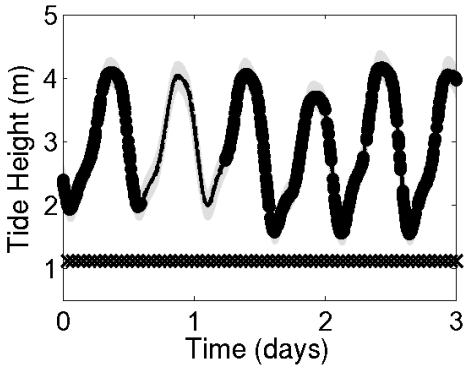
- Network of tide height sensors in the solent — tide heights are correlated.
- Data kindly provided by Alex Rogers (see Osborne et al., 2008).
- $d = 3$ and $N = 1000$ of the 4320 for the training set.
- Simulate sensor failure by knocking out onse sensor for a given time.
- For the other two sensors we used all 1000 training observations.
- Take $k = 100$.

Tide Height Results

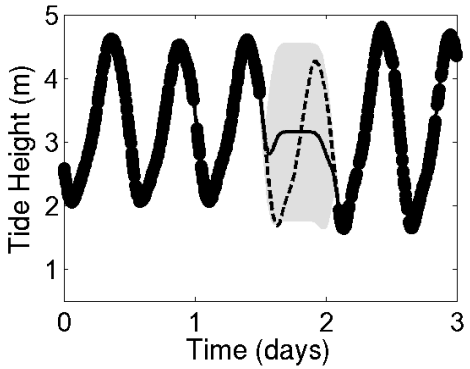
Mauricio Alvarez



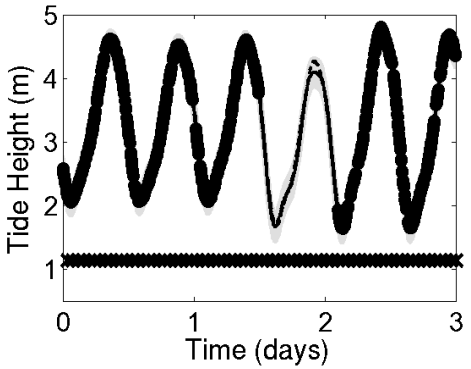
(a) Bramblemet Independent



(b) Bramblemet PITC



(c) Cambermet Independent



(d) Cambermet PITC

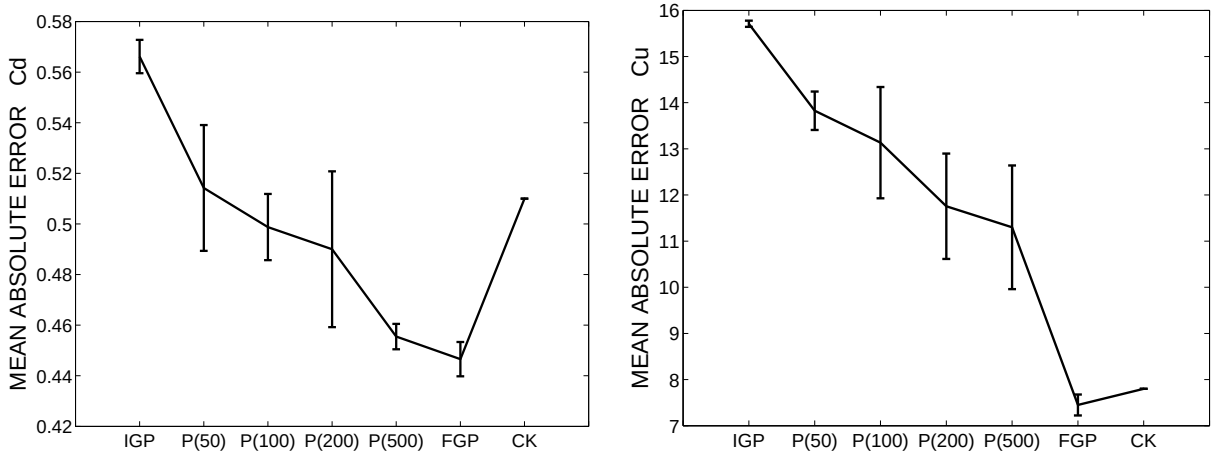
Cokriging Jura

Mauricio Alvarez

- Jura dataset — concentrations of several heavy metals (Atteia et al., 1994).
- Prediction 259 data, validation 100 data points.
- Predict *primary variables* (cadmium and copper) at prediction locations in conjunction with some *secondary variables* (nickel and zinc for cadmium; lead, nickel and zinc for copper) (Goovaerts, 1997, p. 248,249).

Swiss Jura Results

Mauricio Alvarez



(a) Cadmium (b) Copper

Figure : Mean absolute error. IGP stands for independent GP, P(M) stands for PITC with M inducing values, FGP stands for full GP and CK stands for ordinary co-kriging.

MAP-Laplace Approximation

Laplace's method: approximate posterior mode as Gaussian

$$p(\mathbf{p} \mid m) = N\left(\hat{\mathbf{p}}, \mathbf{A}^{-1}\right) \propto \exp\left(-\frac{1}{2}(\mathbf{p}-\hat{\mathbf{p}})^{\top}\mathbf{A}(\mathbf{p}-\hat{\mathbf{p}})\right)$$

where $\hat{\mathbf{p}} = \text{argmax} p(\mathbf{p} \mid \mathbf{m})$ and $\mathbf{A} = -\nabla\nabla \log p(\mathbf{p} \mid \mathbf{m})\,|_{\mathbf{p}=\hat{\mathbf{p}}}$ is the Hessian of the negative posterior at that point. To obtain $\hat{\mathbf{p}}$ and

$\mathbf{A}$, we define the following function $\psi(\mathbf{p})$ as:

$$\log p(\mathbf{p}|\mathbf{m}) \propto \psi(\mathbf{p}) = \log p(\mathbf{m} \mid \mathbf{p}) + \log p(\mathbf{p})$$

MAP-Laplace Approximation

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

$$\log p\left(\mathbf{p}\right) = -\frac{1}{2}\mathbf{p}^{\top}\mathbf{K}^{-1}\mathbf{p} - \frac{1}{2}\log|\mathbf{K}| - \frac{n}{2}\log 2\pi$$

where $\mathbf{K}$ is the covariance matrix of $p(t)$. Hence,

$$\nabla\psi(\mathbf{p}) = \nabla\log p(\mathbf{m}|\mathbf{p}) - \mathbf{K}^{-1}\mathbf{p}$$
$$\nabla\nabla\psi(\mathbf{p}) = \nabla\nabla\log p(\mathbf{m}|\mathbf{p}) - \mathbf{K}^{-1} = -\mathbf{W} - \mathbf{K}^{-1}$$

Estimation of $\psi(\mathbf{p})$

Newton's method is applied to find the maximum of $\psi(\mathbf{p})$ as

$$\begin{aligned}\mathbf{p}^{new} &= \mathbf{p} - (\nabla\nabla\psi(\mathbf{p}))^{-1}\nabla\psi(\mathbf{p}) \\ &= (\mathbf{W} + \mathbf{K}^{-1})^{-1}\left(\mathbf{W}\mathbf{p} - \nabla \log p(\mathbf{m}|\mathbf{p})\right)\end{aligned}$$

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

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

Model Parameter Estimation

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

$$p\left(\mathbf{m}|\boldsymbol{\theta},\phi\right)=\int p\left(\mathbf{m}|\mathbf{p},\boldsymbol{\theta}\right)p\left(\mathbf{p}|\phi\right)\mathrm{d}p=\int\exp\left(\psi\left(\mathbf{p}\right)\right)\mathrm{d}p$$

Using Taylor expansion of $\psi(\mathbf{p})$,

$$\log p(\mathbf{m}|\boldsymbol{\theta},\phi)=\log p\left(\mathbf{m}|\hat{\mathbf{p}},\boldsymbol{\theta},\phi\right)-\frac{1}{2}\mathbf{p}^{\top}\mathbf{K}^{-1}\mathbf{p}-\frac{1}{2}\log|\mathbf{I}+\mathbf{K}\mathbf{W}|$$

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

$$\frac{\partial\log p\left(\mathbf{m}|\boldsymbol{\eta}\right)}{\partial\boldsymbol{\eta}}=\frac{\partial\log p\left(\mathbf{m}|\boldsymbol{\eta}\right)}{\partial\boldsymbol{\eta}}\Big|_{\text{explicit}}+\frac{\partial\log p\left(\mathbf{m}|\boldsymbol{\eta}\right)}{\partial\hat{\mathbf{p}}}\frac{\partial\hat{\mathbf{p}}}{\partial\boldsymbol{\eta}}$$

SOS Response

- DNA damage in bacteria 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.

Khanin et al. Model

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

$$m_i(t) = m_i^0 e^{-d_i t} + \frac{b_i}{d_i} + s_i e^{-d_i t} \int_0^t F(p(u)) e^{d_i u} \mathrm{d}u.$$

$$m_i(t) = m_i^0 e^{-d_i t} + \frac{b_i}{d_i} + s_i e^{-d_i t} \frac{1}{t_{j+1} - t_j} \sum_{j=0}^{N-2} F(\bar{p}_j) \left(e^{d_i t_{j+1}} - e^{d_i t_j} \right)$$

where $\bar{p}_j = \frac{(p(t_j) + p(t_{j+1}))}{2}$ on each subinterval $\left(t_j, t_j + 1\right)$, $j = 0, \ldots, N-2$. This is under the simplifying assumption that $p(t)$ is a piece-wise constant function on each subinterval $\left(t_j, t_j + 1\right)$. Repression model: $F(p(t)) = \frac{1}{\gamma + e^{p(t)}}$.

Khanin et al. Results

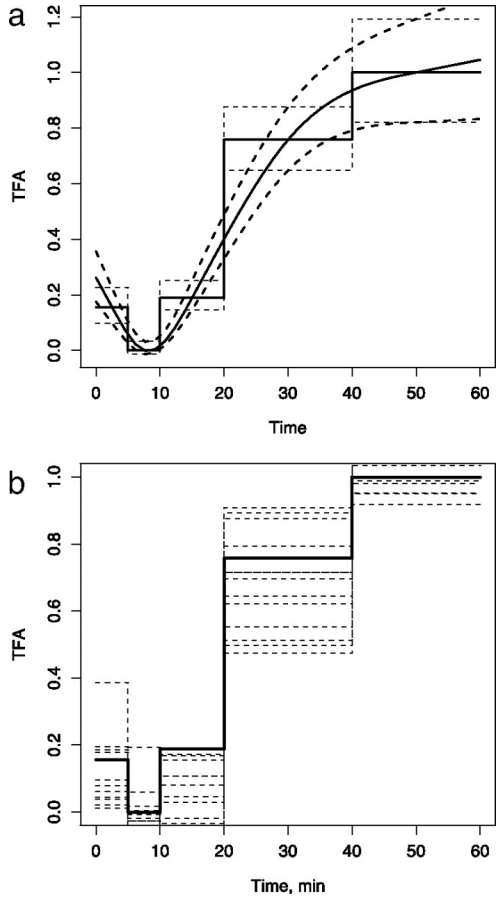


Figure : Fig. 2 from Khanin et al. (2006): Reconstructed activity level of master repressor LexA, following a UV dose of 40 J/m2.

Khanin et al. Results

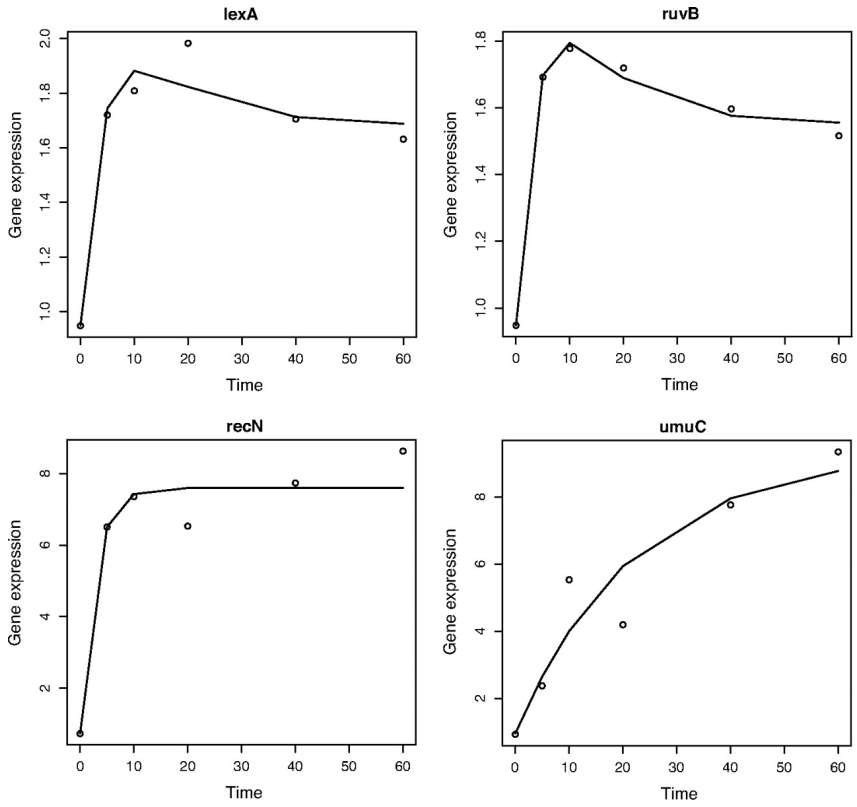


Figure : Fig. 3 from Khanin et al. (2006): Reconstructed profiles for four genes in the LexA SIM.

