Kernels for Vector-Valued Functions

Neil D. Lawrence
includes work with Mauricio Alvarez and Lorenzo Rosasco

ICML Workshop on Next Generation Kernels
30th June 2012

Outline

1. Background
2. Convolution Processes
3. Motion Capture Example

Latent Function Perspective

- Introduce vector valued functions through latent function perspective.
- Vector valued function, $f(\cdot)$ is linearly dependent on latent function, $u(\cdot)$.
- Gaussian process perspective:
  - If the latent function is a Gaussian process.
  - Observed function is also a Gaussian process.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$E[u(\mathbf{x})u(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}')$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\langle u(\mathbf{x})u(\mathbf{x}')\rangle = k(\mathbf{x}, \mathbf{x}')$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$f_1(\mathbf{x}) = w_1 u(\mathbf{x})$$
$$f_2(\mathbf{x}) = w_2 u(\mathbf{x})$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} w_1 u(\mathbf{x}) \\ w_2 u(\mathbf{x}) \end{bmatrix}$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\left\langle \mathbf{f}(\mathbf{x})\mathbf{f}(\mathbf{x}')^\top \right\rangle = \begin{bmatrix} w_1^2 k(\mathbf{x}, \mathbf{x}') & w_1 w_2 k(\mathbf{x}, \mathbf{x}') \\ w_1 w_2 k(\mathbf{x}, \mathbf{x}') & w_2^2 k(\mathbf{x}, \mathbf{x}') \end{bmatrix}$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\left\langle \mathbf{f}(\mathbf{x})\mathbf{f}(\mathbf{x}')^\top \right\rangle = k(\mathbf{x}, \mathbf{x}') \begin{bmatrix} w_1^2 & w_1 w_2 \\ w_1 w_2 & w_2^2 \end{bmatrix}$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\left\langle \mathbf{f}(\mathbf{x})\mathbf{f}(\mathbf{x}')^\top \right\rangle = k(\mathbf{x}, \mathbf{x}')\mathbf{w}\mathbf{w}^\top$$

- For zero mean process (convenient) it is the second moment.

Probabilistic Perspective

- Kernel function is covariance of probabilistic process (need not be Gaussian!).

$$\left\langle \mathbf{f}(\mathbf{x})\mathbf{f}(\mathbf{x}')^\top \right\rangle = k(\mathbf{x}, \mathbf{x}')\mathbf{B}$$

- For zero mean process (convenient) it is the second moment.

Coregionalization Matrix

- In above example coregionalization matrix, $\mathbf{B}$, is reduced rank.
- If $\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{u}(\mathbf{x})$
- Where elements of $\mathbf{u}(\mathbf{x})$ are *independent* each with covariance $k(\mathbf{x}, \mathbf{x}')$.

$$\left\langle \mathbf{f}(\mathbf{x})\mathbf{f}(\mathbf{x}')^\top \right\rangle = k(\mathbf{x}, \mathbf{x}')\mathbf{B}$$
$$\mathbf{B} = \mathbf{W}\mathbf{W}^\top$$

Simple Markov Chain

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

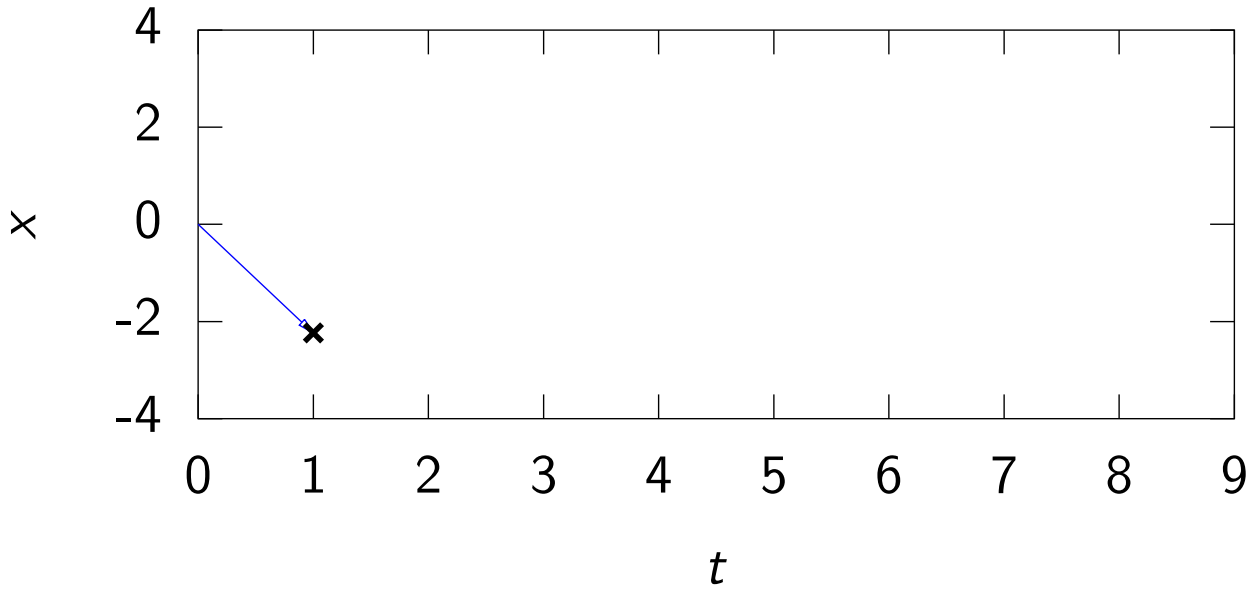
$$\begin{aligned} x_i =& x_{i-1} + \epsilon_i, \\ \epsilon_i \sim& \mathcal{N}(0, \alpha) \\ \implies x_i \sim& \mathcal{N}(x_{i-1}, \alpha) \end{aligned}$$