Repression Model

Pei Gao

- We can use the same model of repression,

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

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

$$m_j\left(t\right) = \alpha_j e^{-d_j t} + \frac{b_j}{d_j} + s_j \int_0^t e^{-d_j(t-u)} F_j(p\left(u\right))\mathrm{d}u$$

Results for the repressor LexA

Pei Gao

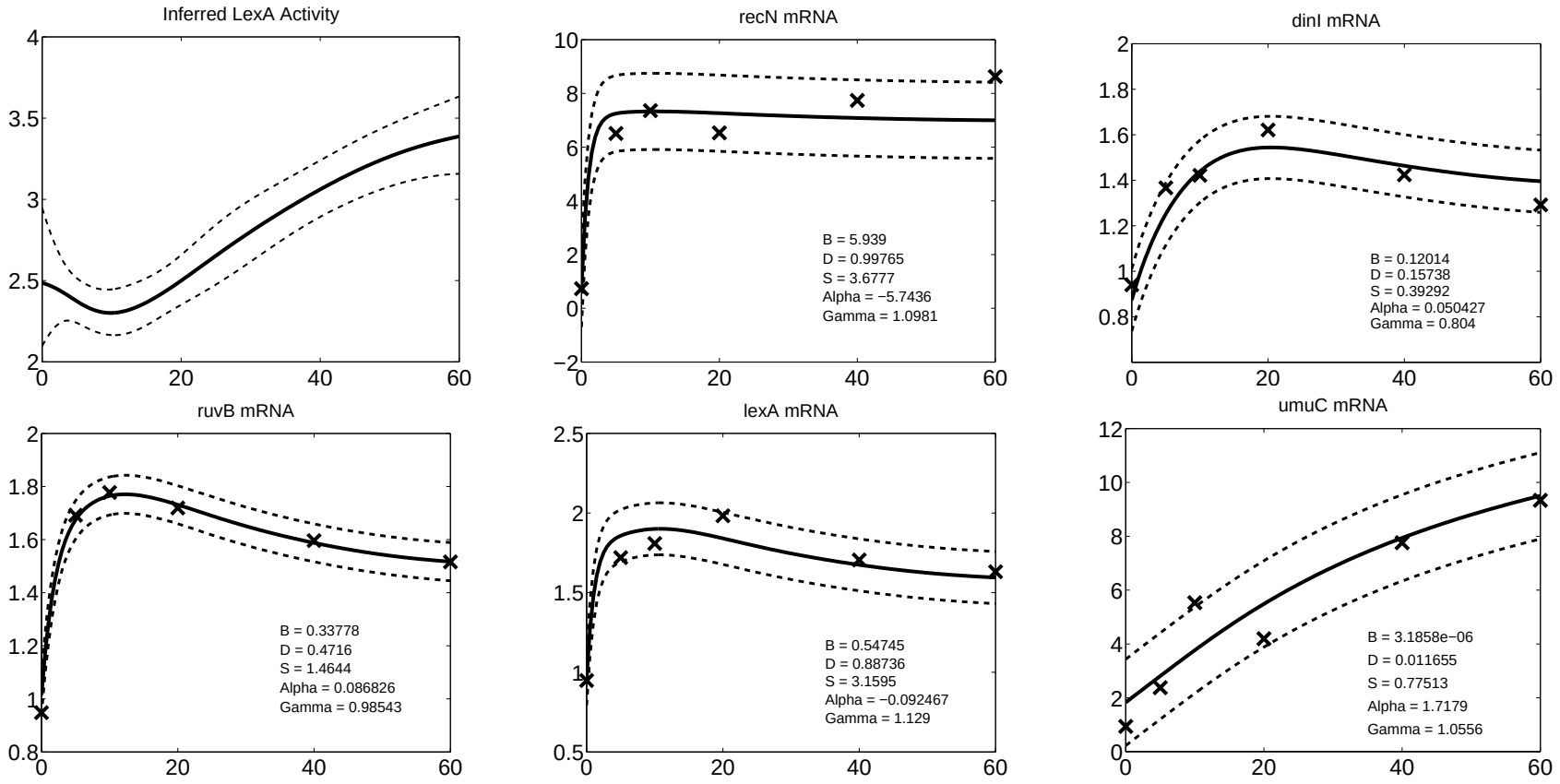


Figure : Our results using an MLP kernel. From Gao et al. (2008).

Use Samples to Represent Posterior

Michalis Titsias

- Sample in Gaussian processes

$$p\left(\mathbf{p}|\mathbf{m}\right) \propto p\left(\mathbf{m}|\mathbf{p}\right) p\left(\mathbf{p}\right)$$

- Likelihood relates GP to data through

$$m_j\left(t\right) = \alpha_j e^{-d_j t} + \frac{b_j}{d_j} + s_j \int_0^t e^{-d_j(t-u)} F_j(p\left(u\right))\mathrm{d}u$$

- We use *control points* for fast sampling.

MCMC for Non Linear Response

The Metropolis-Hastings algorithm

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

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

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

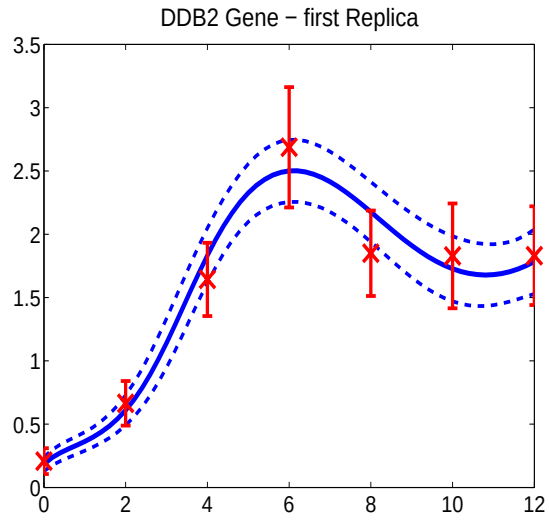
p53 System Again

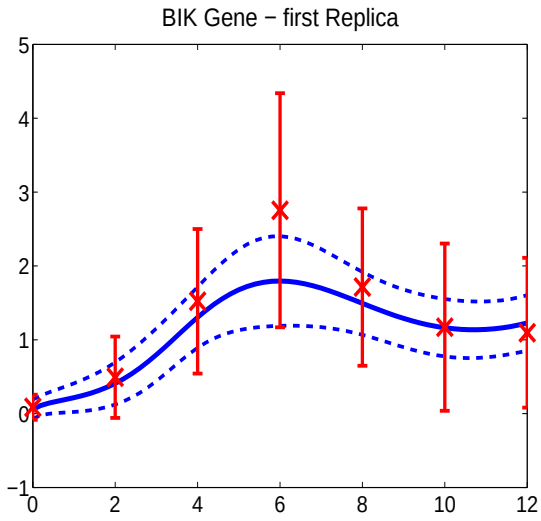
- One transcription factor (p53) that acts as an activator. We consider the Michaelis-Menten kinetic equation

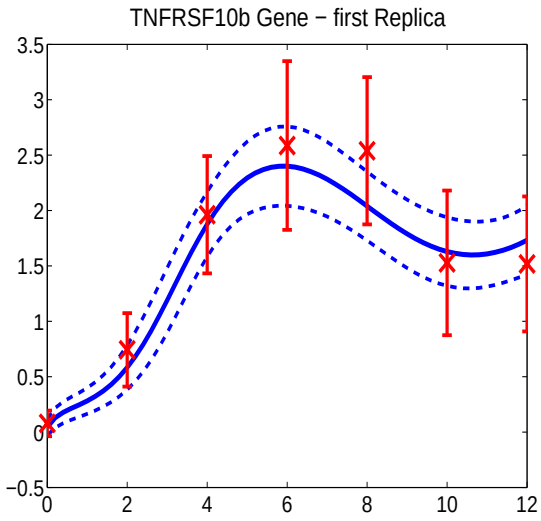
$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j \frac{\exp(p(t))}{\exp(p(t)) + \gamma_j} - d_j m_j(t)$$

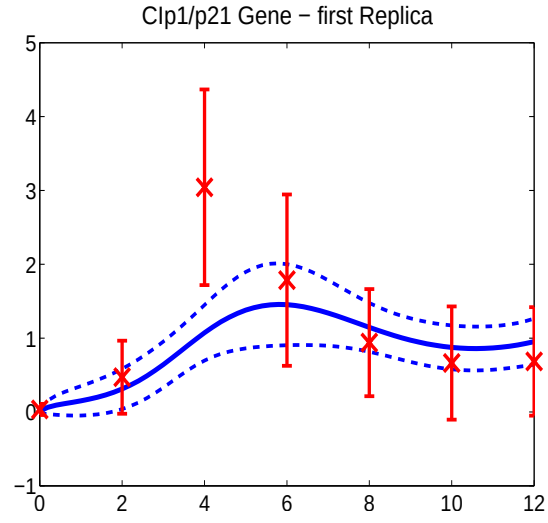
- We have 5 genes
- Gene expressions are available for $T = 7$ times and there are 3 replicas of the time series data
- TF ($\mathbf{p}$) is discretized using 121 points
- MCMC details:
  - 7 control points are used (placed in a equally spaced grid)
  - Running time 4/5 hours for 2 million sampling iterations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

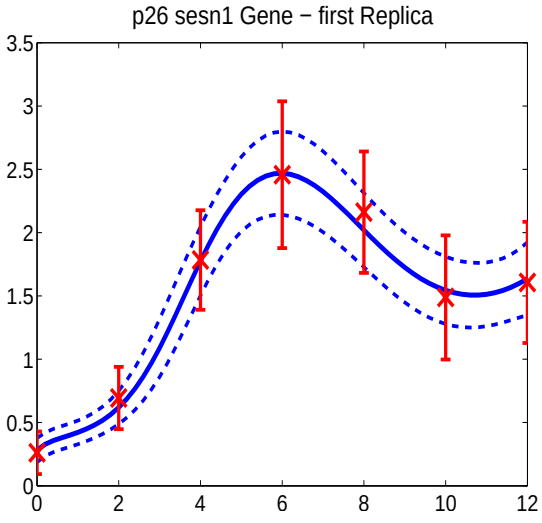
Data used by Barenco et al. (2006): Predicted gene expressions for the 1st replica



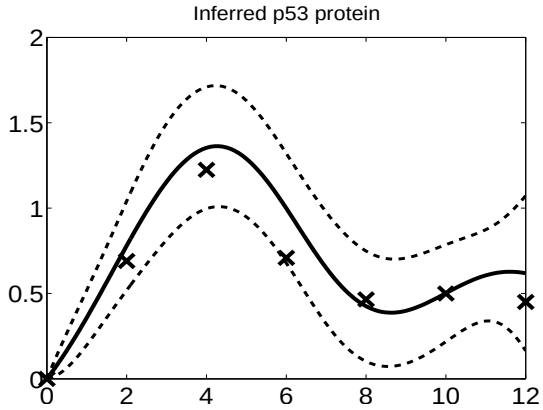


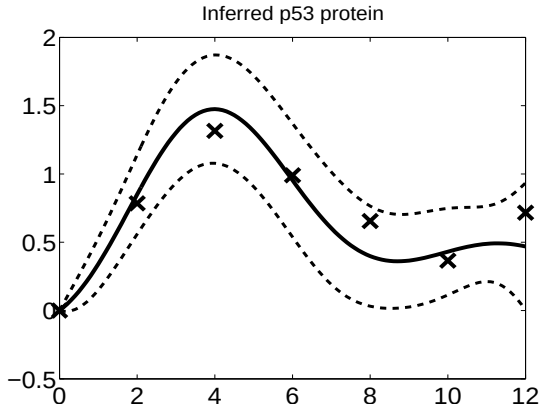


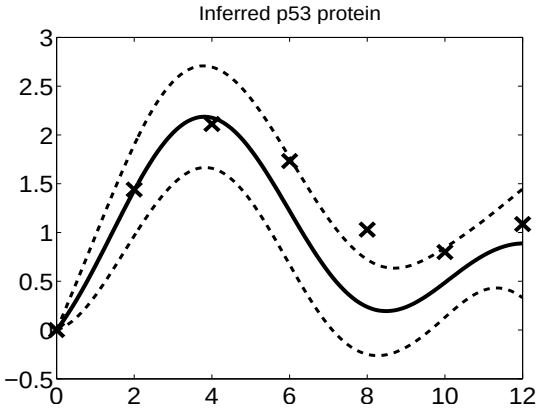




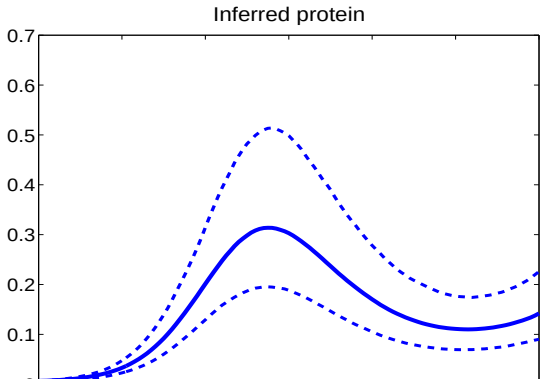
Data used by Barenco et al. (2006): Protein concentrations

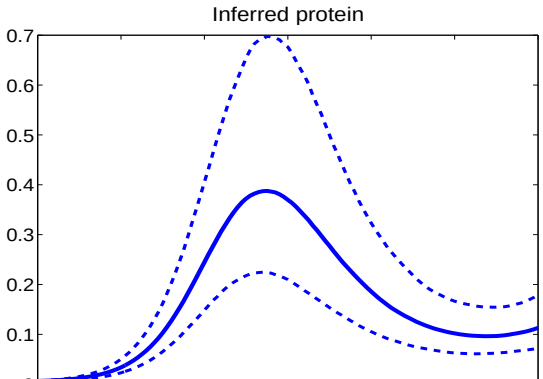


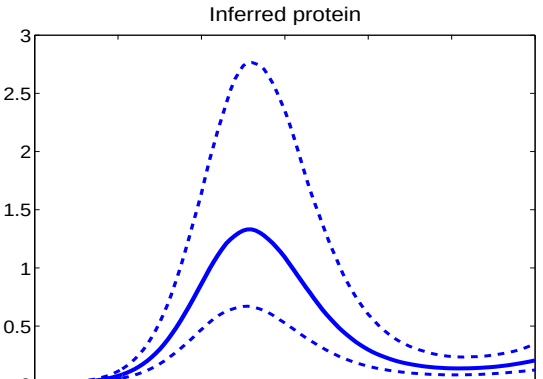




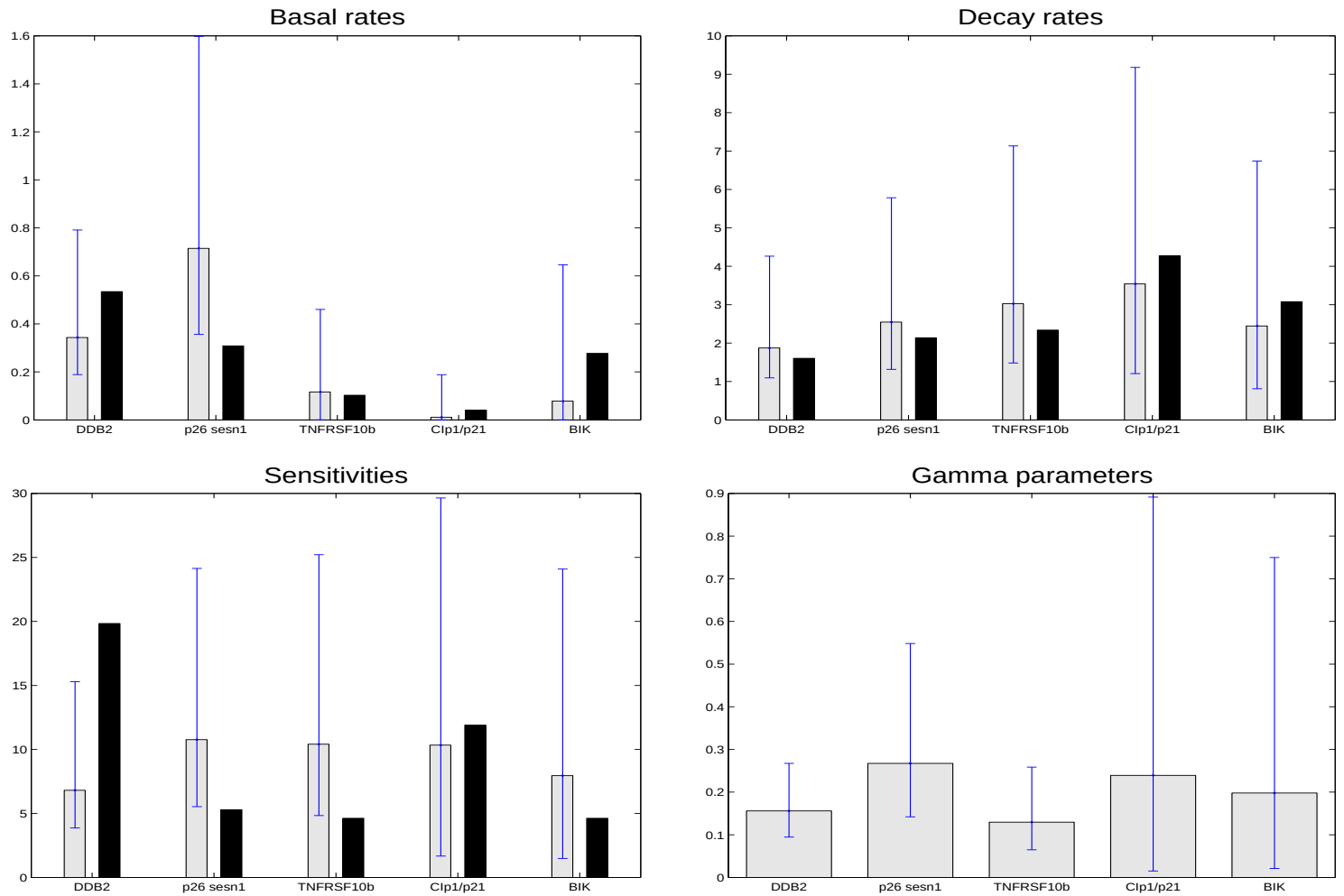
Linear model (Barenco et al. predictions are shown as crosses)







p53 Data Kinetic parameters



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

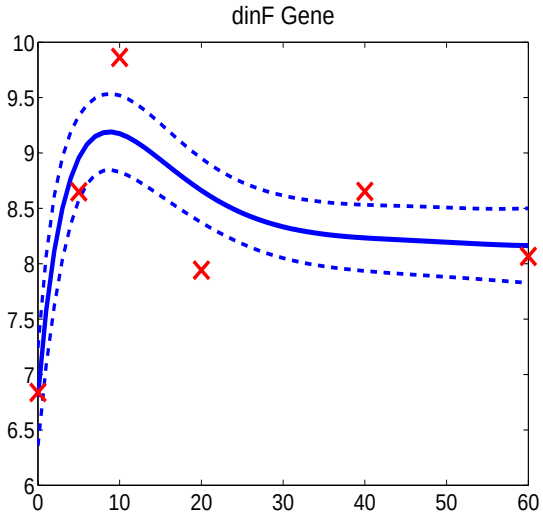
Results on SOS System

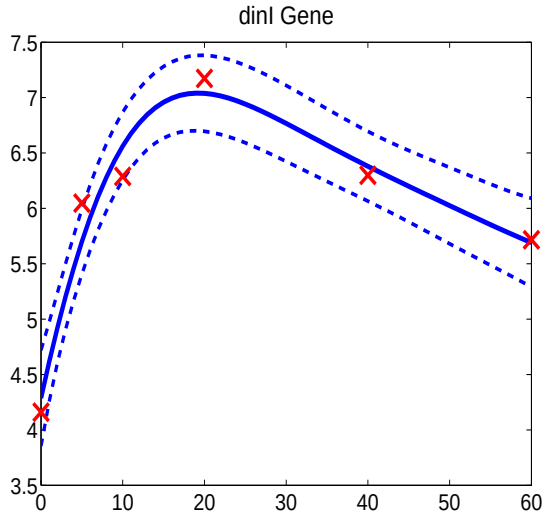
- Again consider the Michaelis-Menten kinetic equation

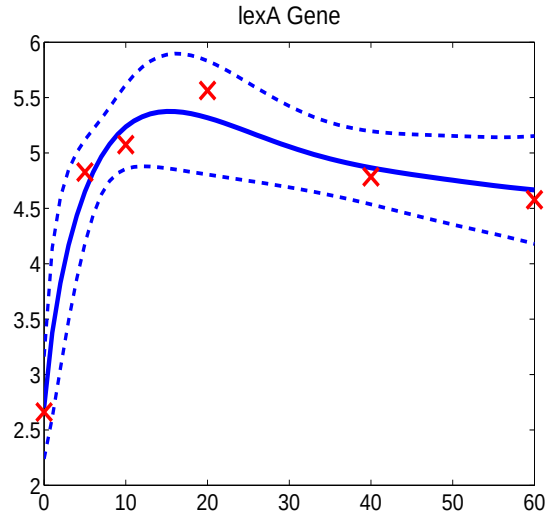
$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = b_j + s_j \frac{1}{\exp(p(t)) + \gamma_j} - d_j m_j(t)$$

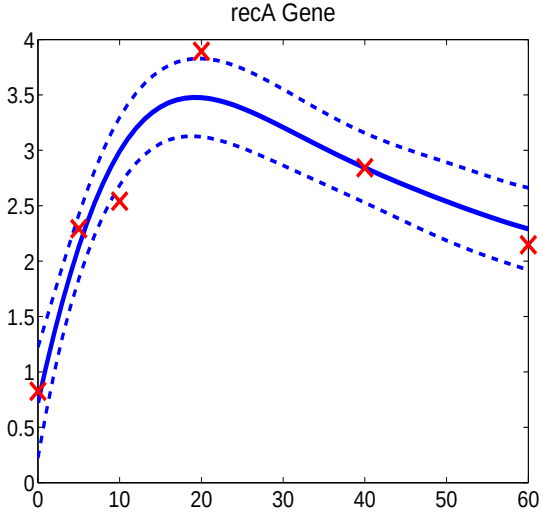
- We have 14 genes (5 kinetic parameters each)
- Gene expressions are available for $T = 6$ time slots
- TF ($\mathbf{p}$) is discretized using 121 points
- MCMC details:
  - 6 control points are used (placed in a equally spaced grid)
  - Running time was 5 hours for 2 million sampling iterations plus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

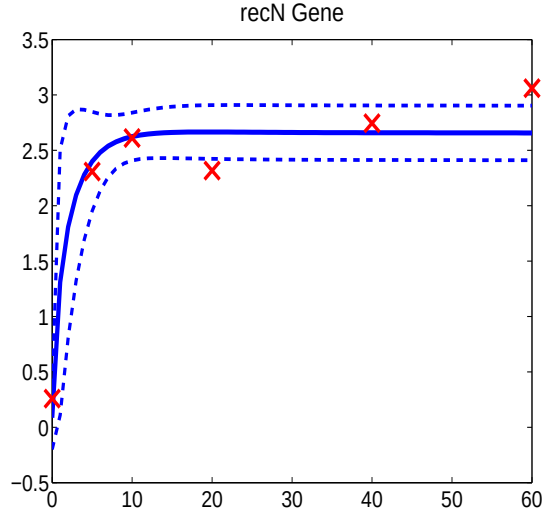
Results in E.coli data: Predicted gene expressions

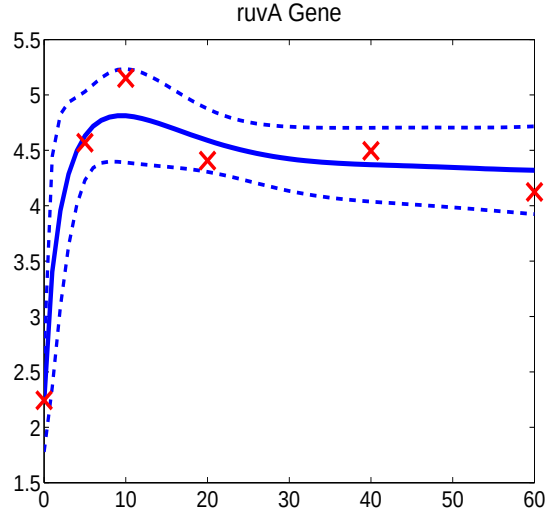




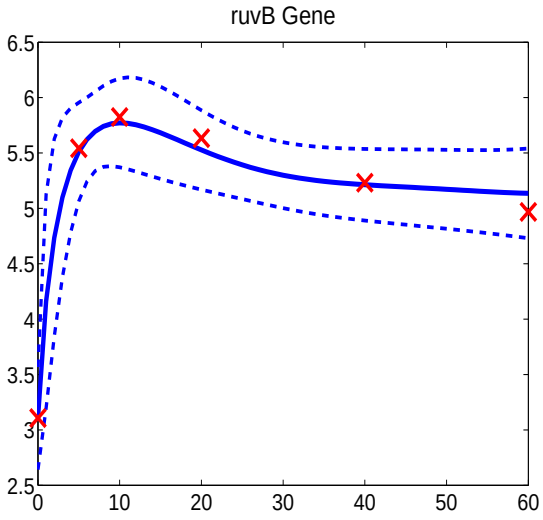


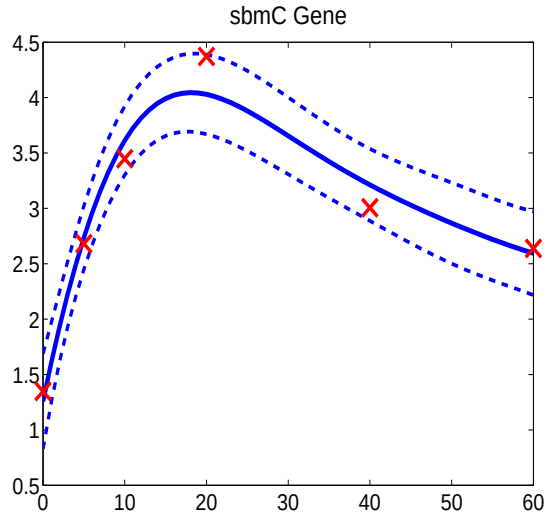


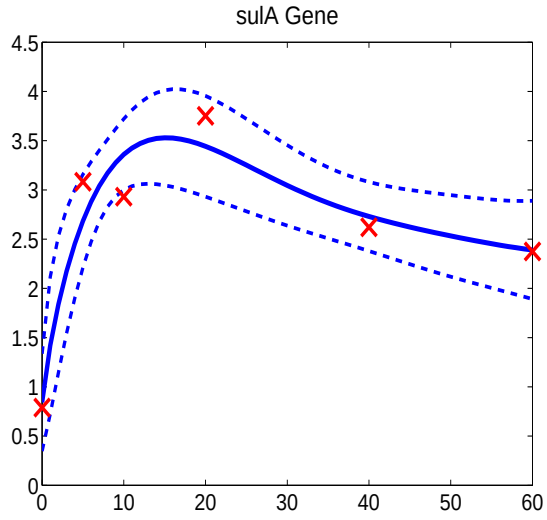


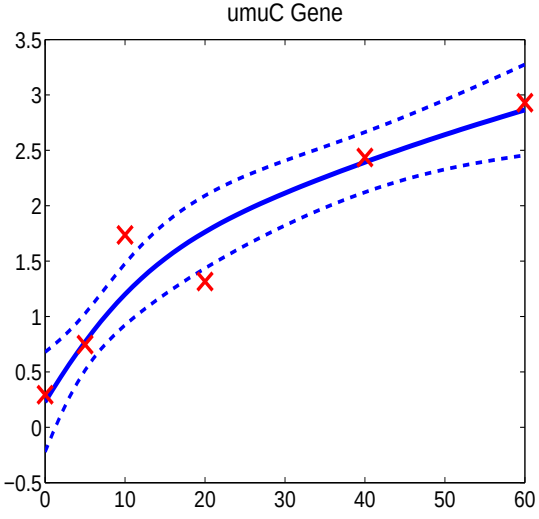


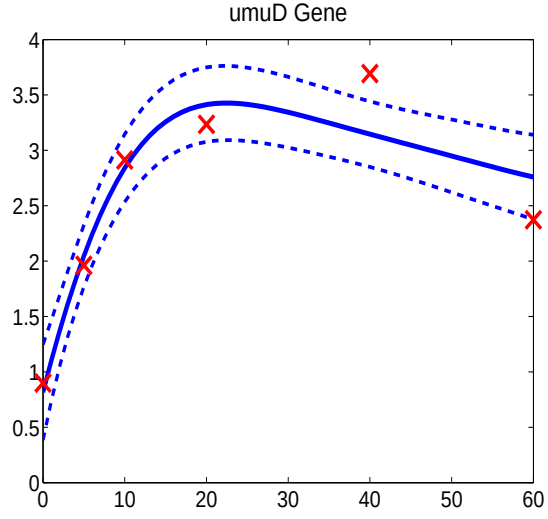
Results in E.coli data: Predicted gene expressions