- 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

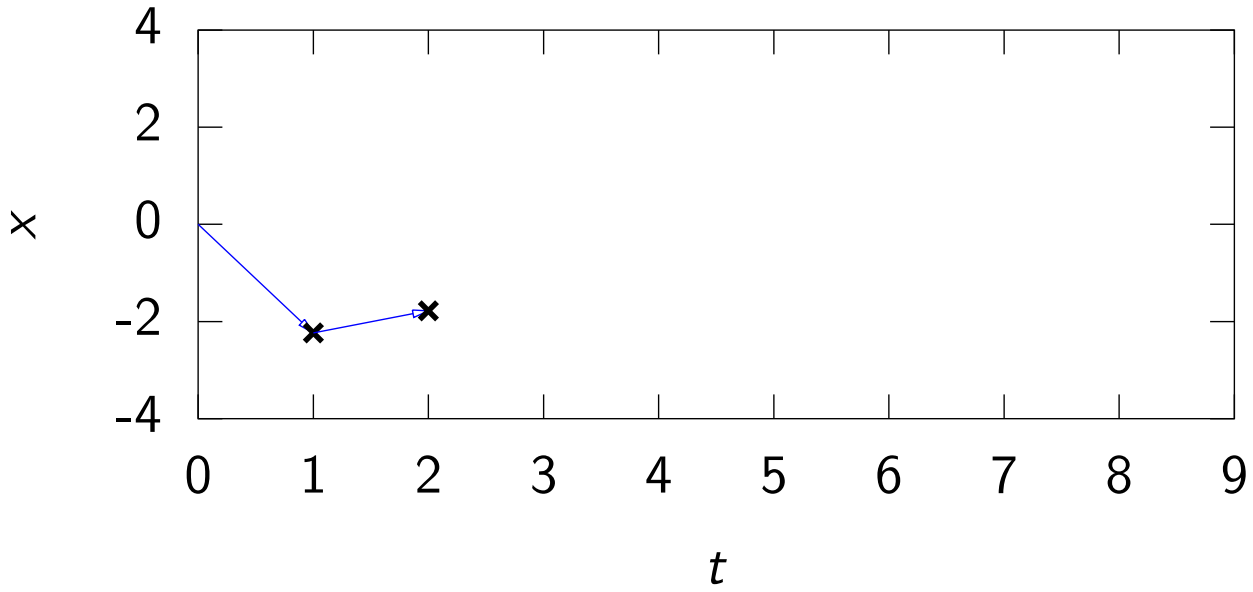


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

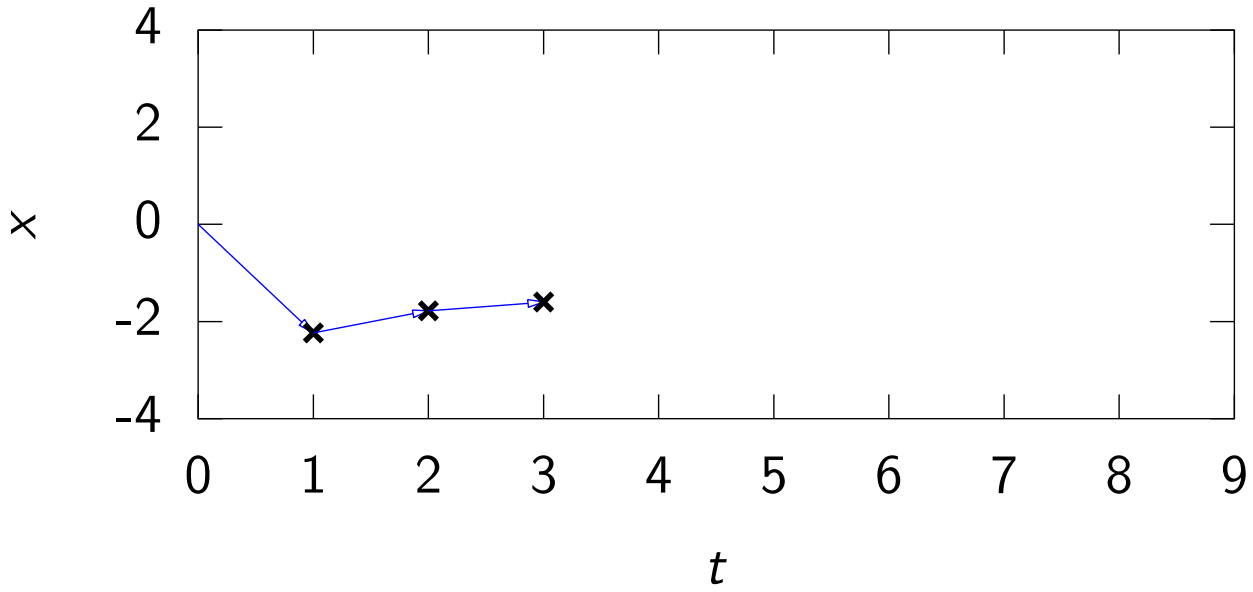


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

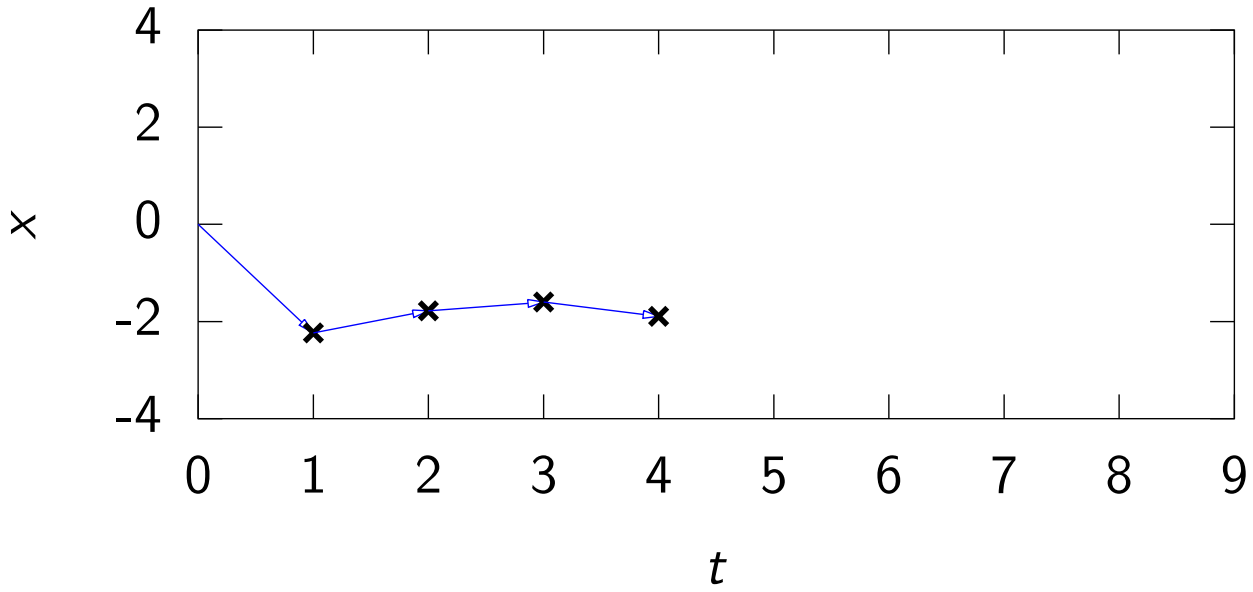


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

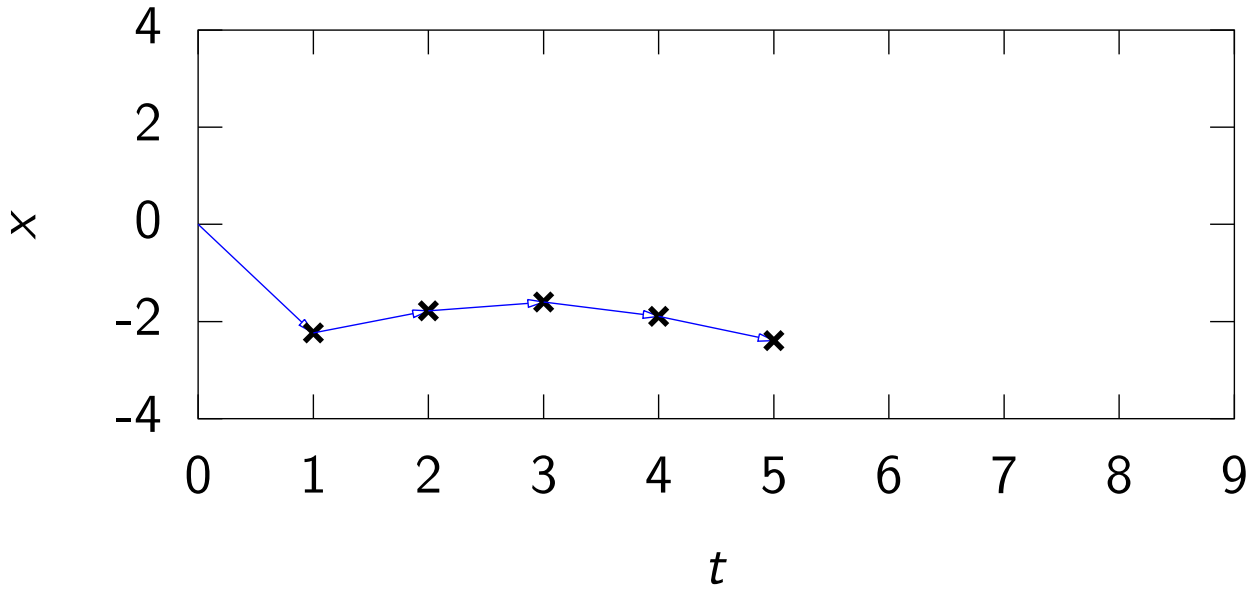


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

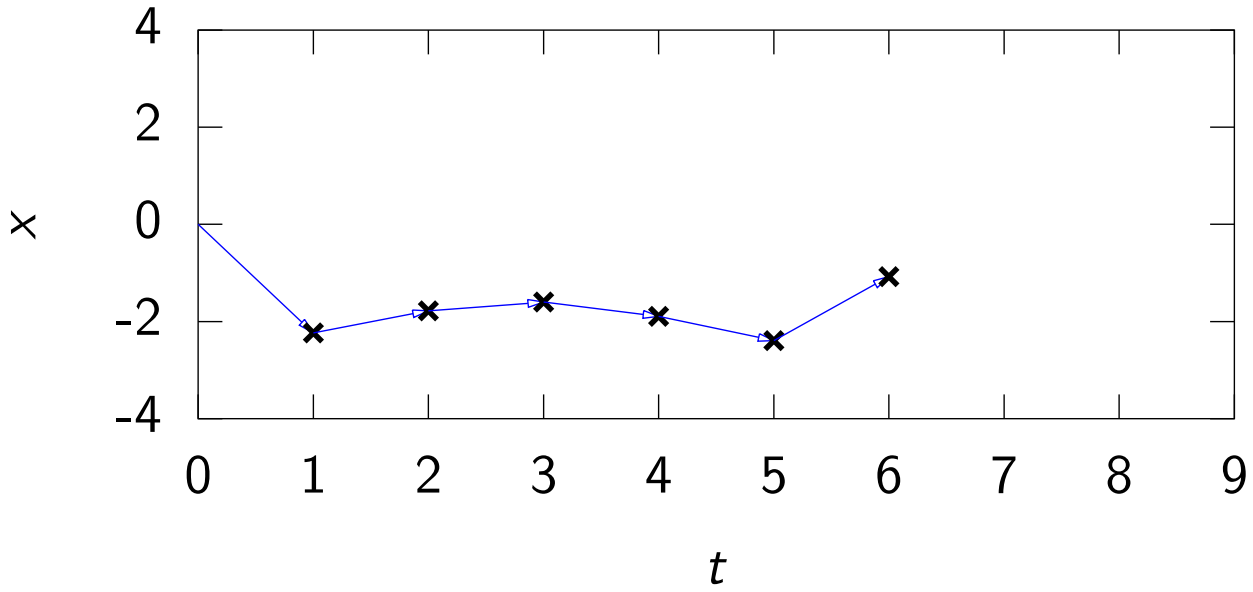


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

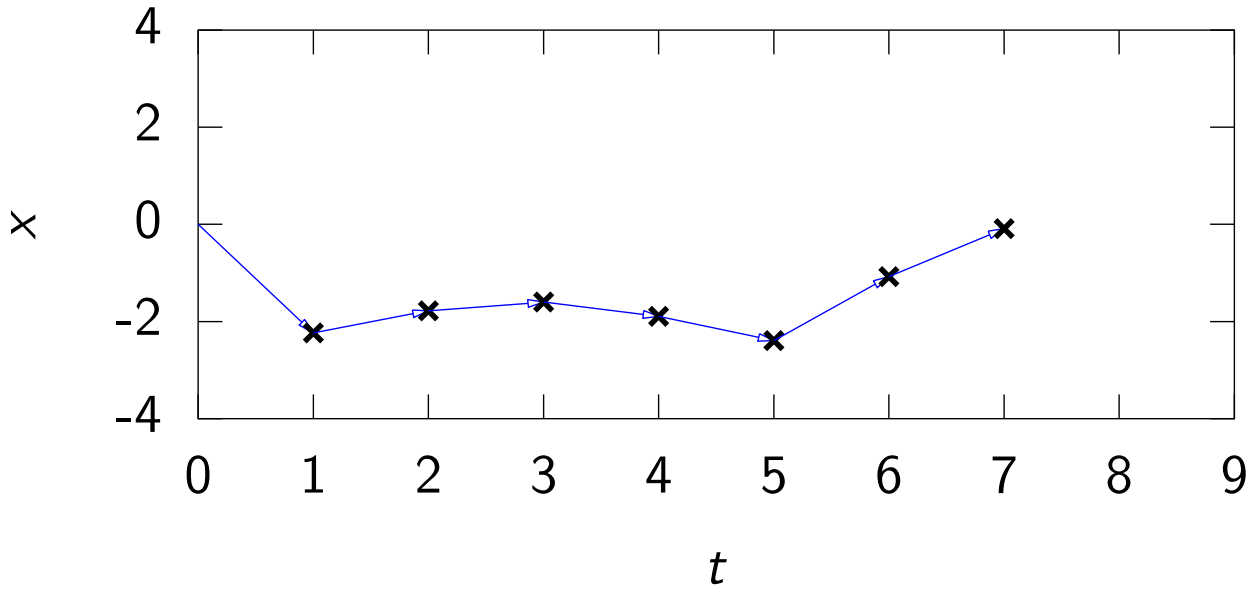


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

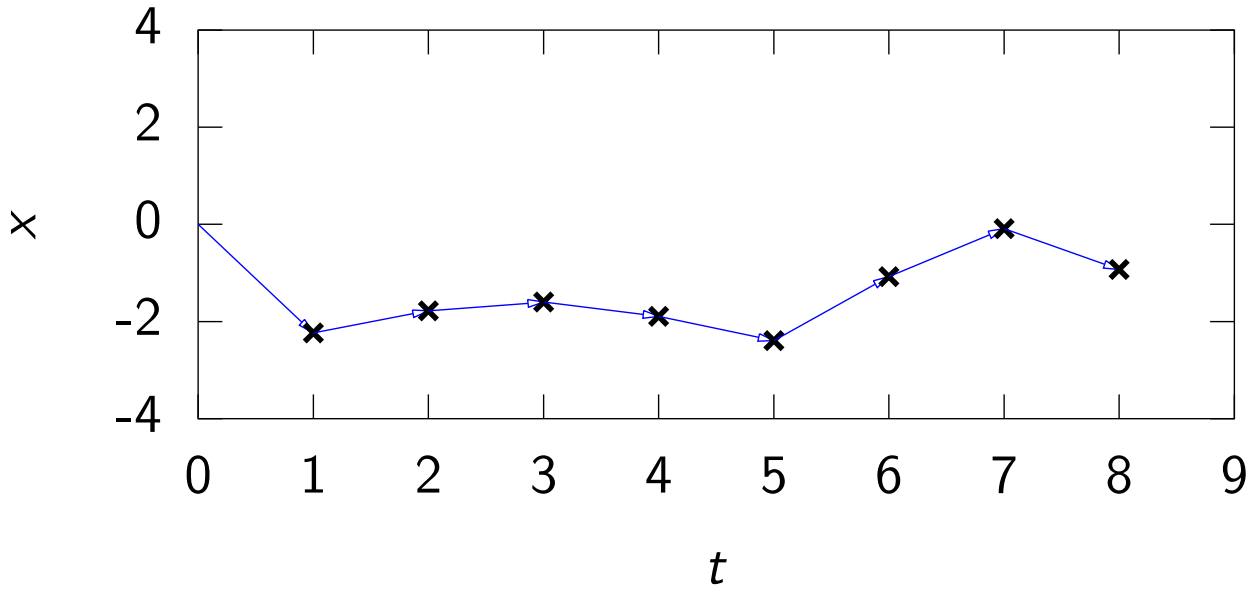


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

Gauss Markov Chain

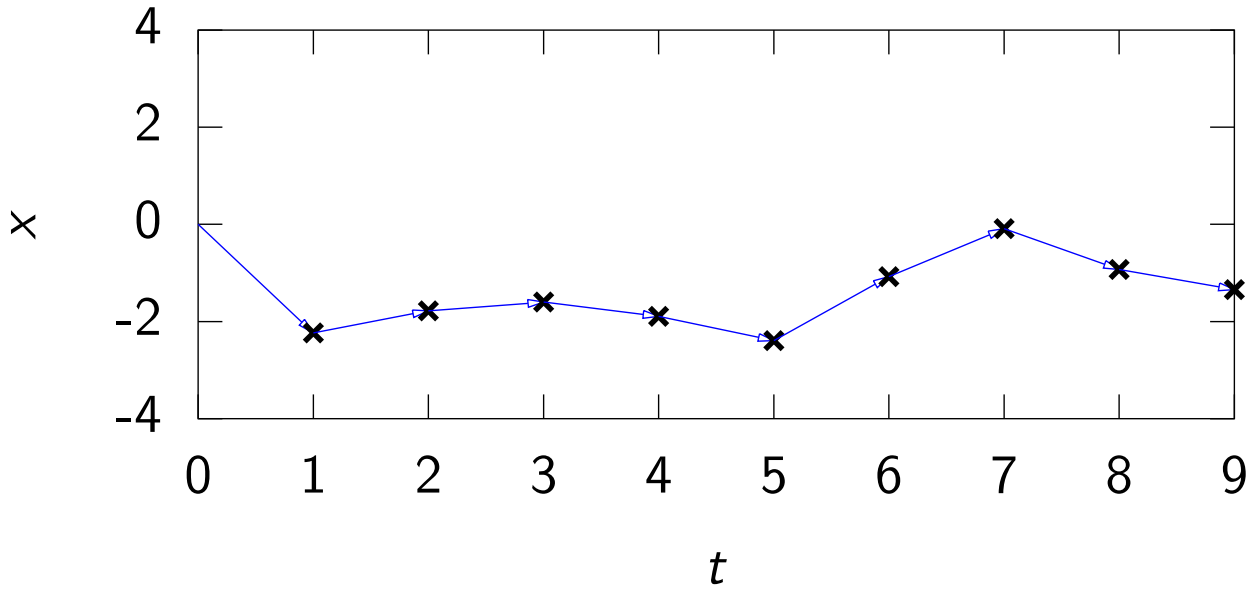


$$x_0 = 0, \quad \epsilon_i \sim \mathcal{N}(0, 1)$$

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

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

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

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

and

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

then

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

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

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

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

Matrix Representation of Latent Variables

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

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

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

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

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

Latent Process

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

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

Latent Process

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

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

$$\Longrightarrow$$

Latent Process

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

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

$$\Longrightarrow$$

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

Covariance for Latent Process II

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

Covariance for Latent Process II

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

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

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

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 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 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 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

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

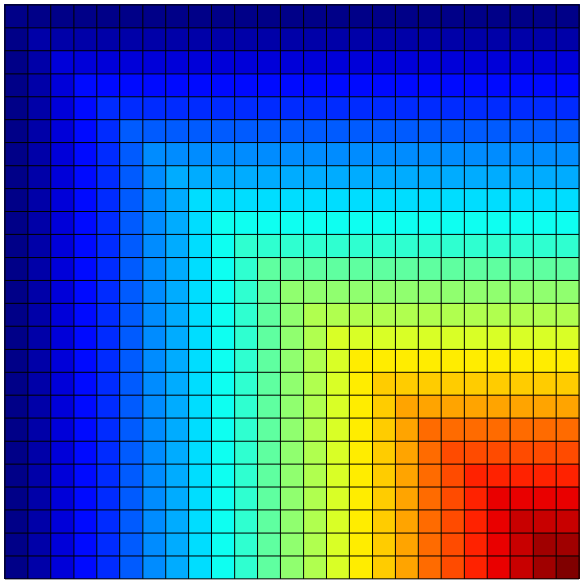
Covariance Functions

Where did this covariance matrix come from?