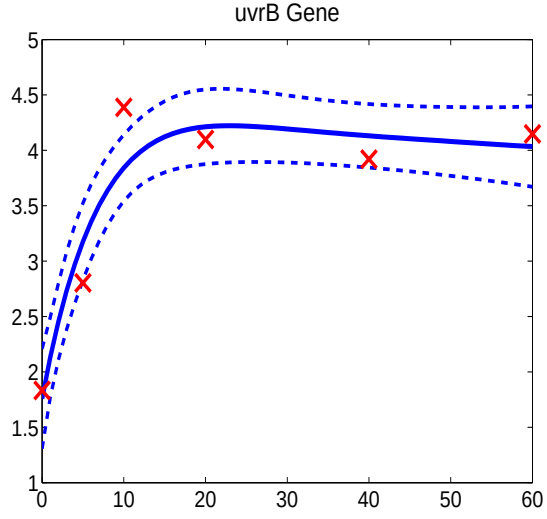




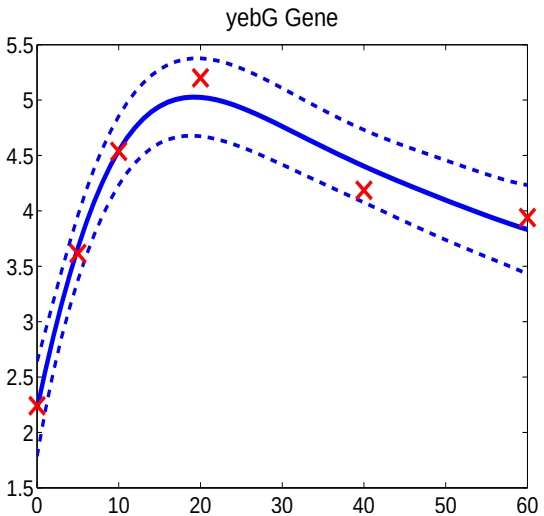


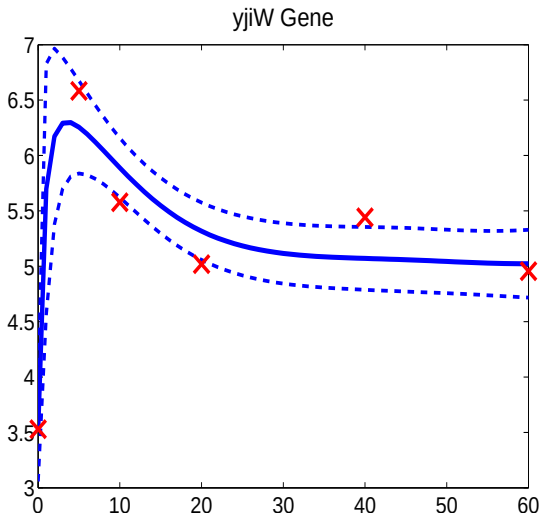




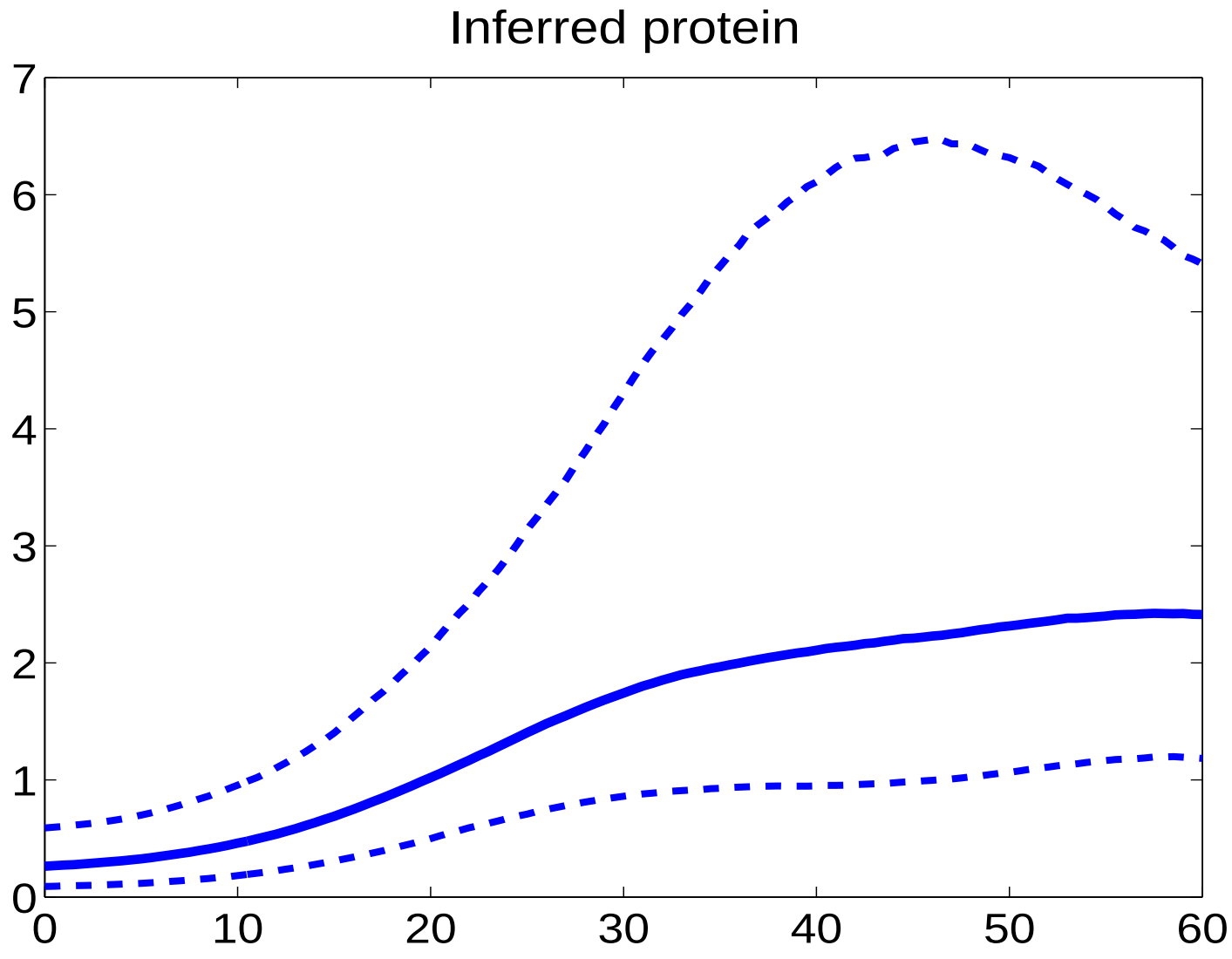


Results in E.coli data: Predicted gene expressions

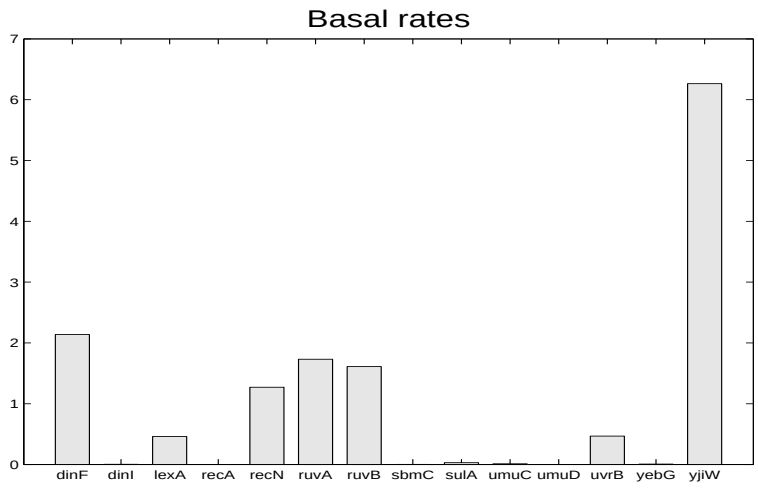


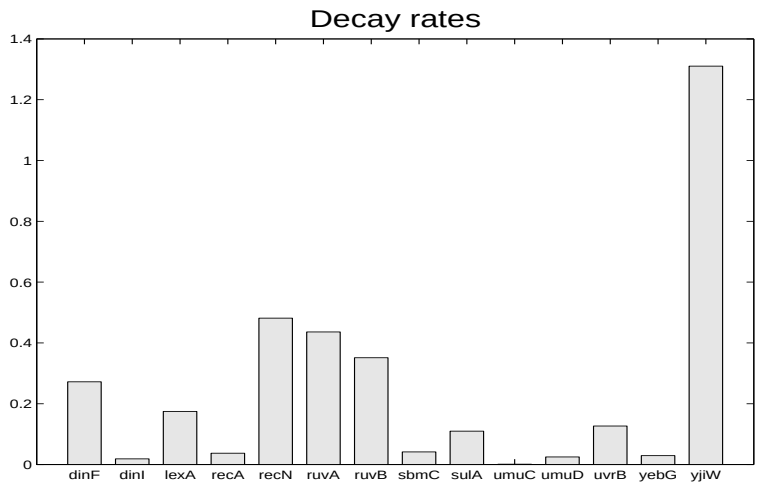


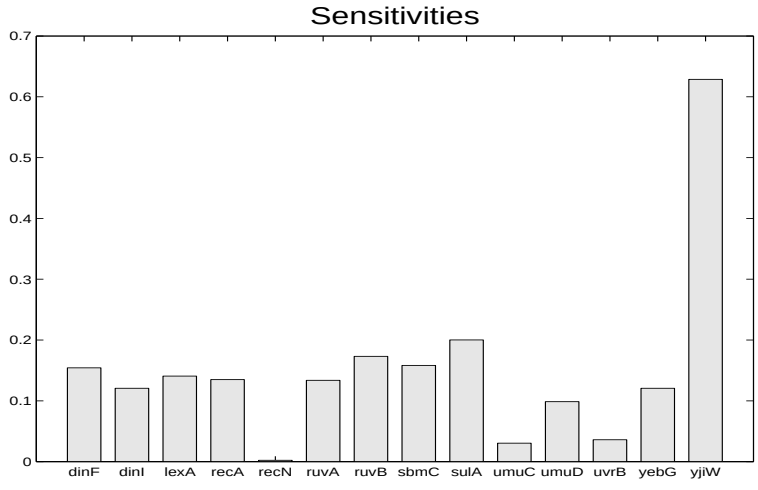
Results in E.coli data: Protein concentration

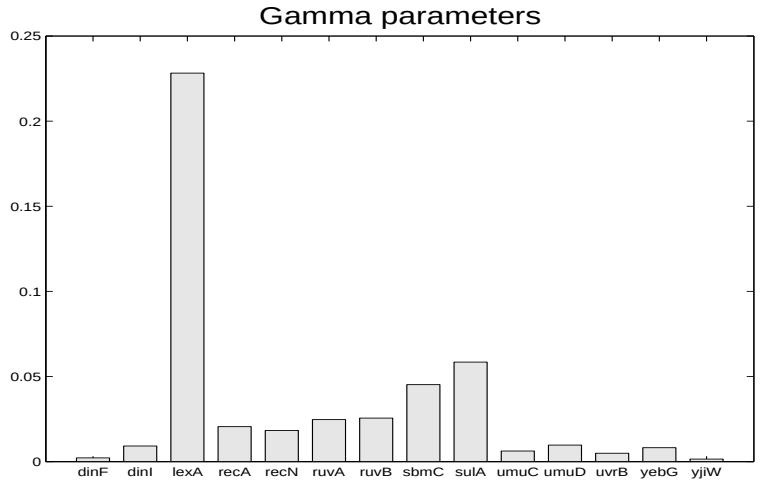


Results in E.coli data: Kinetic parameters

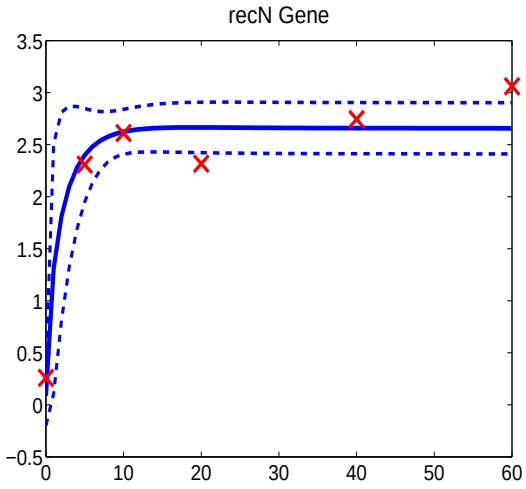


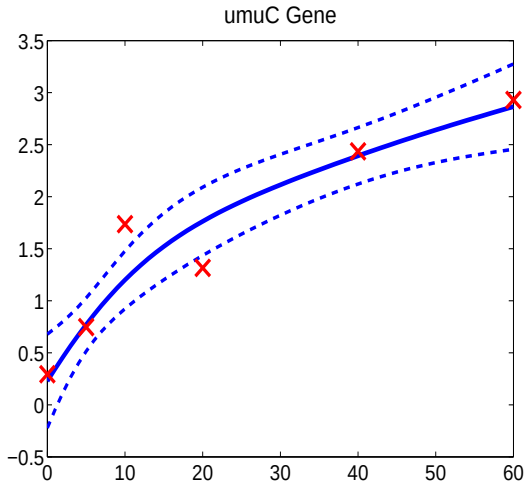


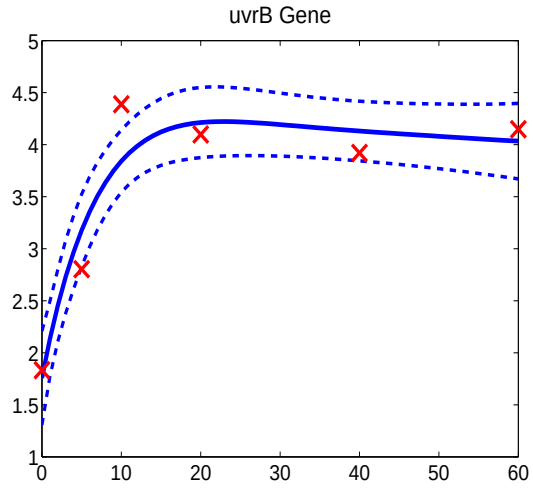


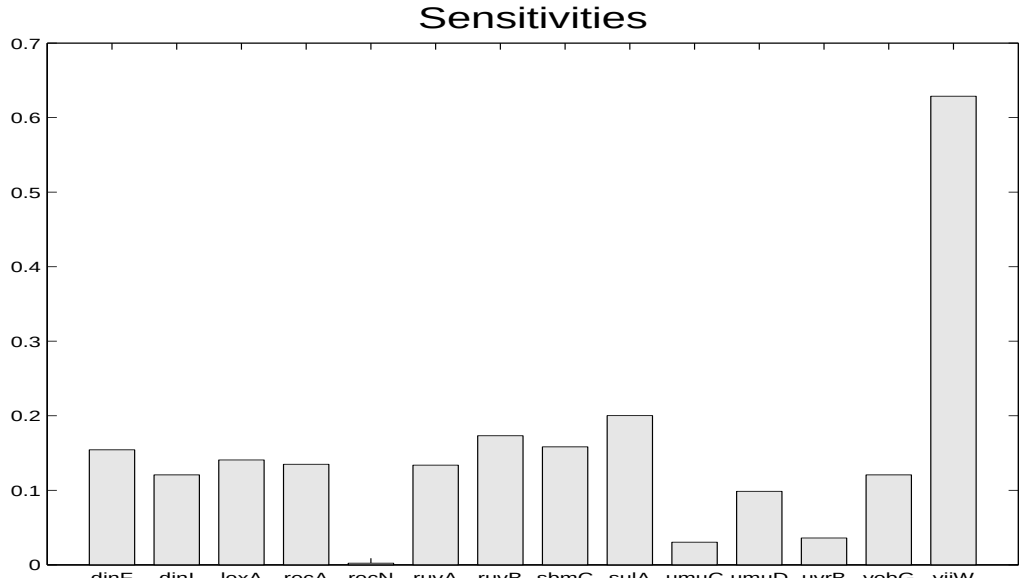


Results in E.coli data: Genes with low sensitivity value

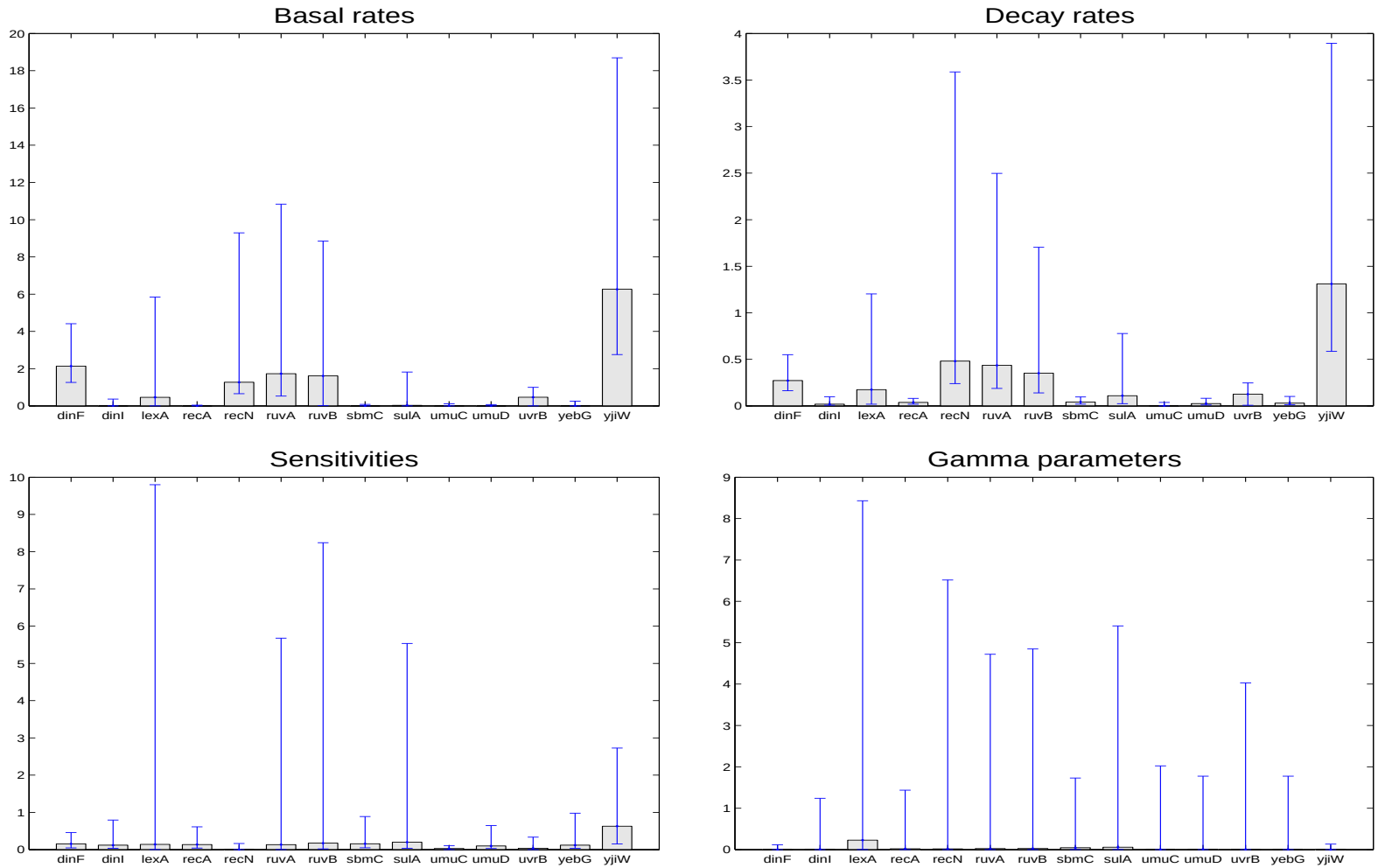








Results in E.coli data: Confidence intervals for the kinetic parameters



BMC Systems Biology

BioMed Central
The Open Access Publisher

Identifying targets of multiple co-regulating transcription factors from expression time-series by Bayesian model comparison

BMC Systems Biology 2012, **6**:53 doi:10.1186/1752-0509-6-53

Michalis K Titsias (mtitsias@well.ox.ac.uk)
Antti Honkela (antti.honkela@hiit.fi)
Neil D Lawrence (n.lawrence@sheffield.ac.uk)
Magnus Rattray (m.rattray@sheffield.ac.uk)

ISSN 1752-0509

Article type Methodology article

A "middle-out" approach for inferring regulatory networks

Task: find targets of a small number of co-regulating transcription factors (TFs) from time-series expression data:

- Stage 1: Sub-network training (~100 targets):
  - Fit regulation model on sub-network of known structure
  - Infer TF protein concentration functions
- Stage 2: Genome-wide scanning:
  - Fit alternative regulation models to all potential targets
  - Score models and identify well supported TF-target links
- Challenges:
  - Fitting and scoring >10000 models
  - Not all regulation is modelled: an open system

A "middle-out" approach for inferring regulatory networks

Task: find targets of a small number of co-regulating transcription factors (TFs) from time-series expression data:

- Stage 1: Sub-network training (~100 targets):
  - Fit regulation model on sub-network of known structure
  - Infer TF protein concentration functions
- Stage 2: Genome-wide scanning:
  - Fit alternative regulation models to all potential targets
  - Score models and identify well supported TF-target links
- Challenges:
  - Fitting and scoring >10000 models
  - Not all regulation is modelled: an open system

A "middle-out" approach for inferring regulatory networks

Task: find targets of a small number of co-regulating transcription factors (TFs) from time-series expression data:

- Stage 1: Sub-network training (~100 targets):
  - Fit regulation model on sub-network of known structure
  - Infer TF protein concentration functions
- Stage 2: Genome-wide scanning:
  - Fit alternative regulation models to all potential targets
  - Score models and identify well supported TF-target links
- Challenges:
  - Fitting and scoring >10000 models
  - Not all regulation is modelled: an open system

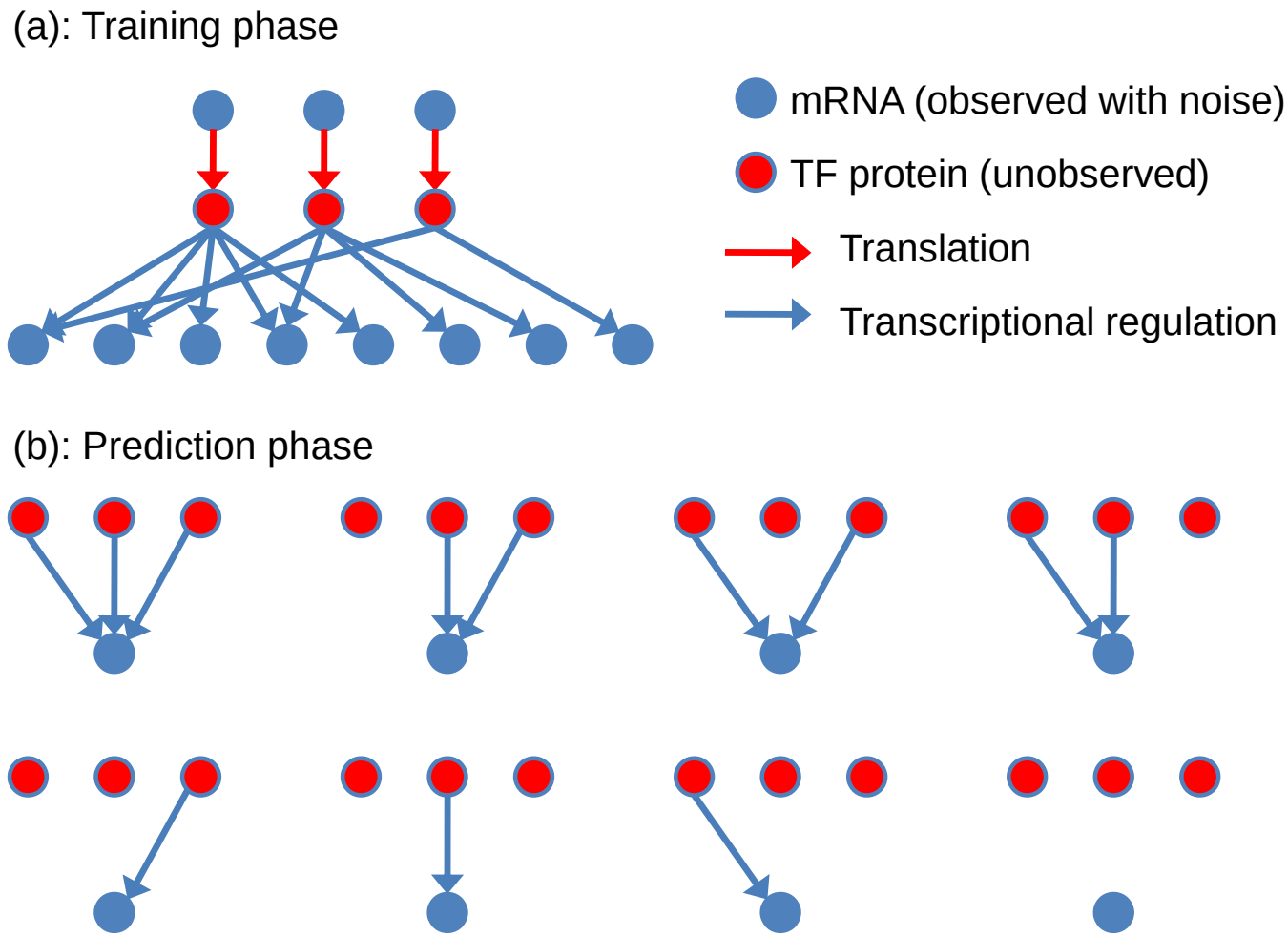
A "middle-out" approach for inferring regulatory networks

Task: find targets of a small number of co-regulating transcription factors (TFs) from time-series expression data:

- Stage 1: Sub-network training (~100 targets):
  - Fit regulation model on sub-network of known structure
  - Infer TF protein concentration functions
- Stage 2: Genome-wide scanning:
  - Fit alternative regulation models to all potential targets
  - Score models and identify well supported TF-target links
- Challenges:
  - Fitting and scoring >10000 models
  - Not all regulation is modelled: an open system