Markov Process

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

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



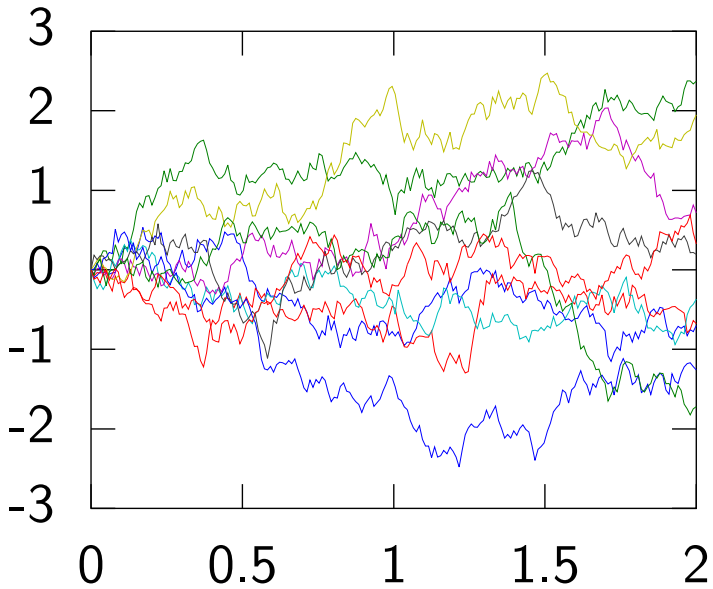
Covariance Functions

Where did this covariance matrix come from?

Markov Process

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

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



Simple Kalman Filter I

- We have state vector $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_q] \in \mathbb{R}^{T \times q}$ and if each state evolves independently we have
- $p(\mathbf{X}) = \prod_{i=1}^{q} p(\mathbf{x}_{:,i})$ $p(\mathbf{x}_{:,i}) = \mathcal{N}(\mathbf{x}_{:,i}|\mathbf{0}, \mathbf{K})$.
- We want to obtain outputs through:

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

Stacking and Kronecker Products I

- Represent with a 'stacked' system:

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

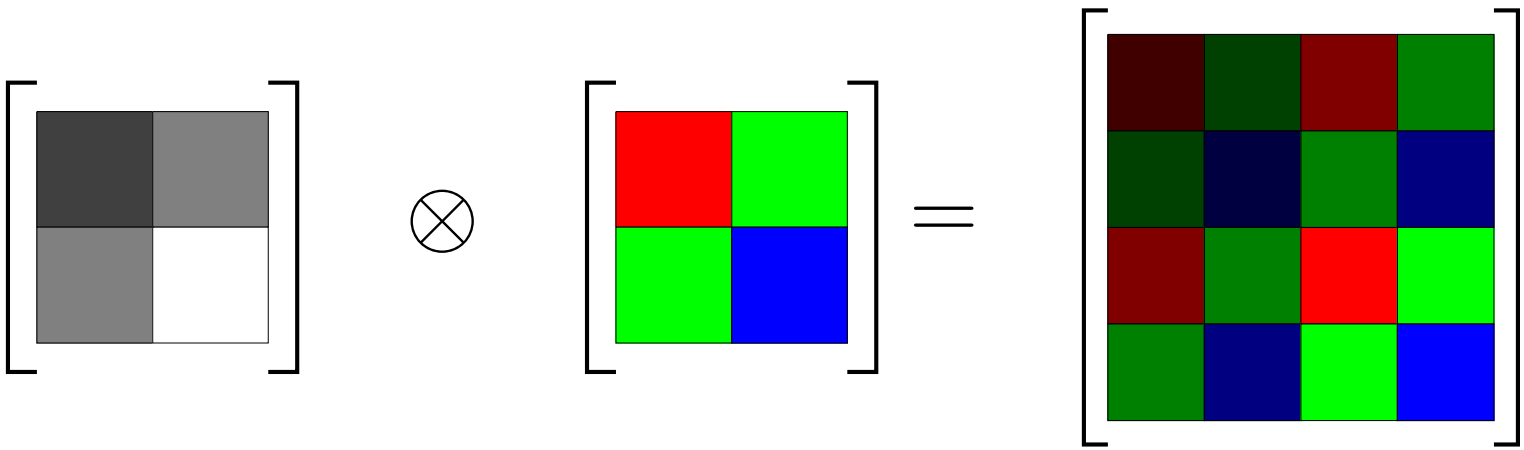
where the stacking is placing each column of $\mathbf{X}$ one on top of another as

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

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

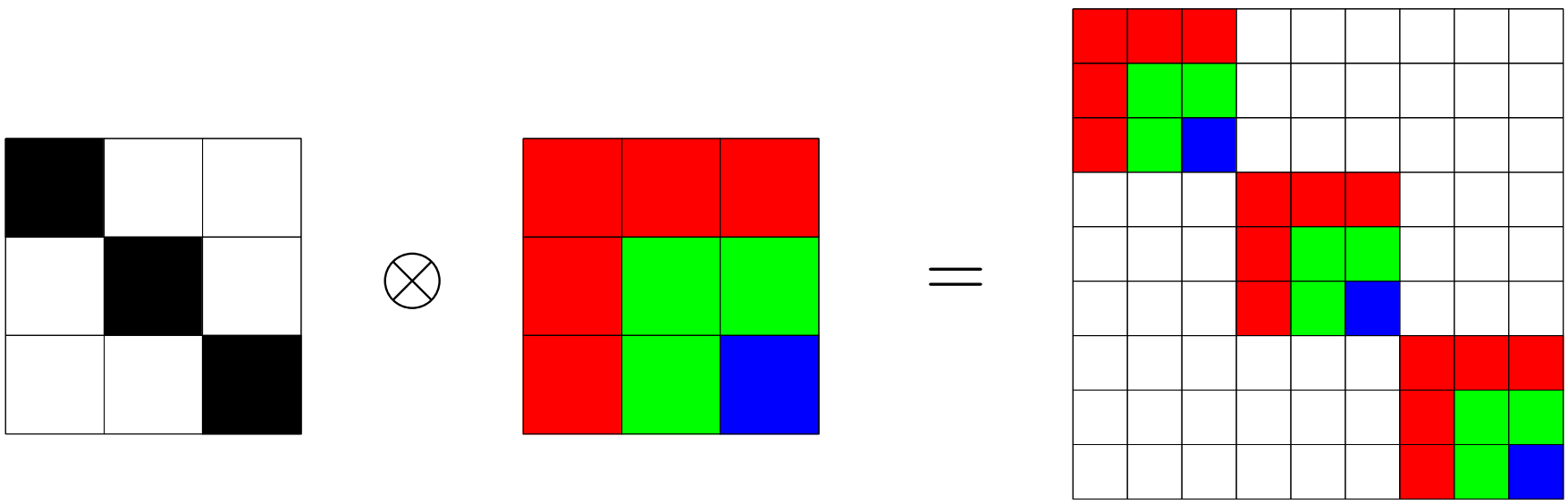
- Represent with a 'stacked' system:

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

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



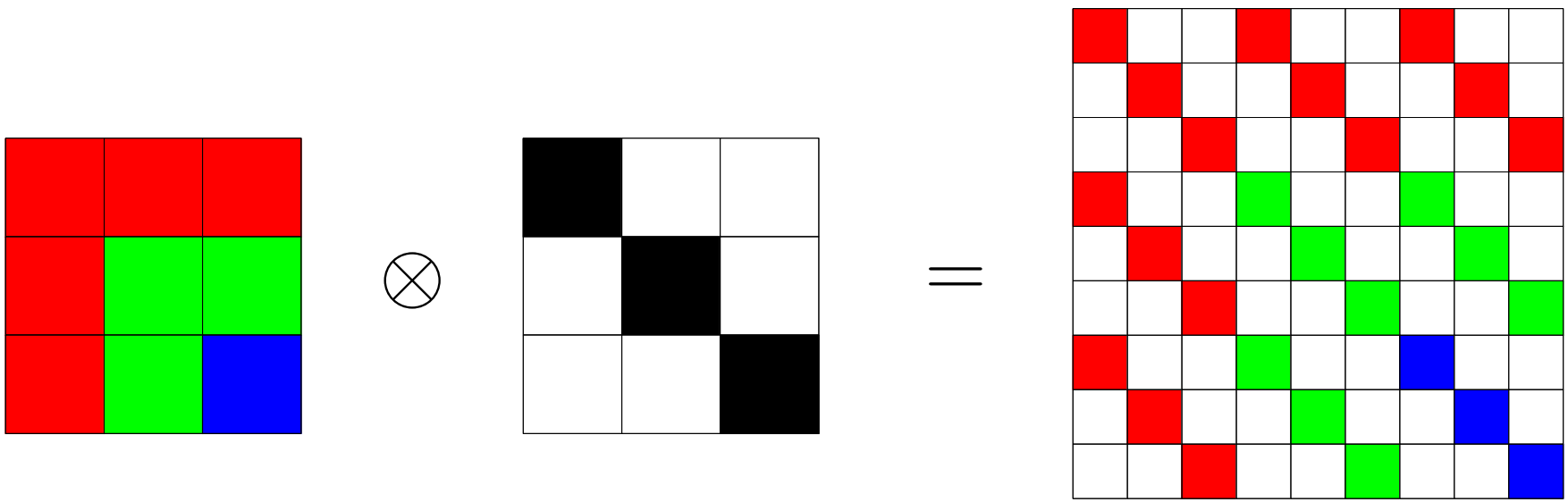
Two Ways of Stacking

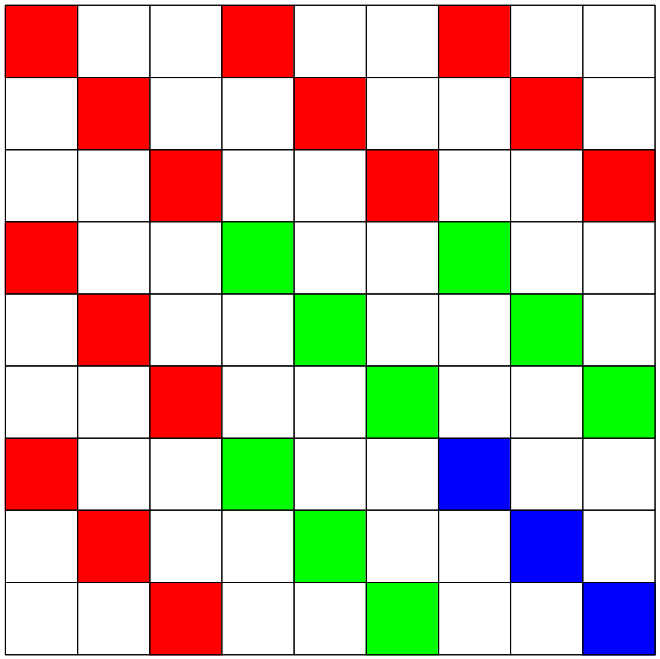
Can also stack as follows:

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

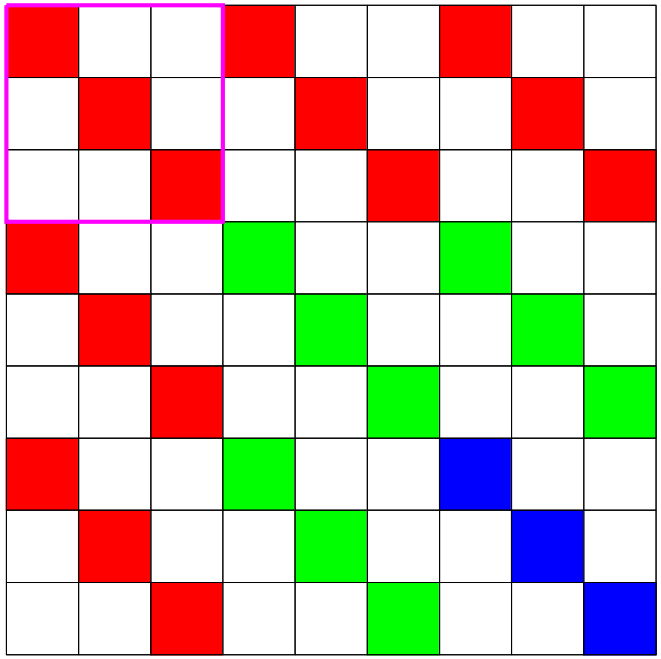
$$p(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}|\mathbf{0}, \mathbf{K} \otimes \mathbf{I}\right)$$

Row Stacking

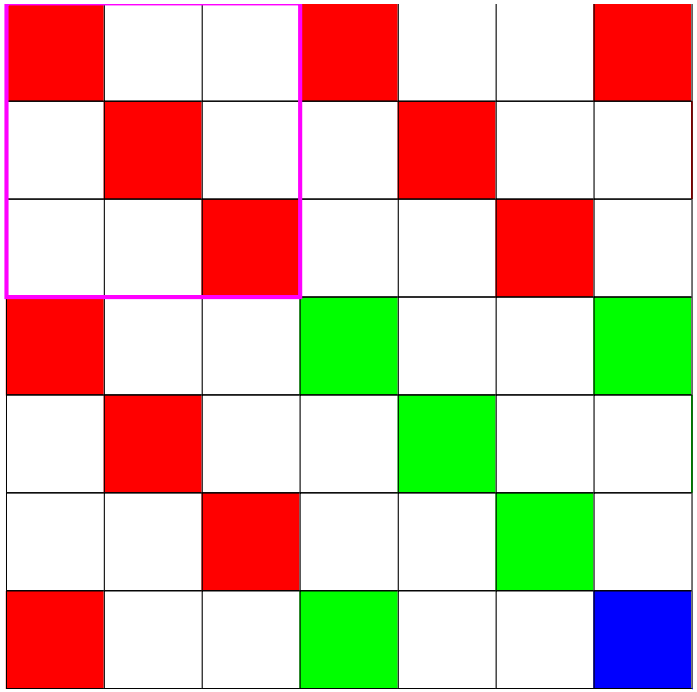




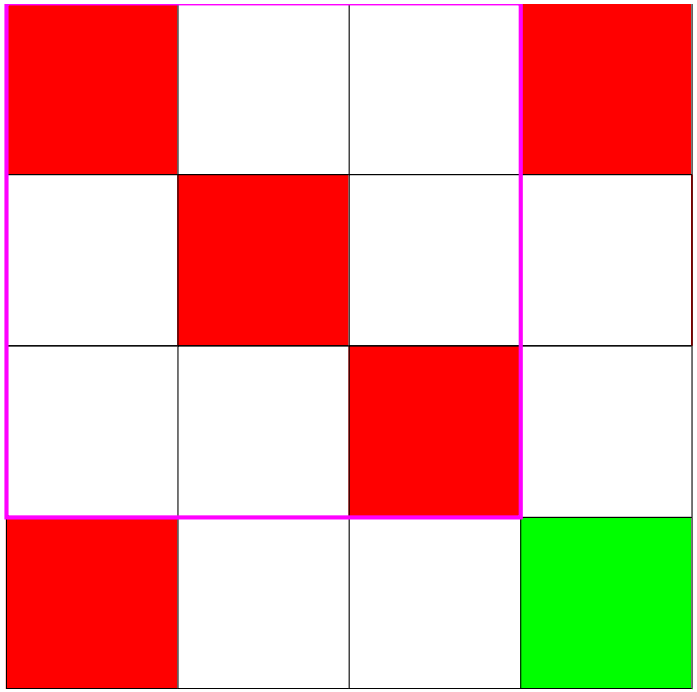
For this stacking the marginal distribution over the latent variables is given by the block diagonals.



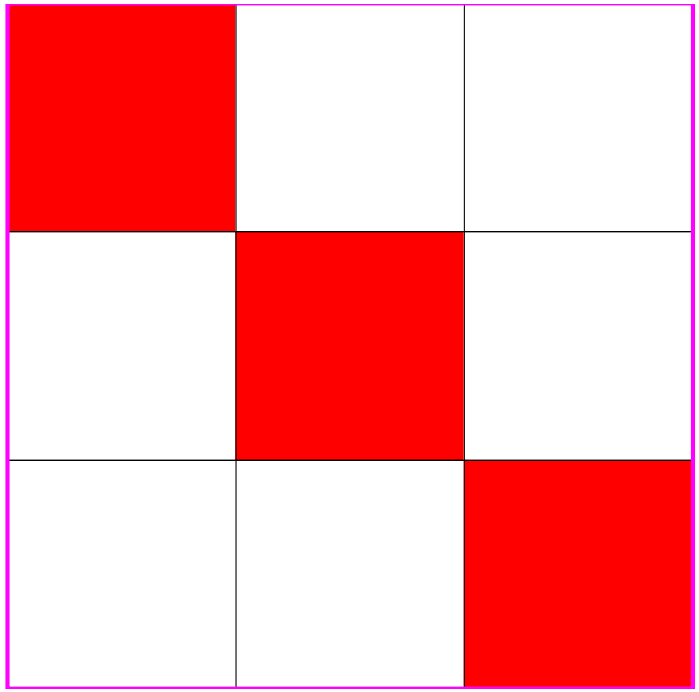
For this stacking the marginal distribution over the latent variables is given by the block diagonals.



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



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



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

Observed Process

If we relate the observations to the data as follows:

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

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

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

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

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\mathcal{N}\left(\mathbf{y}|\mathbf{0},\mathbf{K}\right) = \frac{1}{\left(2\pi\right)^{\frac{n}{2}}|\mathbf{K}|}\exp\left(-\frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}\right)$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\mathcal{N}\left(\mathbf{y}|\mathbf{0},\mathbf{K}\right) = \frac{1}{\left(2\pi\right)^{\frac{n}{2}}|\mathbf{K}|}\exp\left(-\frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}\right)$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

$$\log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}\right) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Learning Covariance Parameters

Can we determine covariance parameters from the data?

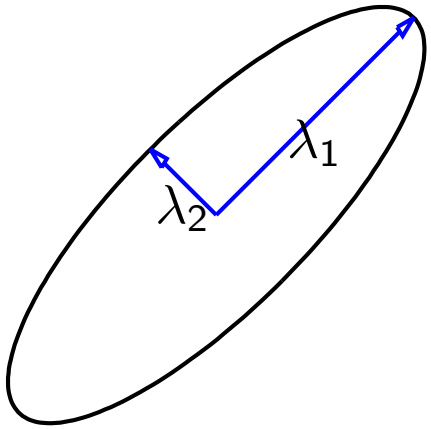
$$E(\boldsymbol{\theta}) = \frac{1}{2}\log|\mathbf{K}| + \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

The parameters are *inside* the covariance function (matrix).

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta})$$

Eigendecomposition of Covariance

$$\mathbf{K} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^\top$$

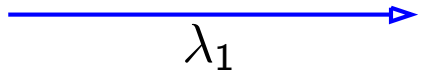


where $\mathbf{\Lambda}$ is a *diagonal* matrix and $\mathbf{R}^\top\mathbf{R} = \mathbf{I}$.

Useful representation since $|\mathbf{K}| = \left|\mathbf{\Lambda}^2\right| = |\mathbf{\Lambda}|^2$.

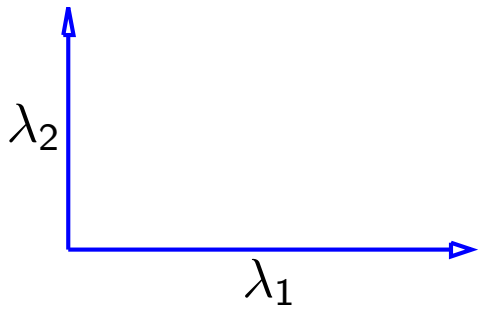
Capacity control: $\log |\mathbf{K}|$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



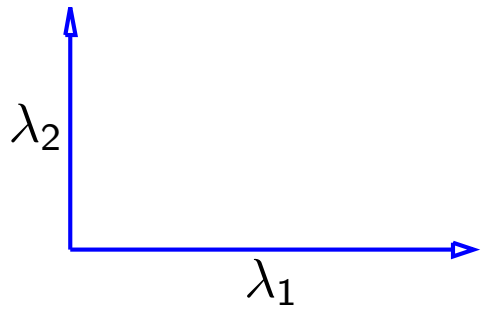
Capacity control: $\log |\mathbf{K}|$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



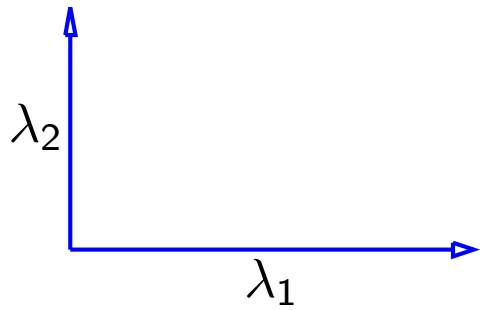
Capacity control: $\log |\mathbf{K}|$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



Capacity control: $\log |\mathbf{K}|$

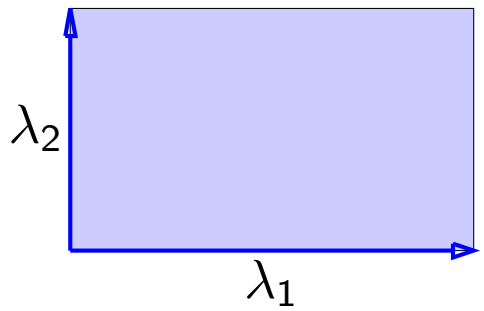
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2$$

Capacity control: $\log |\mathbf{K}|$

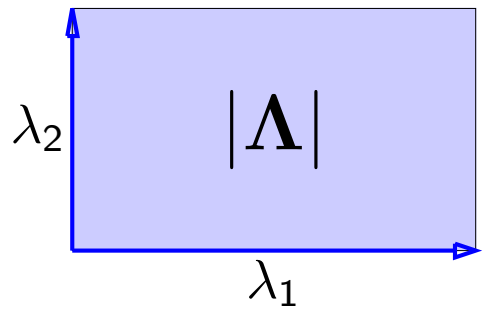
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2$$

Capacity control: $\log |\mathbf{K}|$

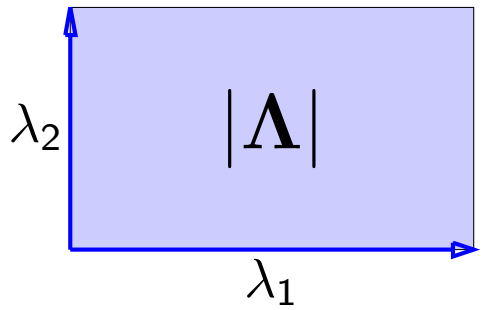
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2$$

Capacity control: $\log |\mathbf{K}|$

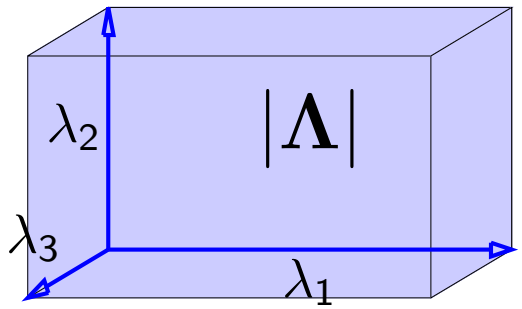
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2$$

Capacity control: $\log |\mathbf{K}|$

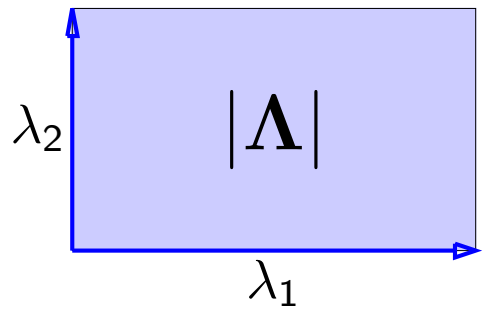
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2 \lambda_3$$