A “middle-out” approach for inferring regulatory networks

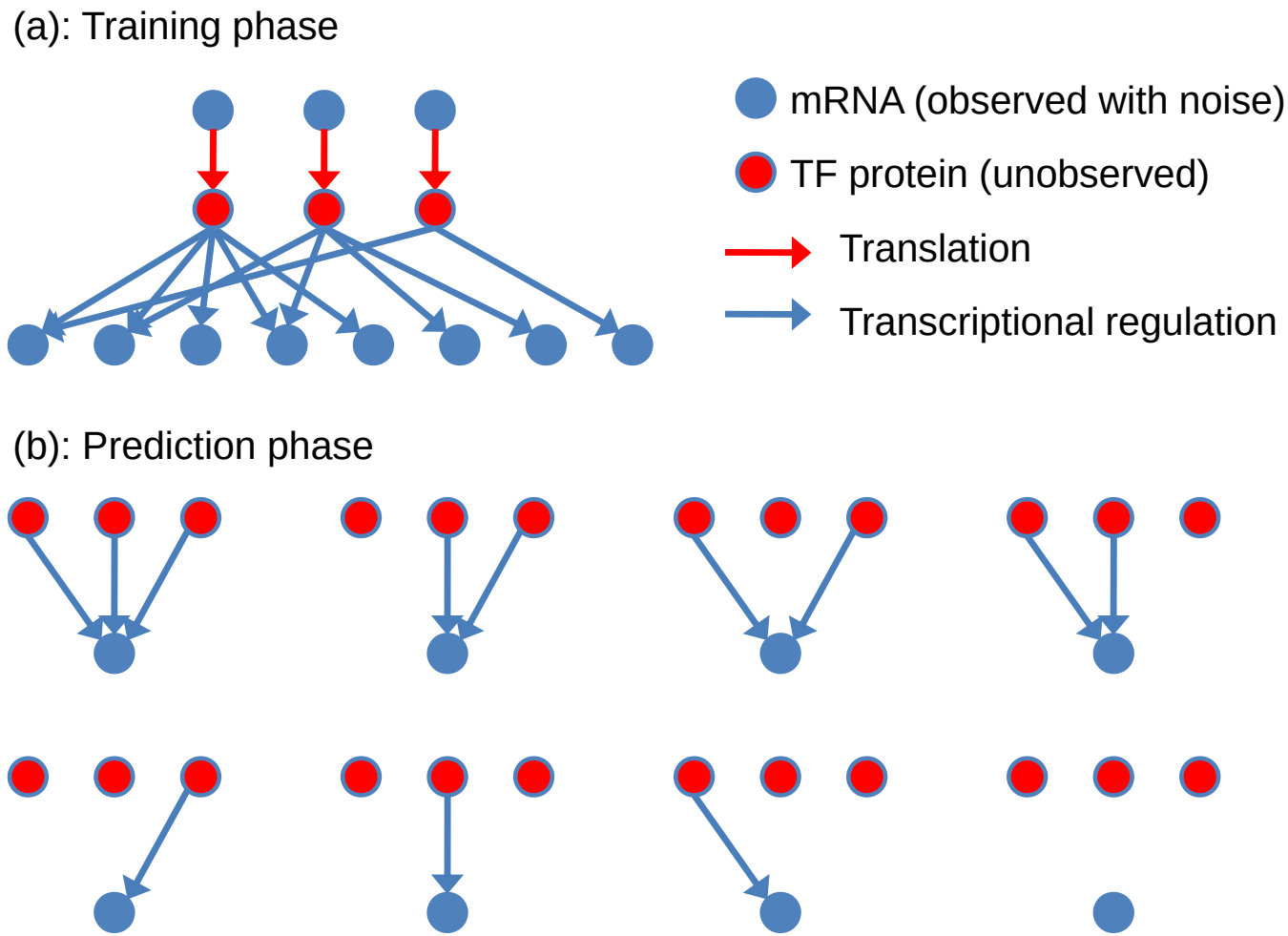
- Training stage: Parameter estimation on known network



- Scanning stage: Bayesian evidence model scoring for

A “middle-out” approach for inferring regulatory networks

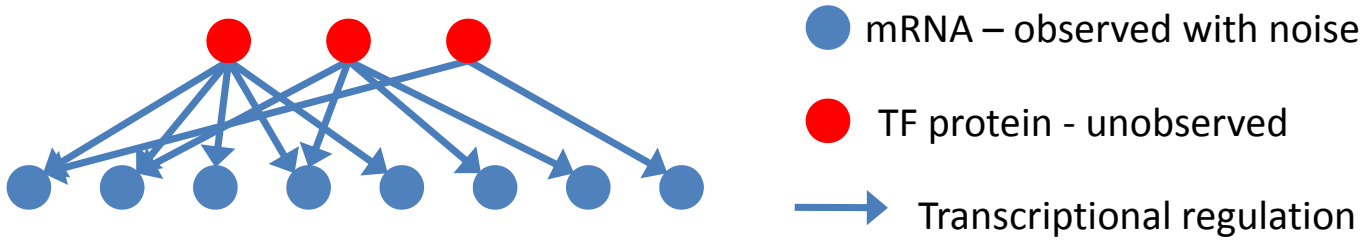
- Training stage: Parameter estimation on known network



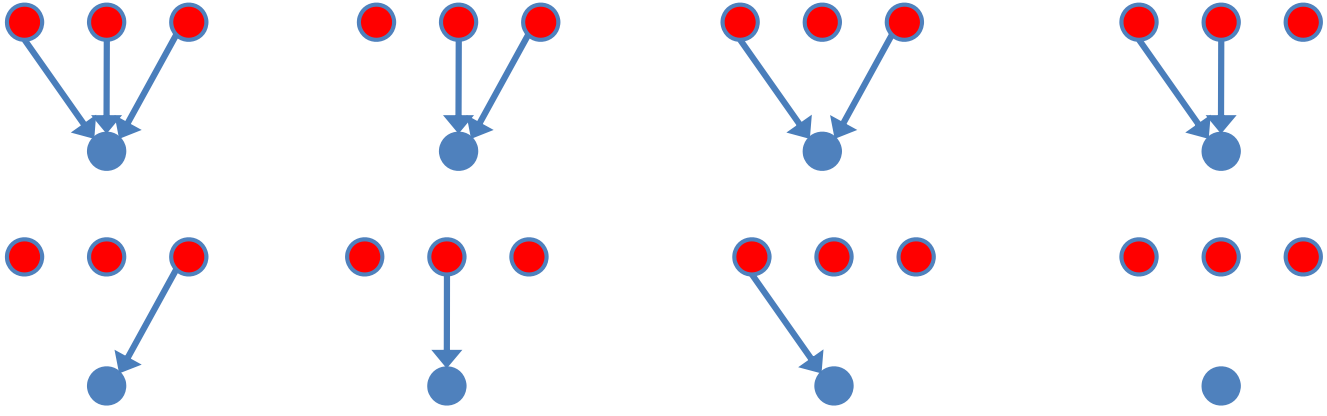
- Scanning stage: Bayesian evidence model scoring for

A “middle-out” approach for inferring regulatory networks

- Training stage with post-translational modification



- Scanning stage: Bayesian evidence model scoring for target inference



Model of transcriptional regulation

- Transcription

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = F\left(p_1(t), \ldots, p_K(t); \boldsymbol{\theta}_j\right) - d_j m_j(t)$$

$m_j(t)$ – target gene j mRNA concentration function
$p_i(t)$ – transcription factor i protein concentration function
$F(\boldsymbol{p}; \boldsymbol{\theta}_j)$ – regulation model, d_j – mRNA decay rate

- Translation (optional)

$$\frac{\mathrm{d}p_i(t)}{\mathrm{d}t} = f_i(t) - \delta_i p_i(t)$$

$f_i(t)$ – transcription factor i mRNA concentration function
δ_i – protein decay rate

Model of transcriptional regulation

- Transcription

$$\frac{\mathrm{d}m_j(t)}{\mathrm{d}t} = F\left(p_1(t), \ldots, p_K(t); \boldsymbol{\theta}_j\right) - d_j m_j(t)$$

$m_j(t)$ – target gene j mRNA concentration function
$p_i(t)$ – transcription factor i protein concentration function
$F(\boldsymbol{p}; \boldsymbol{\theta}_j)$ – regulation model, d_j – mRNA decay rate

- Translation (optional)

$$\frac{\mathrm{d}p_i(t)}{\mathrm{d}t} = f_i(t) - \delta_i p_i(t)$$

$f_i(t)$ – transcription factor i mRNA concentration function
δ_i – protein decay rate

Gaussian process inference over latent functions

- Transcription factors considered **inputs** to the system
- Modelled as samples from a Gaussian process prior distribution
- Equations linear in $m(t)$ can be solved as a function of $p(t)$ so no need for numerical ODE solver to compute likelihood
- Useful way to close an open system
- Can ignore TF mRNA data and treat $p(t)$ as latent function
- Bayesian MCMC used to infer $p(t)$ and all model parameters

Gao et al. (2008); Titsias et al. (2009); Honkela et al. (2010); Titsias et al. (2012)

Gaussian process inference over latent functions

- Transcription factors considered **inputs** to the system
- Modelled as samples from a Gaussian process prior distribution
- Equations linear in $\boldsymbol{m}(t)$ can be solved as a function of $\boldsymbol{p}(t)$ so no need for numerical ODE solver to compute likelihood
- Useful way to close an open system
- Can ignore TF mRNA data and treat $p(t)$ as latent function
- Bayesian MCMC used to infer $p(t)$ and all model parameters

Gao et al. (2008); Titsias et al. (2009); Honkela et al. (2010); Titsias et al. (2012)

Gaussian process inference over latent functions

- Transcription factors considered **inputs** to the system
- Modelled as samples from a Gaussian process prior distribution
- Equations linear in $\boldsymbol{m}(t)$ can be solved as a function of $\boldsymbol{p}(t)$ so no need for numerical ODE solver to compute likelihood
- Useful way to close an open system
- Can ignore TF mRNA data and treat $\boldsymbol{p}(t)$ as latent function
- Bayesian MCMC used to infer $p(t)$ and all model parameters

Gao et al. (2008); Titsias et al. (2009); Honkela et al. (2010); Titsias et al. (2012)

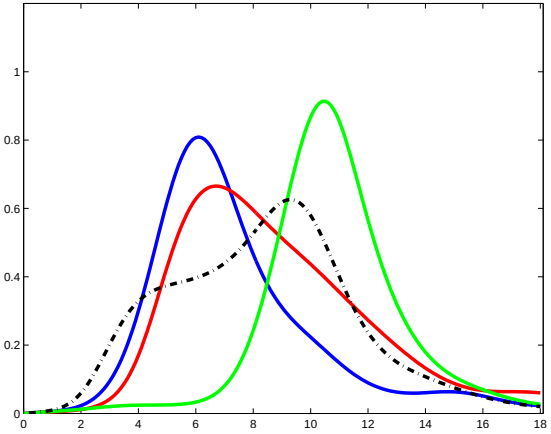
Gaussian process inference over latent functions

- Transcription factors considered **inputs** to the system
- Modelled as samples from a Gaussian process prior distribution
- Equations linear in $\boldsymbol{m}(t)$ can be solved as a function of $\boldsymbol{p}(t)$ so no need for numerical ODE solver to compute likelihood
- Useful way to close an open system
- Can ignore TF mRNA data and treat $\boldsymbol{p}(t)$ as latent function
- Bayesian MCMC used to infer $\boldsymbol{p}(t)$ and all model parameters

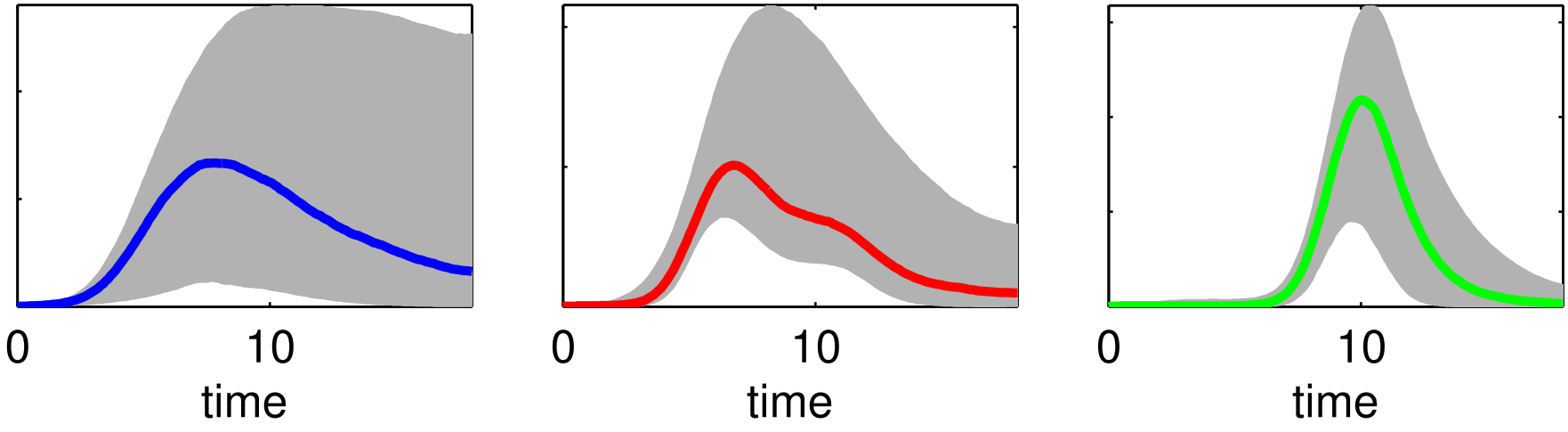
Gao et al. (2008); Titsias et al. (2009); Honkela et al. (2010); Titsias et al. (2012)

Artificial data: one experimental condition

Ground Truth TFs

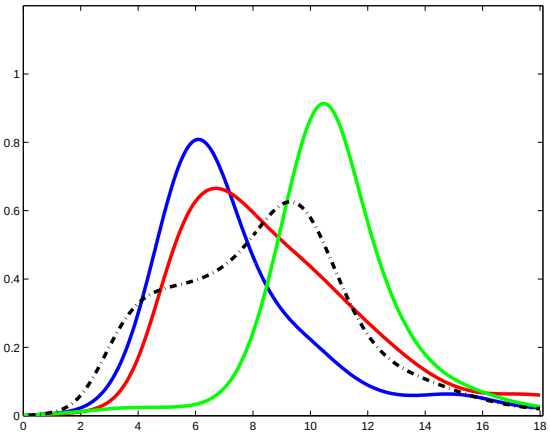


Inferred TF concentrations after training stage

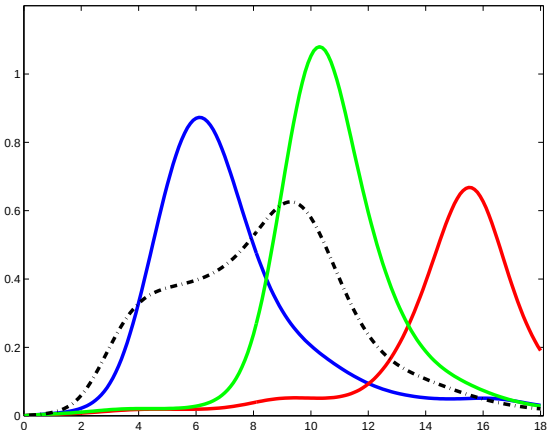


Artificial data: two experimental conditions

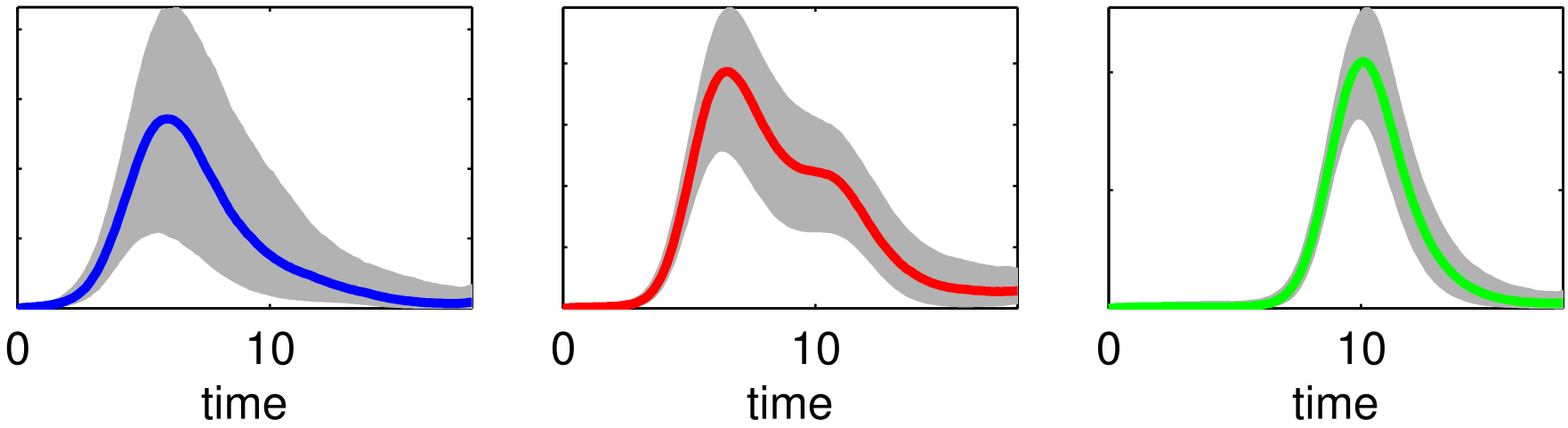
True TFs condition 1



True TFs condition 2

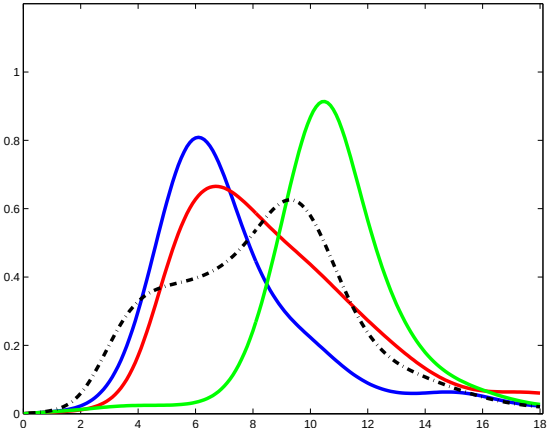


Inferred TF concentrations for condition 1

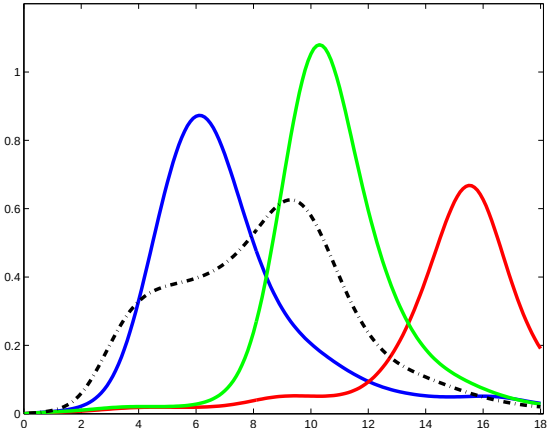


Artificial data: two experimental conditions

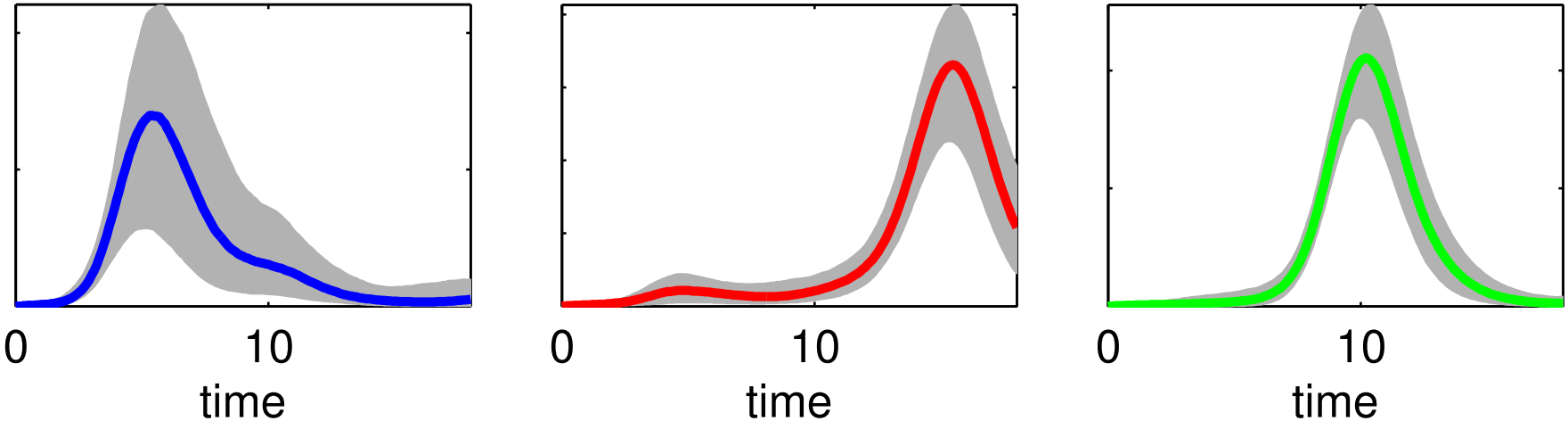
True TFs condition 1



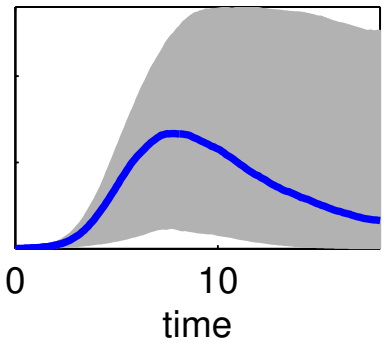
True TFs condition 2

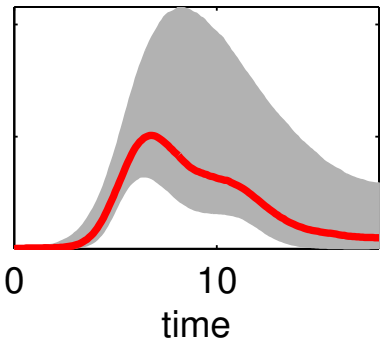


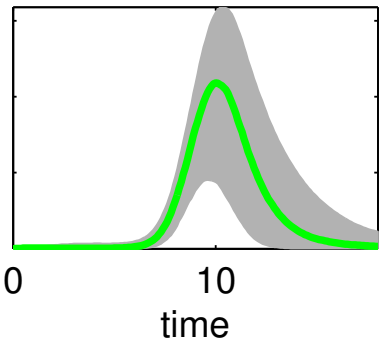
Inferred TF concentrations for condition 2

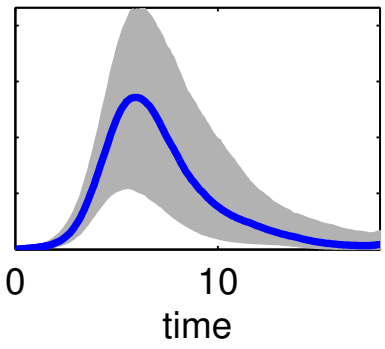


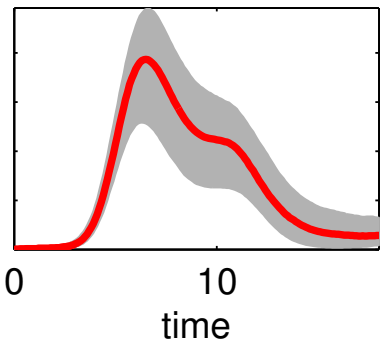
Artificial data: scanning performance for each TF