Capacity control: $\log |\mathbf{K}|$

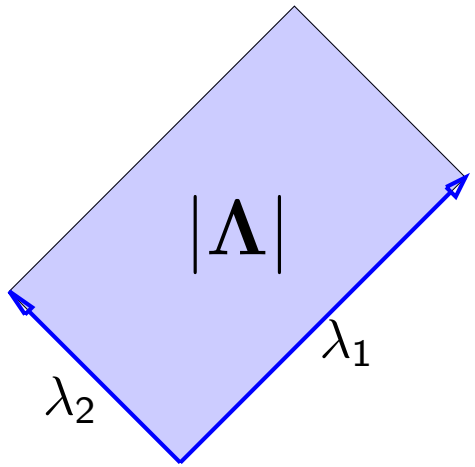
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



$$|\mathbf{\Lambda}| = \lambda_1 \lambda_2$$

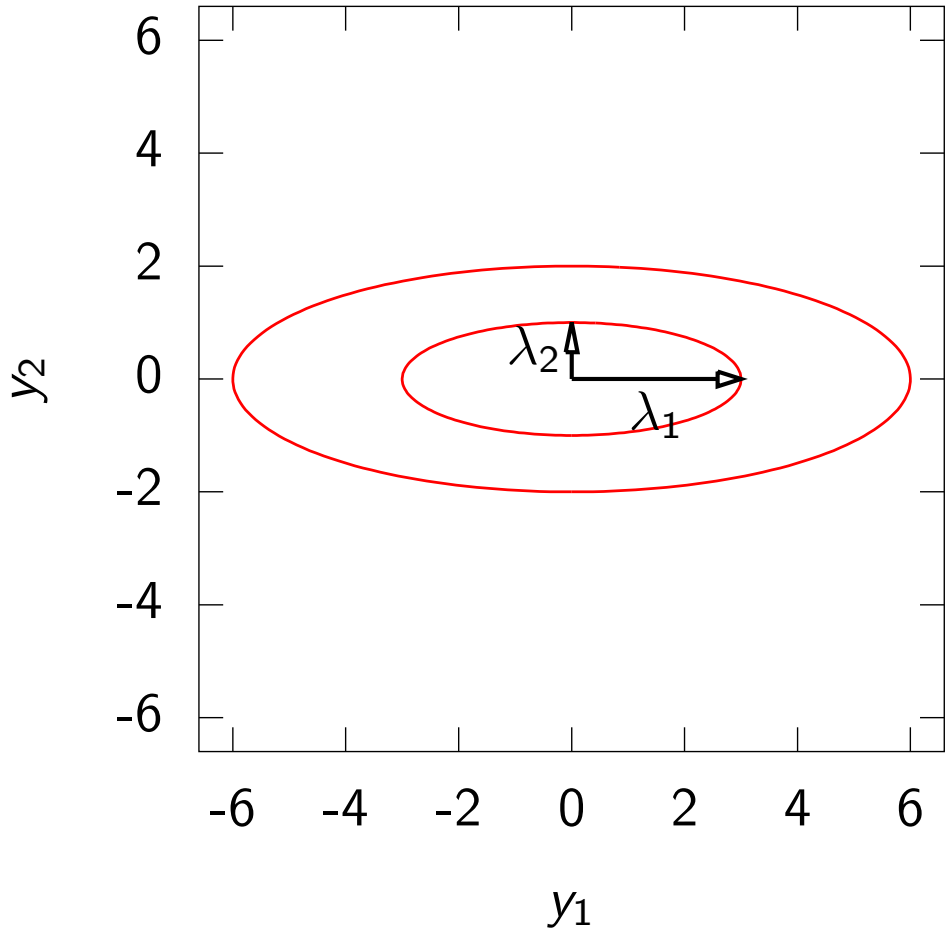
Capacity control: $\log |\mathbf{K}|$

$$\mathbf{R}\mathbf{\Lambda} = \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{bmatrix}$$

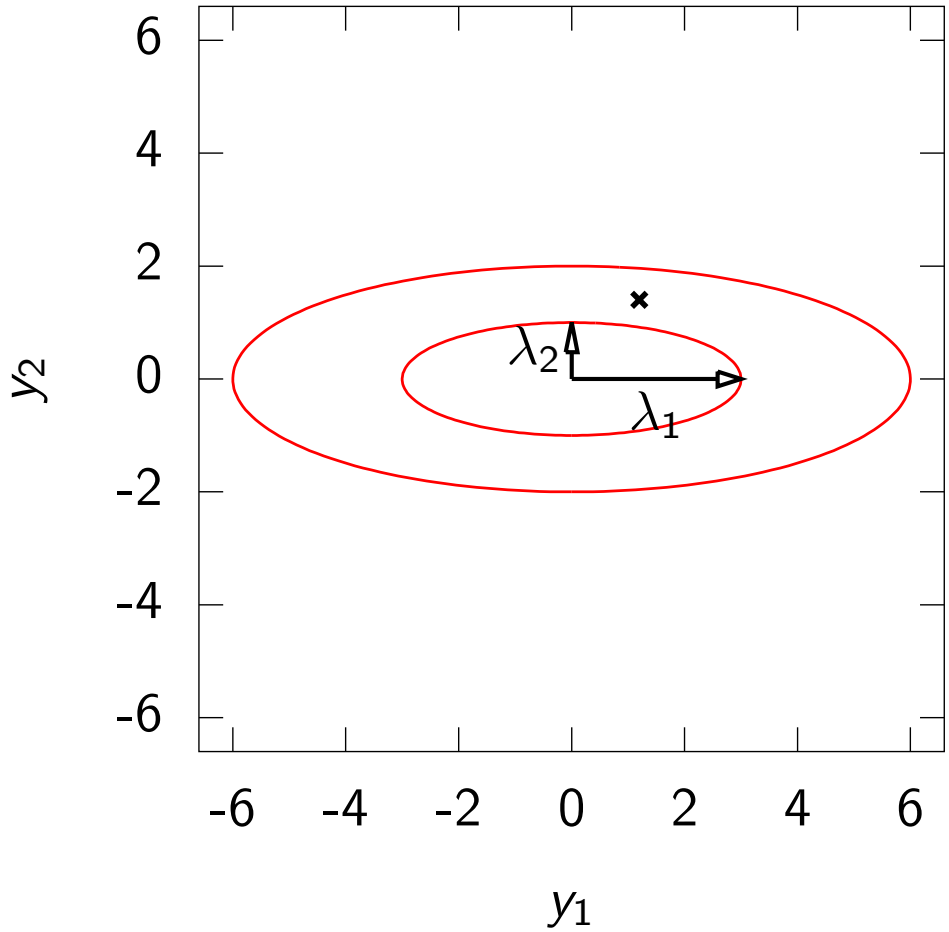


$$|\mathbf{R}\mathbf{\Lambda}| = \lambda_1\lambda_2$$

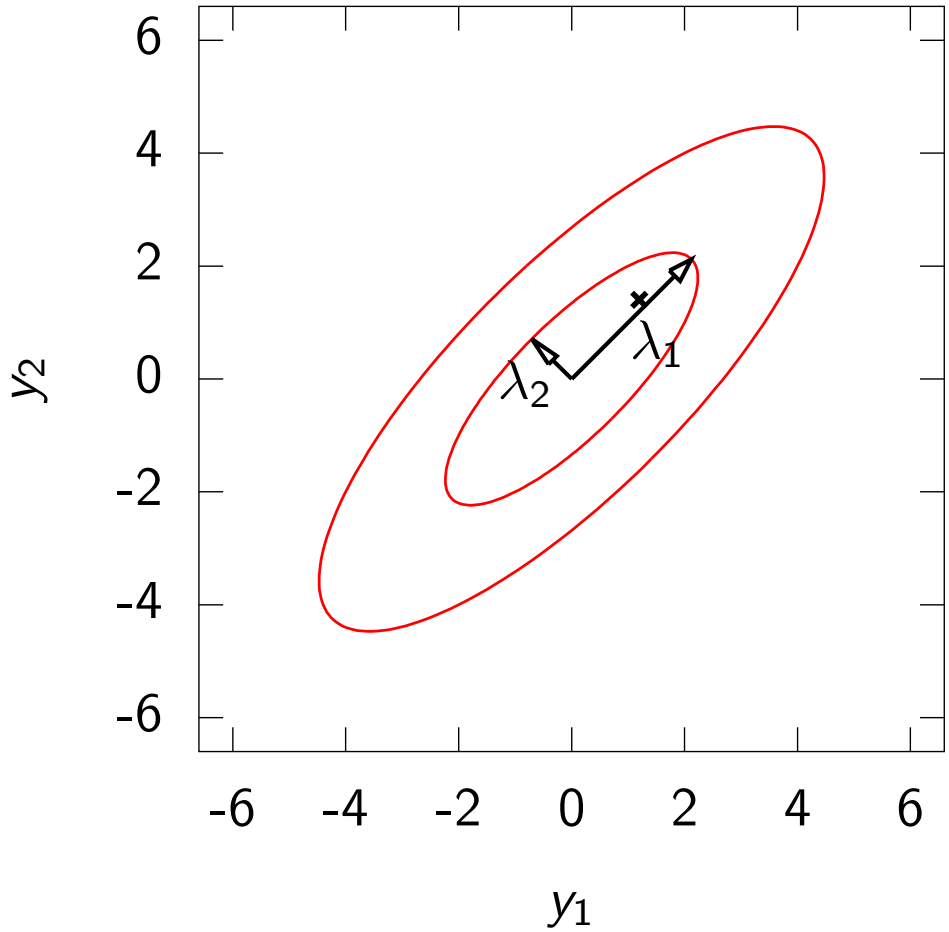
Data Fit: $\frac{\mathbf{y}^{-1}\mathbf{K}^{-1}\mathbf{y}}{2}$



Data Fit: $\frac{\mathbf{y}^{-1}\mathbf{K}^{-1}\mathbf{y}}{2}$

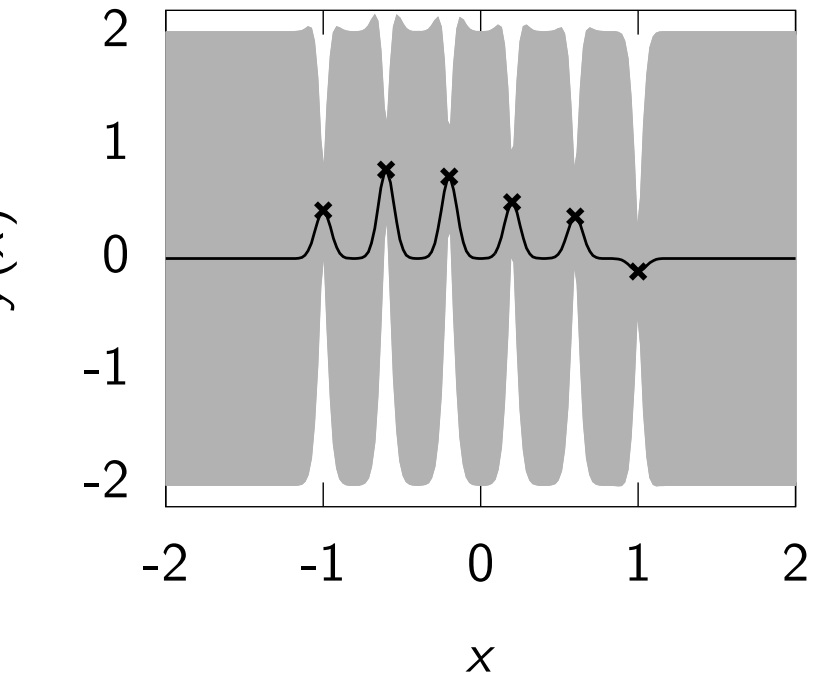


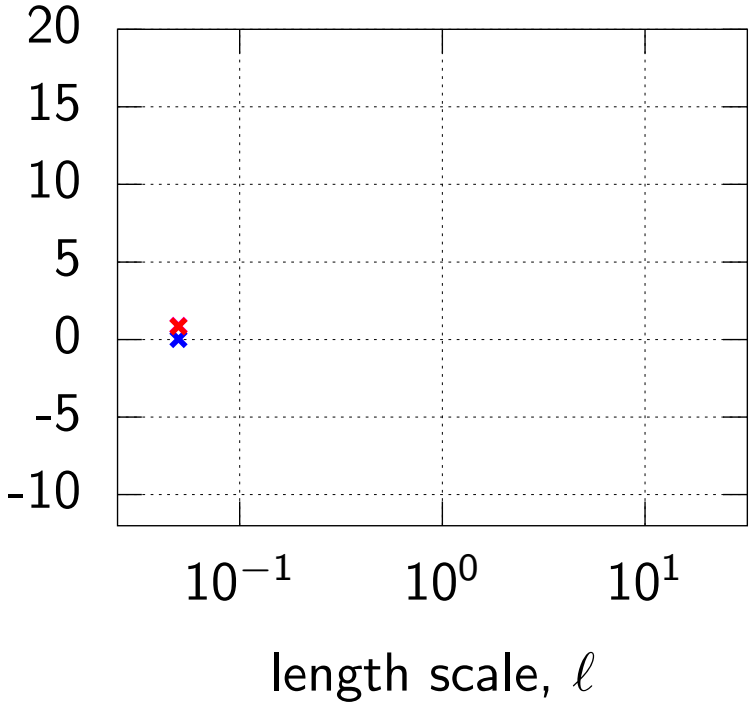
Data Fit: $\frac{\mathbf{y}^{-1}\mathbf{K}^{-1}\mathbf{y}}{2}$



Learning Covariance Parameters

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

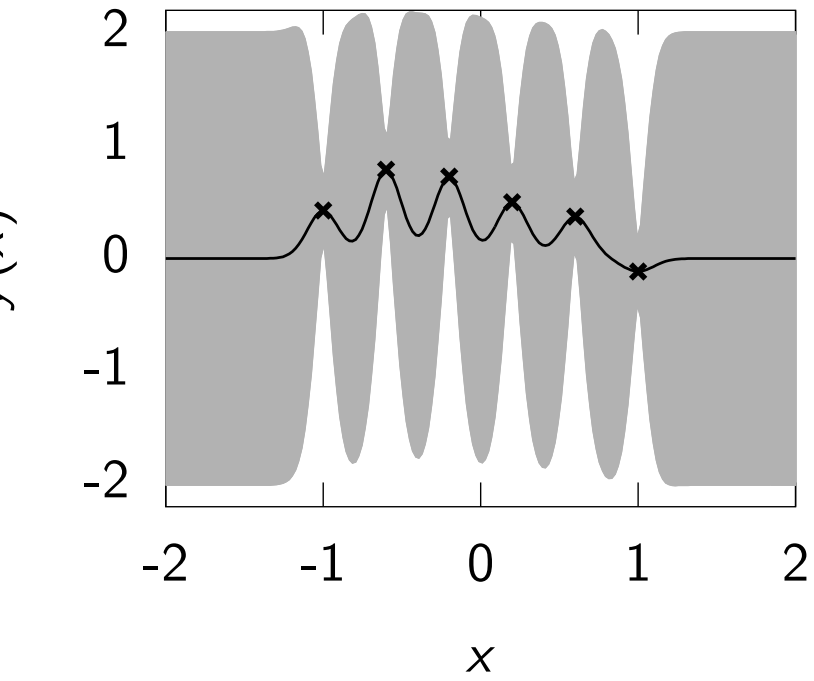


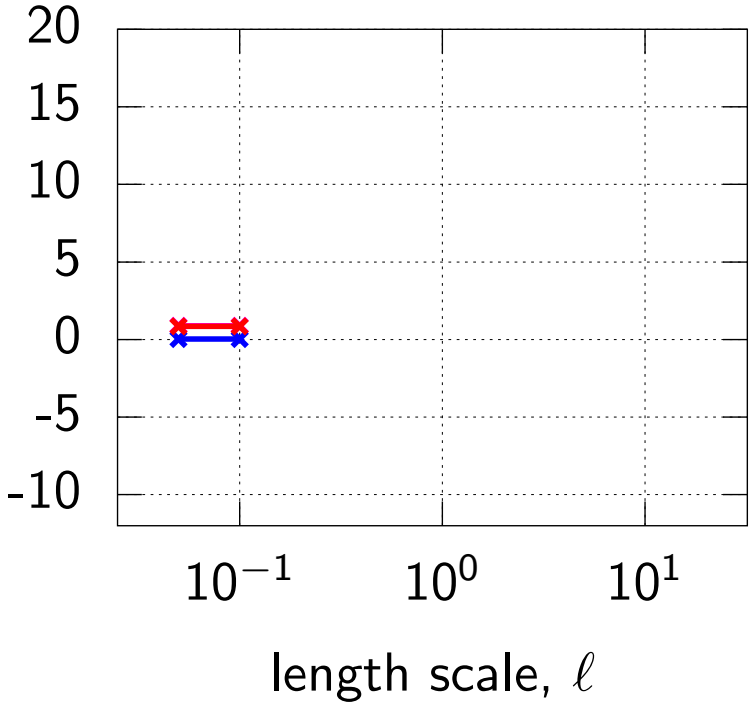


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

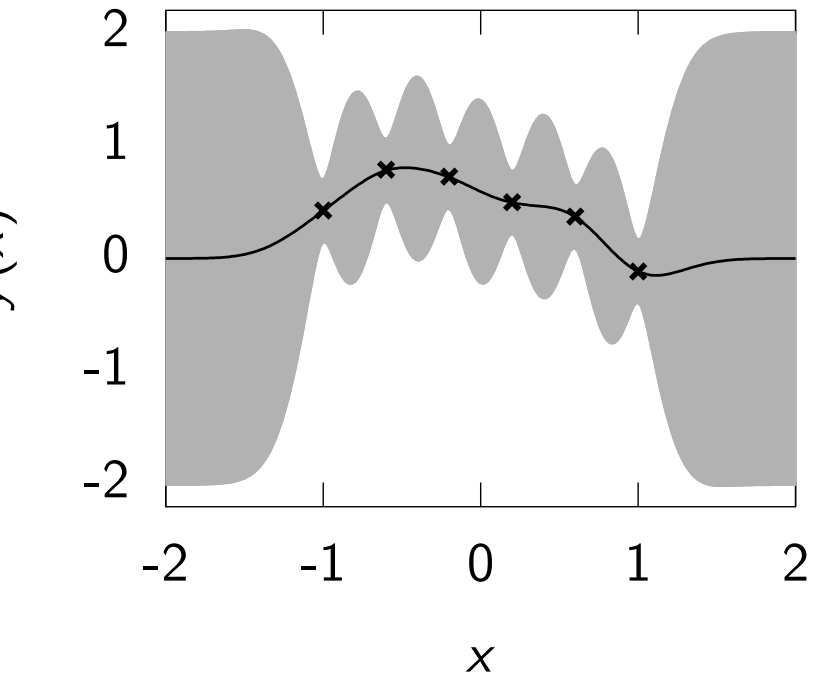


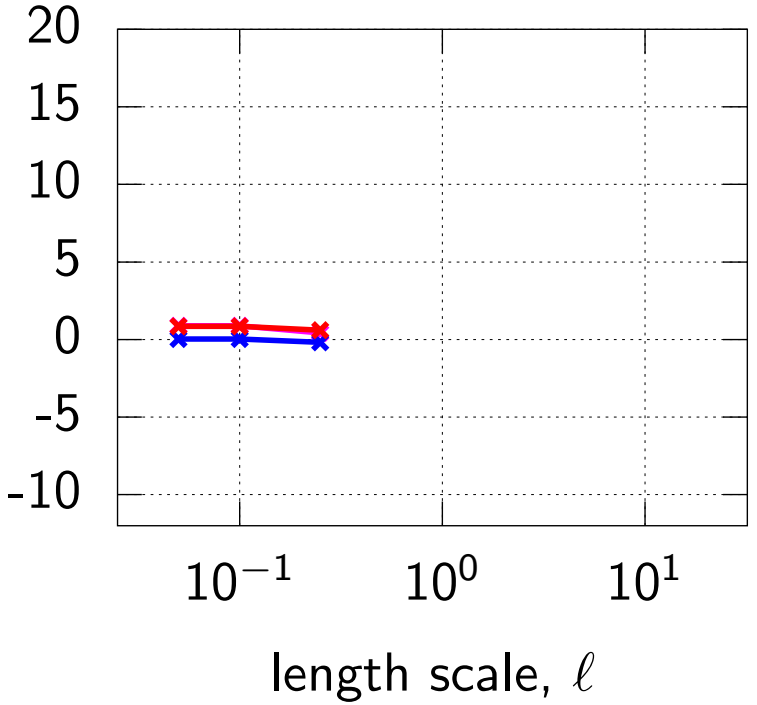


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

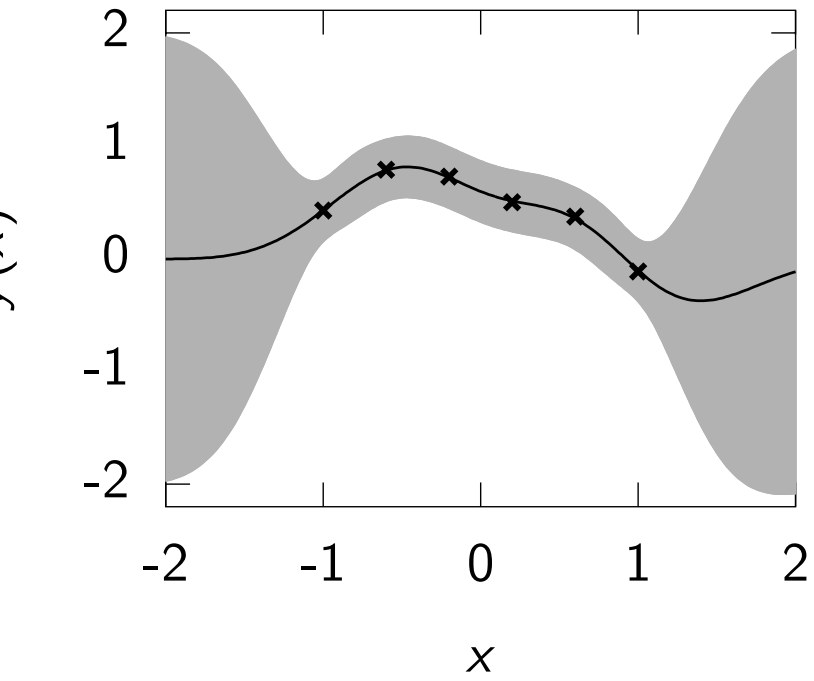


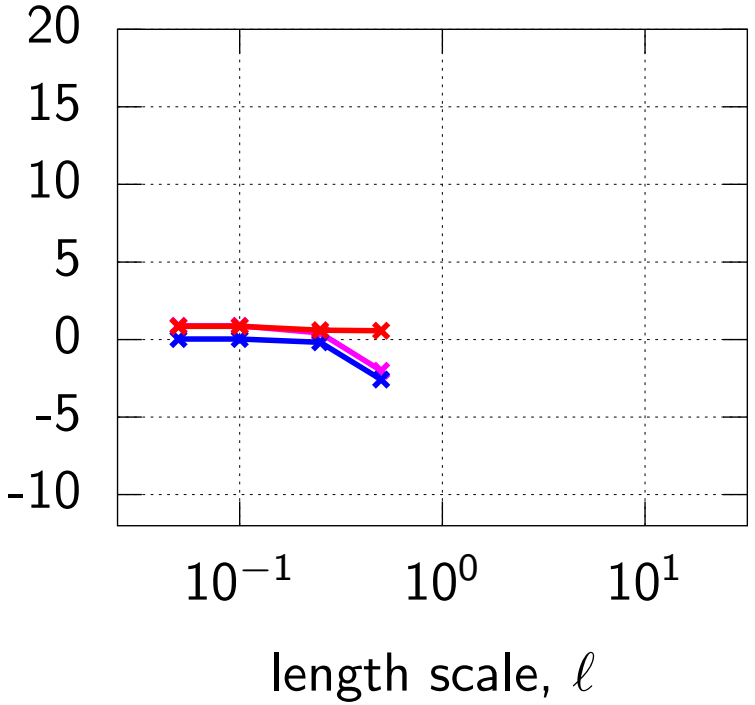


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

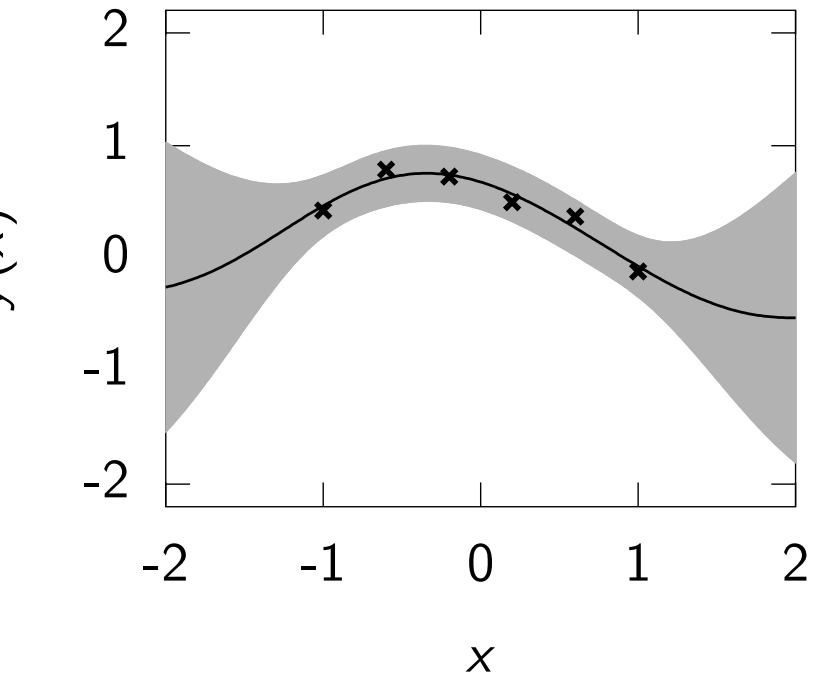


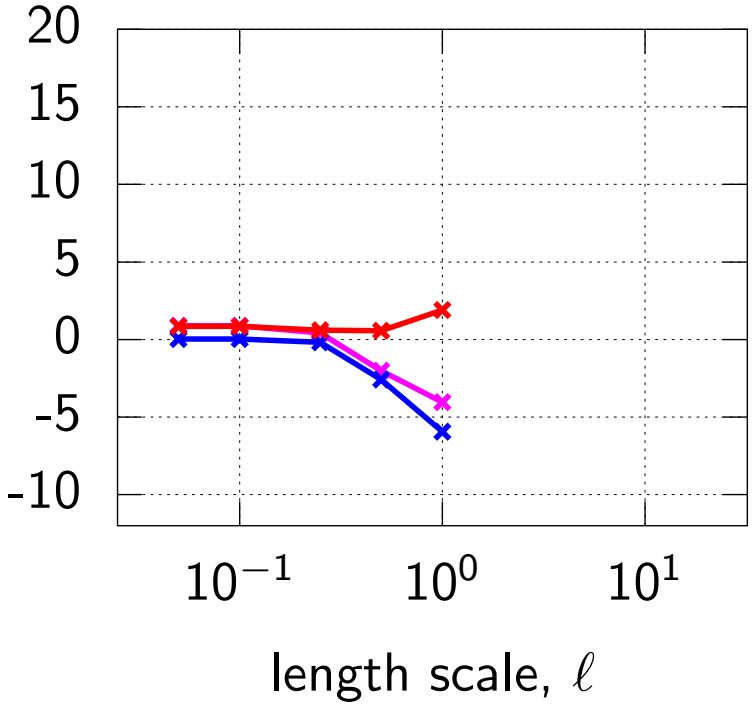


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

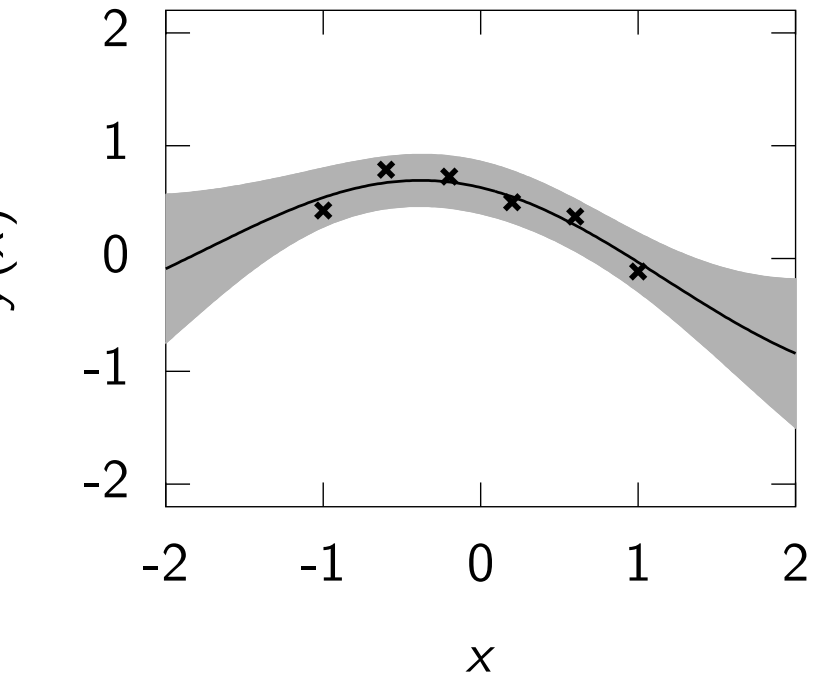


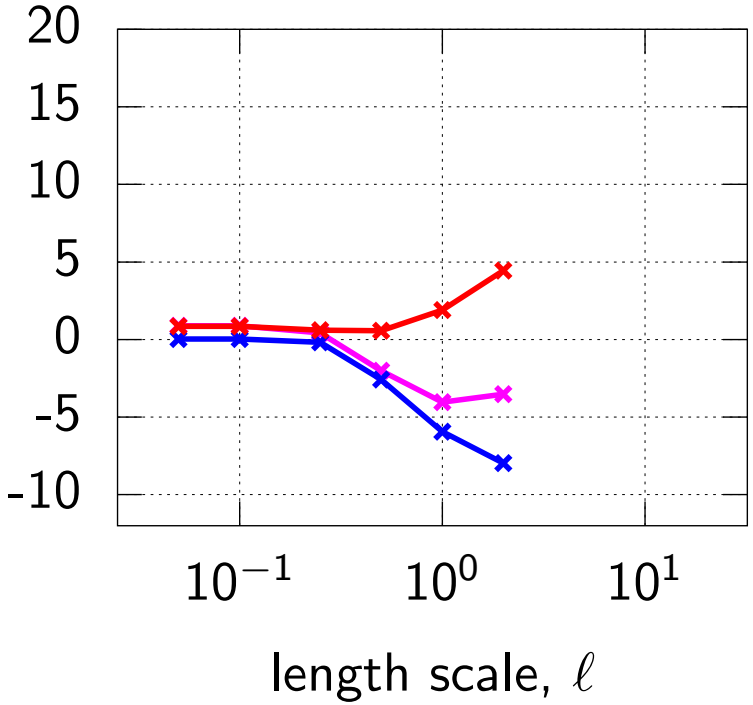


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

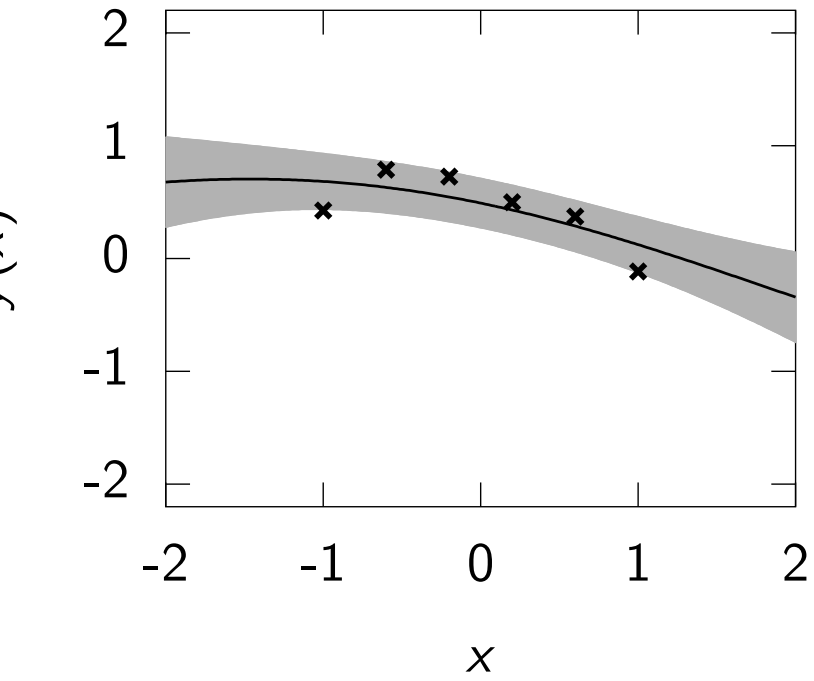


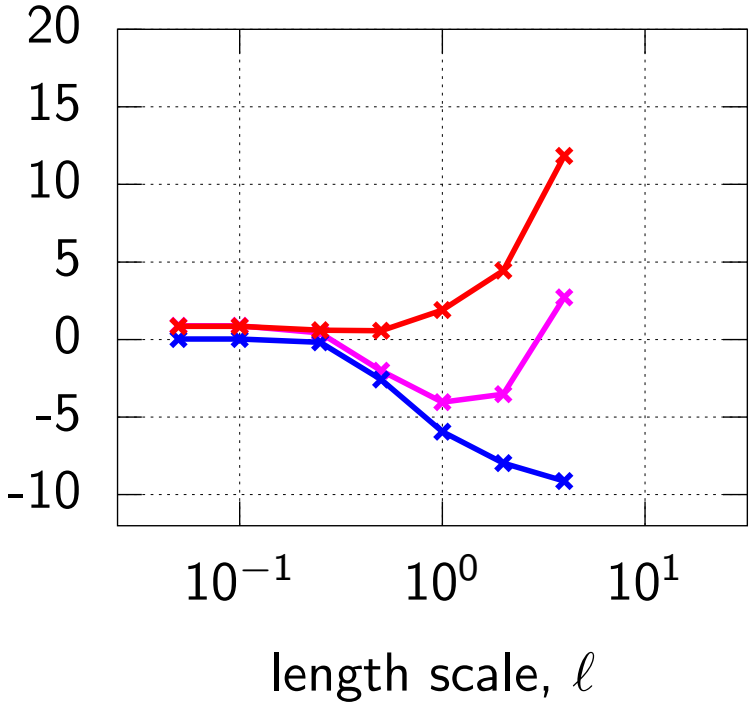


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

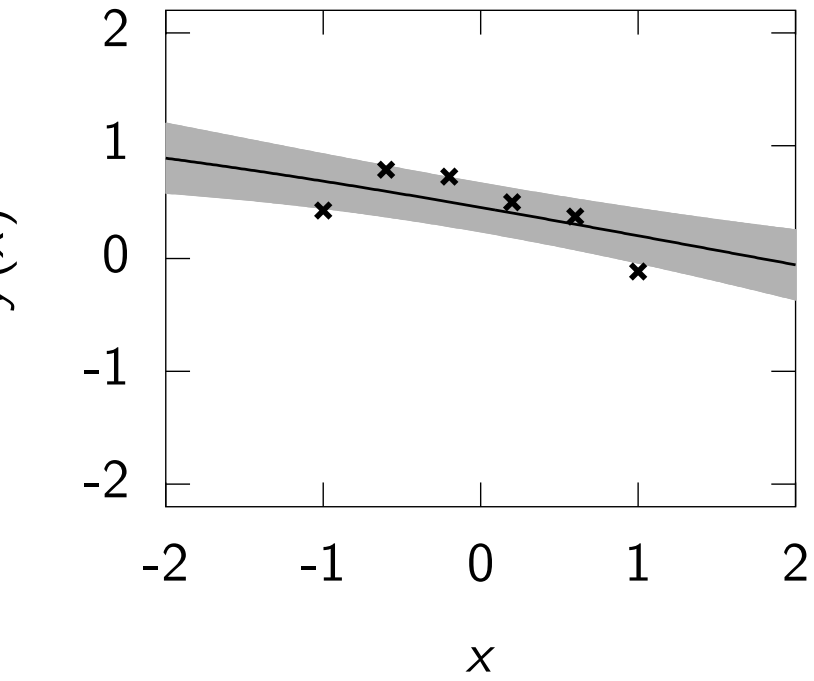


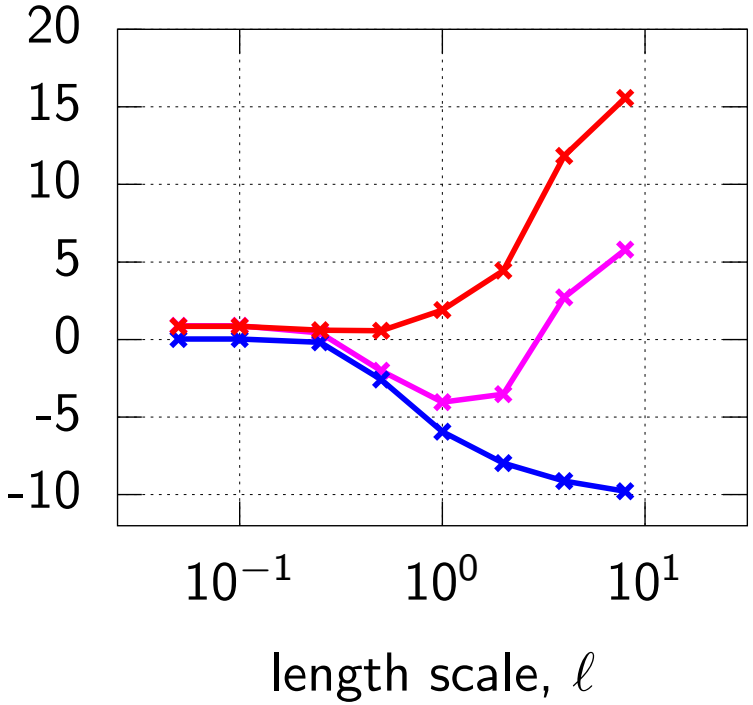


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

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

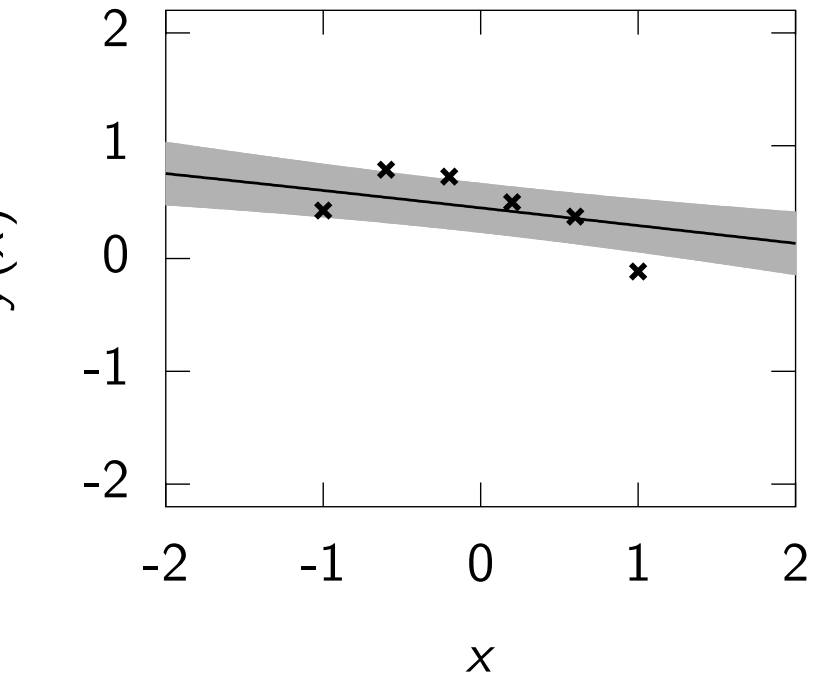


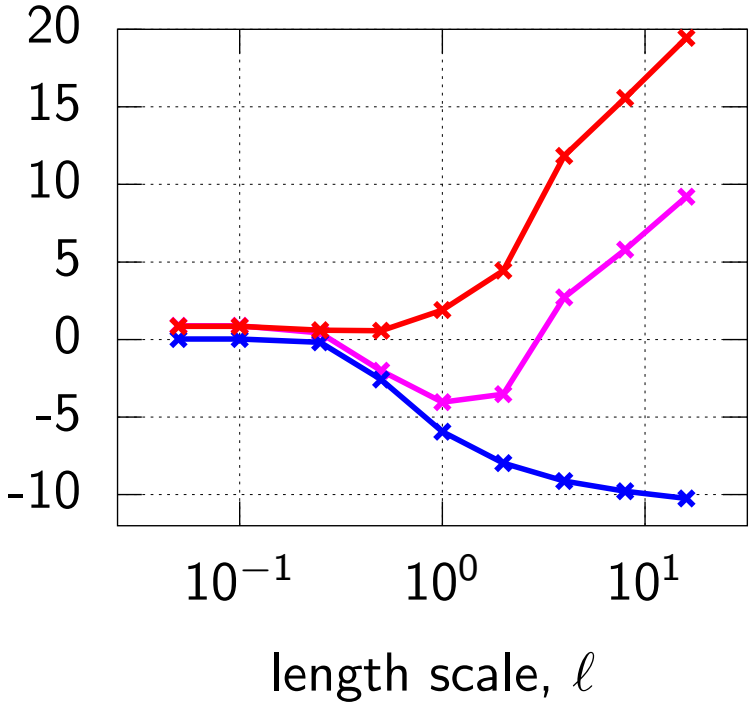


$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Covariance Parameters

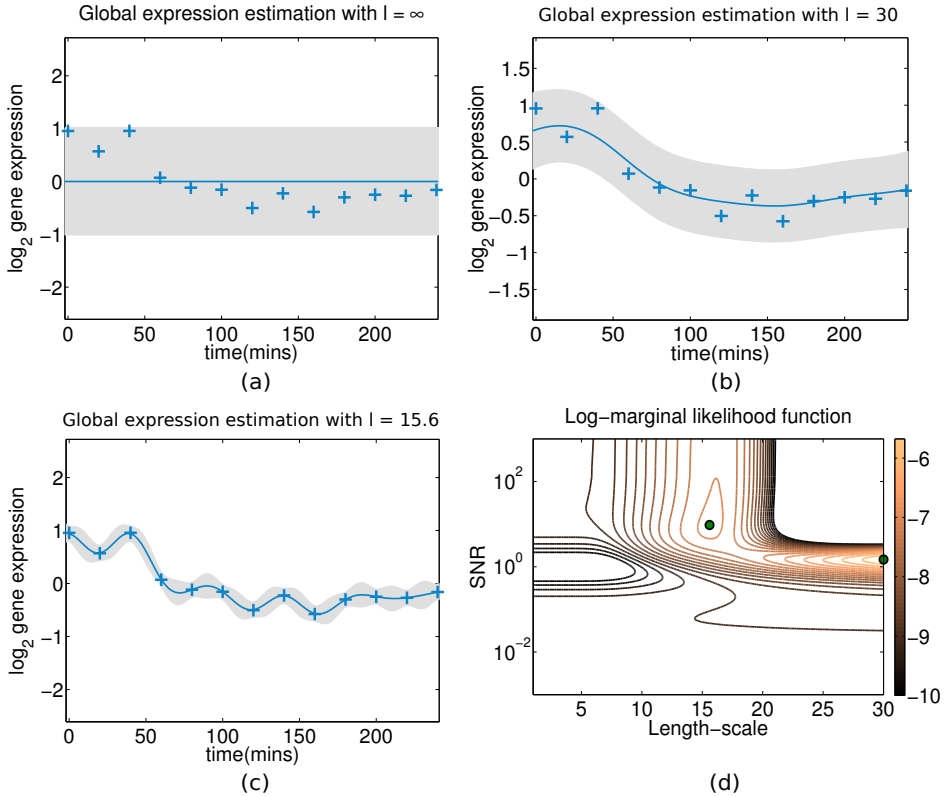