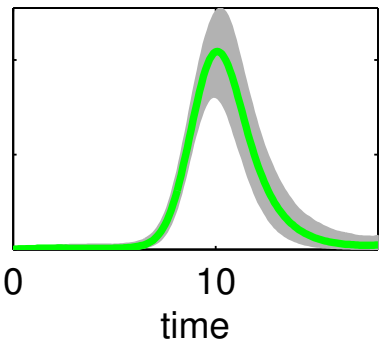


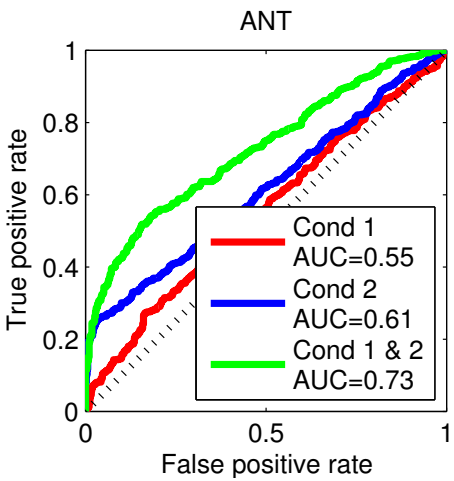


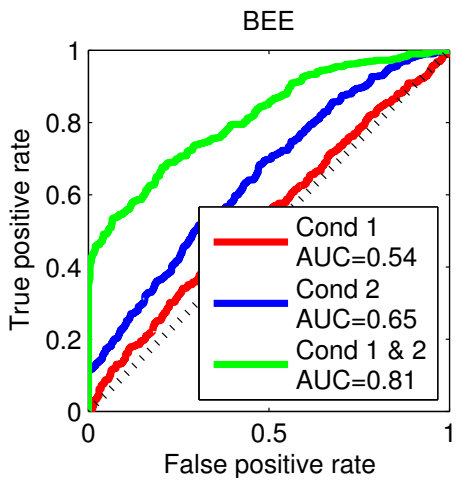


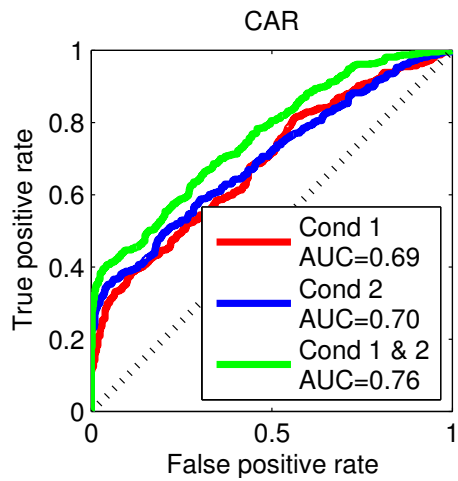




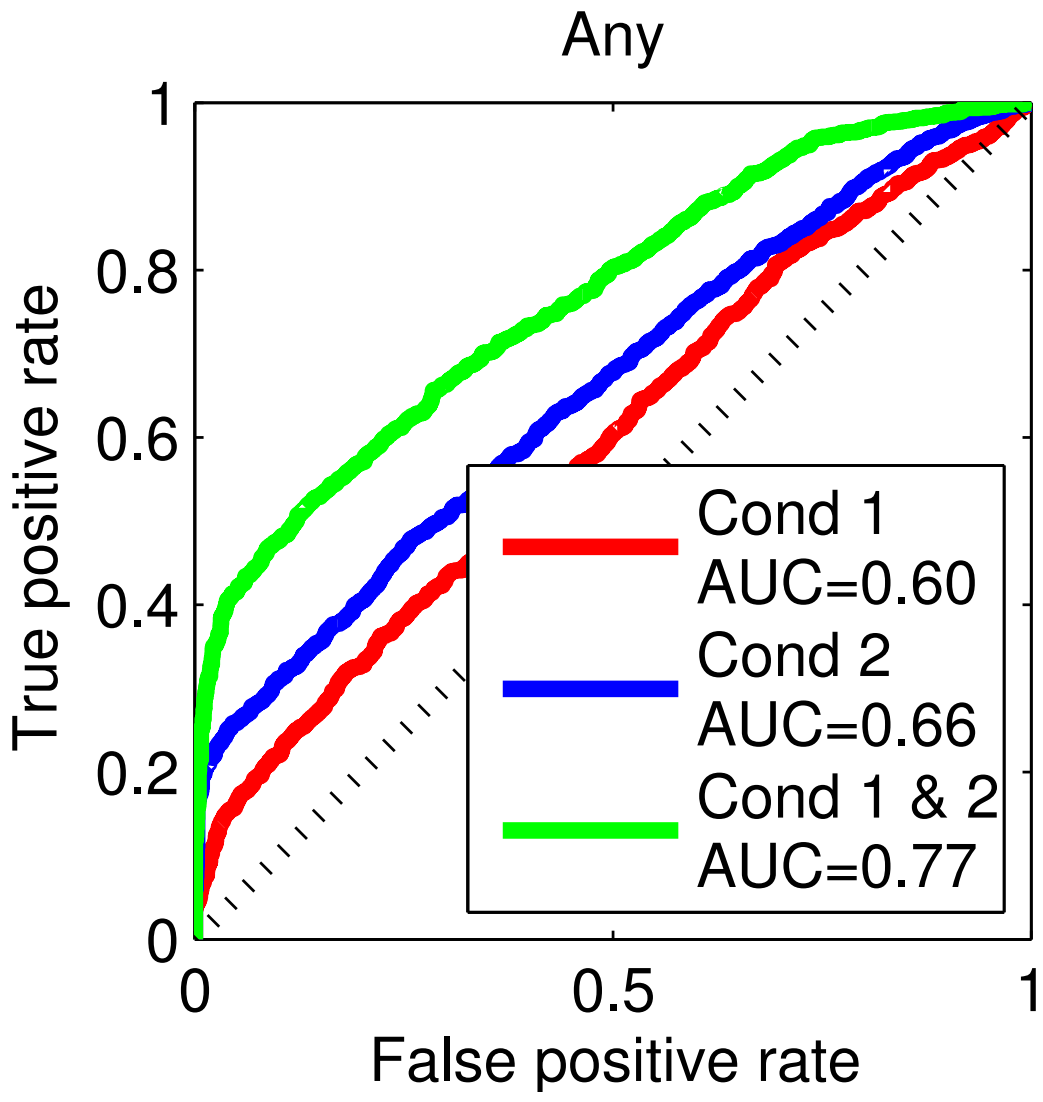






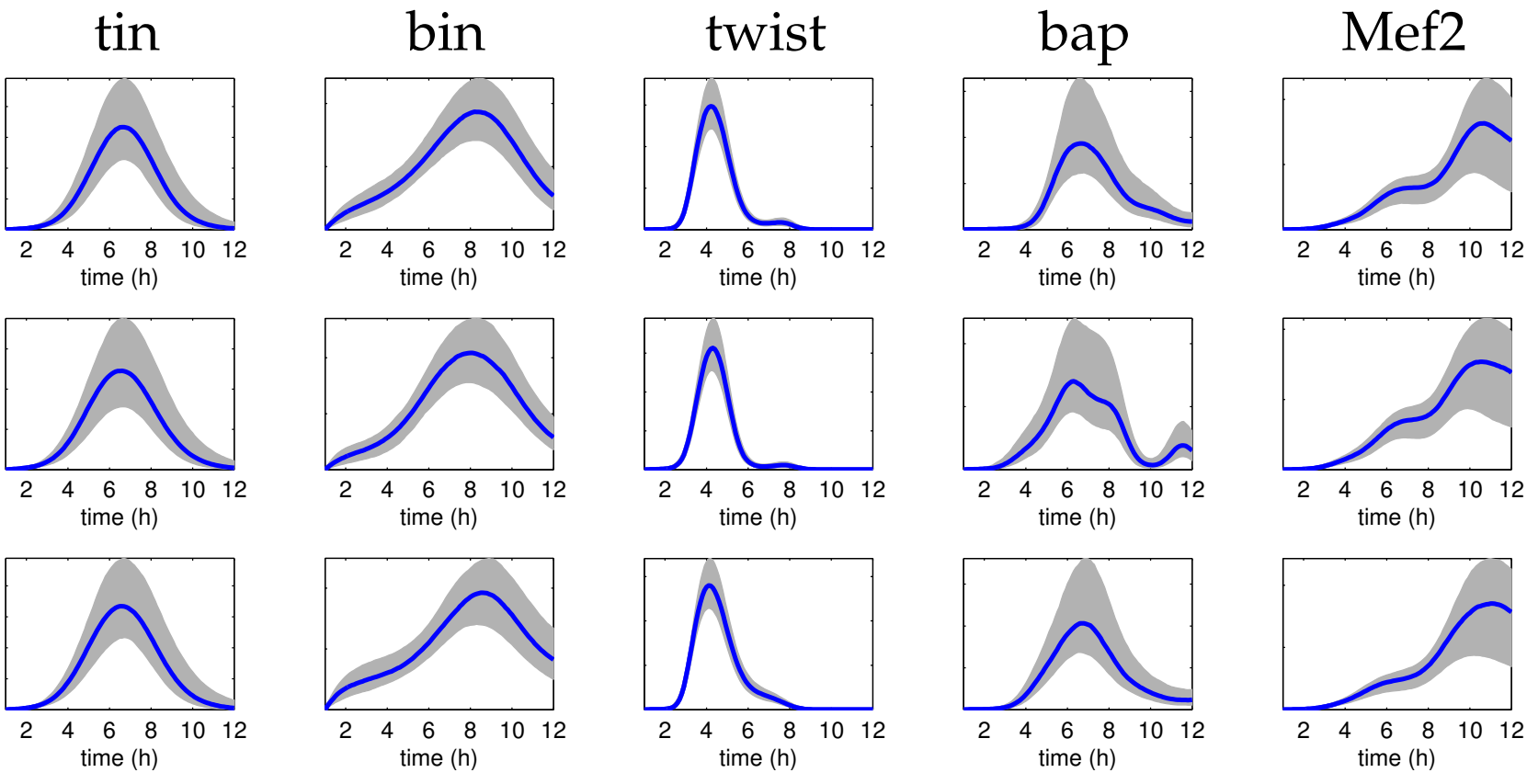


Artificial data: scanning performance for all TFs



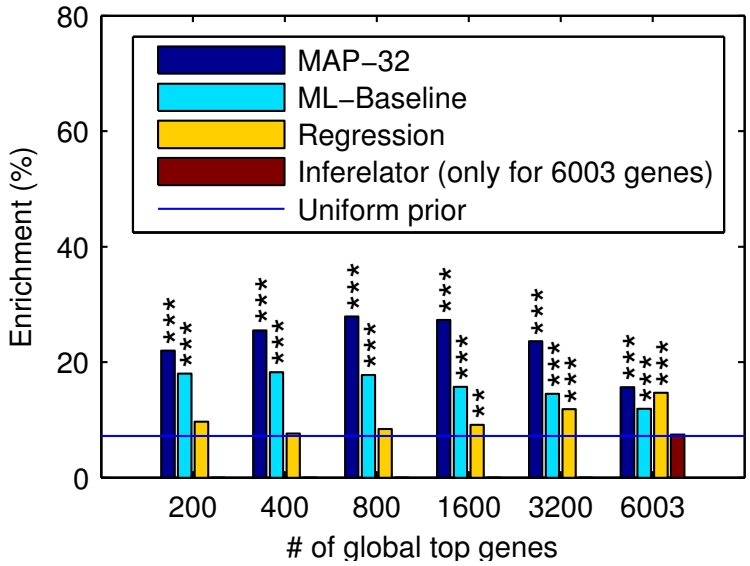
Drosophila training

- Sub-network of 96 genes targeted by 5 TFs during Drosophila mesoderm development (Zinzen et al., 2009).
- Data: wild-type times series, 3 replicates (Tomancak et al., 2002).

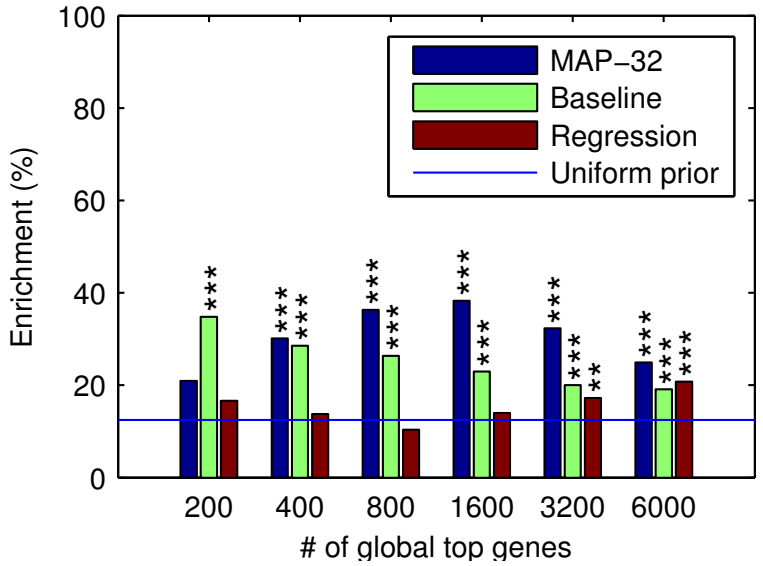


Drosophila scanning: model ranking

- Rank target gene regulation models by their posterior probability across all $2^5 = 32$ possible models
- Validate predicted links by enrichment for genes within 2kb of ChIP-chip TF binding predictions from Zinzen et al. (2009).

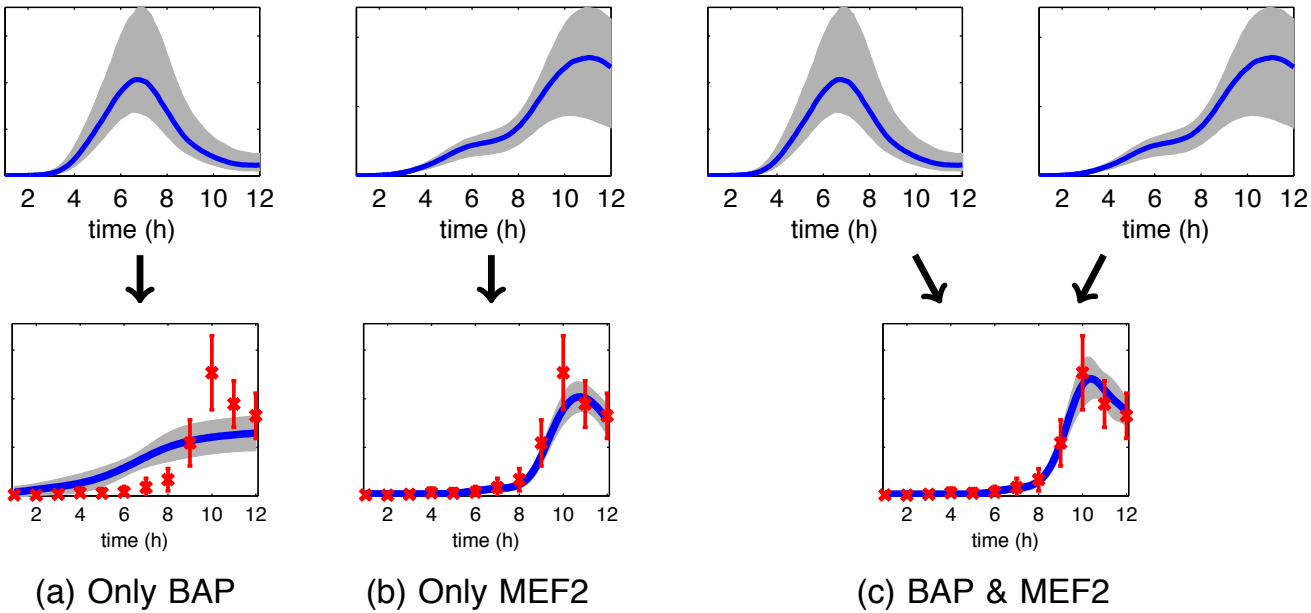


All "non-quiet" genes



All targets with in situ evidence

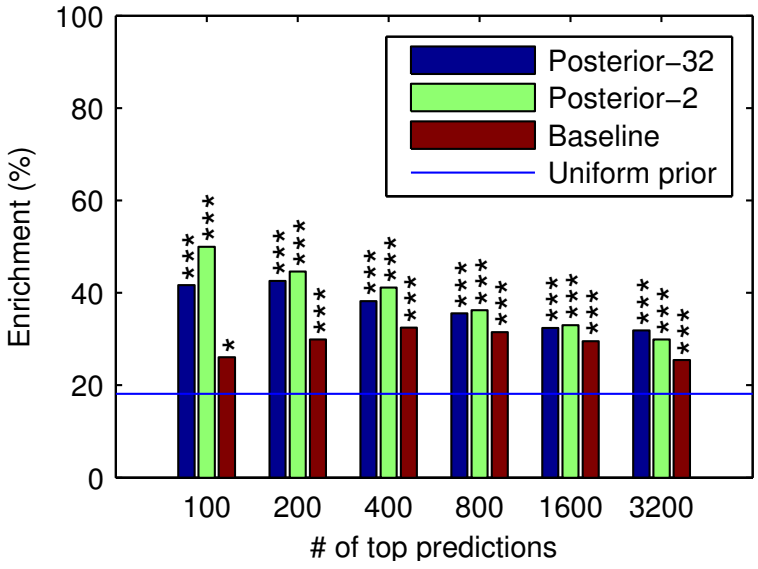
Coregulated Target Example



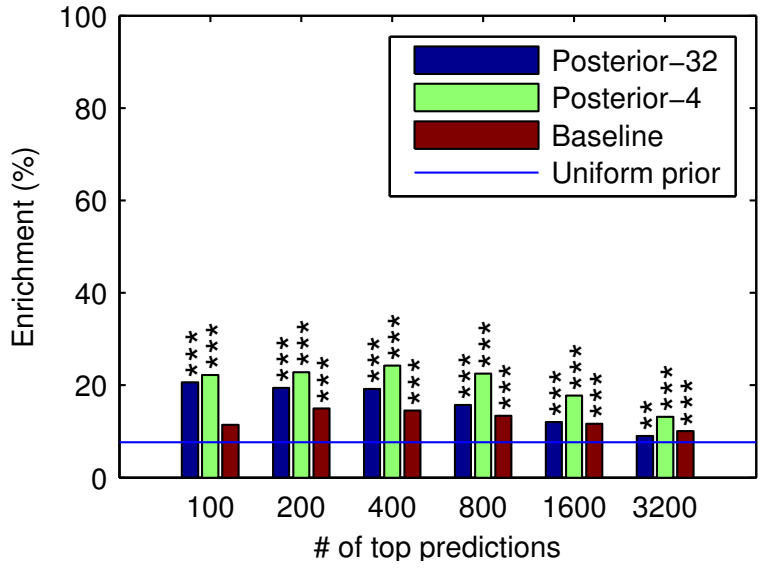
A highly ranked putative joint target of BAP amd MEF2. The candidate gene is confirmed as a joint target by independent ChIP-chip studies Zinzen et al. (2009).

Drosophila scanning: link ranking

- TF-target link and link-pair ranking according to posterior probability of particular single TF or double TF regulations
- Validate predicted links by enrichment for genes within 2kb of ChIP-chip TF binding predictions from Zinzen et al. (2009).



TF regulation



TF pair regulation

Summary and Conclusion

- Middle-out approach: sub-network training followed by genome-wide scanning
- Training: Bayesian inference of regulation model parameters and TF protein concentration functions
- Scanning: Bayesian model scoring for inferring TF-target link probabilities
- More informative conditions $\rightarrow$ better performance
- Robust to existence of some unknown regulating TFs
- Significant enrichment of inferred TF-target links for nearby ChIP-chip binding in drosophila development example

Summary and Conclusion

- Middle-out approach: sub-network training followed by genome-wide scanning
- Training: Bayesian inference of regulation model parameters and TF protein concentration functions
- Scanning: Bayesian model scoring for inferring TF-target link probabilities
- More informative conditions → better performance
- Robust to existence of some unknown regulating TFs
- Significant enrichment of inferred TF-target links for nearby ChIP-chip binding in drosophila development example

Summary and Conclusion

- Middle-out approach: sub-network training followed by genome-wide scanning
- Training: Bayesian inference of regulation model parameters and TF protein concentration functions
- Scanning: Bayesian model scoring for inferring TF-target link probabilities
- More informative conditions $\rightarrow$ better performance
- Robust to existence of some unknown regulating TFs
- Significant enrichment of inferred TF-target links for nearby ChIP-chip binding in drosophila development example