Can we determine length scales and noise levels from the data?





$$E(\boldsymbol{\theta}) = \frac{1}{2}\left|\mathbf{K}\right| + \frac{\mathbf{y}^{\top}\mathbf{K}^{-1}\mathbf{y}}{2}$$

Gene Expression Example



Data from Della Gatta et al. (2008). Figure from Kalaitzis and Lawrence (2011).

Efficient inference in matrix-variate Gaussian models with iid observation noise

Oliver Stegle[1]
Max Planck Institutes
Tübingen, Germany
stegle@tuebingen.mpg.de

Christoph Lippert[1]
Max Planck Institutes
Tübingen, Germany
clippert@tuebingen.mpg.de

Joris Mooij
Institute for Computing and Information Sciences
Radboud University
Nijmegen, The Netherlands

Neil Lawrence
Department of Computer Science
University of Sheffield
Sheffield, UK

(B

et al., 2012; Stegle et al., 2011)

Kernels for Vector Valued Outputs: A Review

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.

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

  where k has $\mathbf{x}$ and k_T has n as inputs.
- Can think of multiple output covariance functions as covariances with augmented input.
- Alongside $\mathbf{x}$ we also input the d 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_d is expressed as a linear sum

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

  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

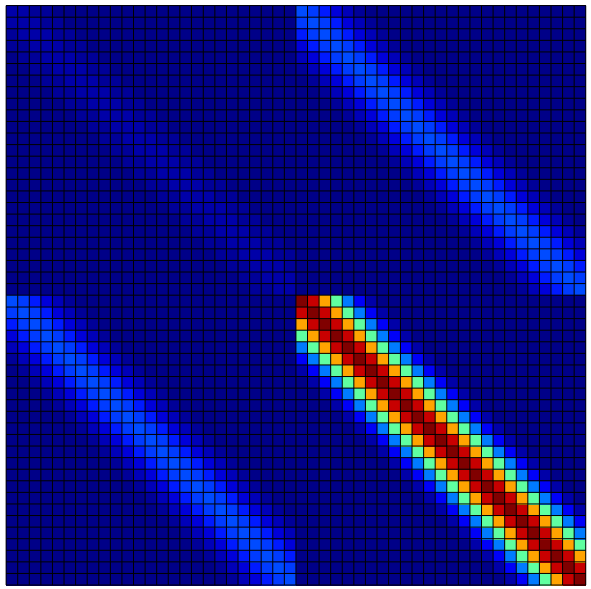
- If outputs are noise-free, maximum likelihood is equivalent to independent fits of $\mathbf{B}$ and $k(\mathbf{x}, \mathbf{x}')$ (Helterbrand and Cressie, 1994).
- In geostatistics this is known as autokrigeability (Wackernagel, 2003).
- In multitask learning its the cancellation of intertask transfer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

Intrinsic Coregionalization Model

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

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

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

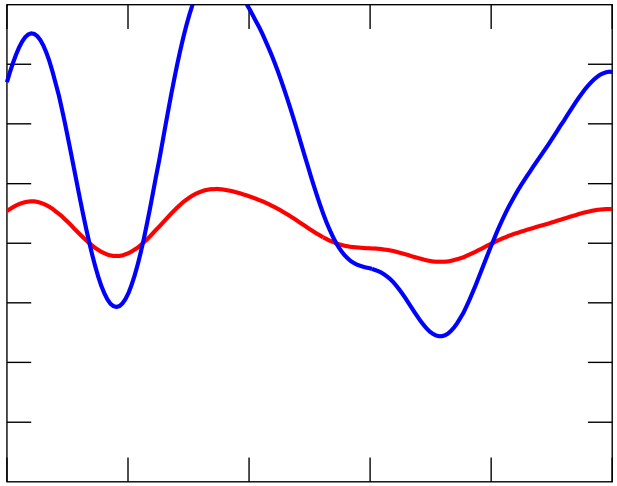


Intrinsic Coregionalization Model

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

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

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

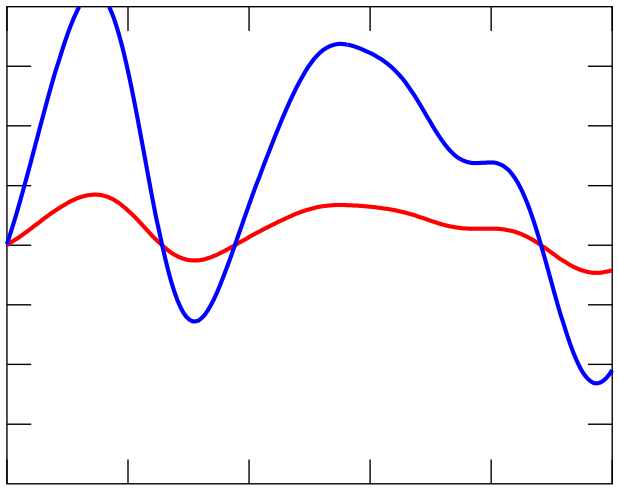


Intrinsic Coregionalization Model

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

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

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

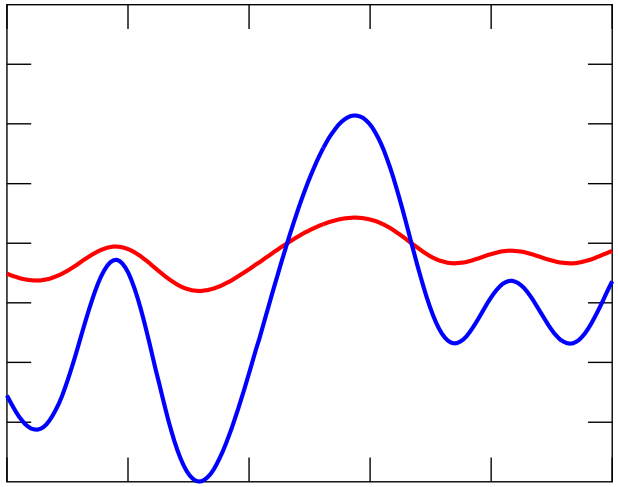


Intrinsic Coregionalization Model

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

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

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

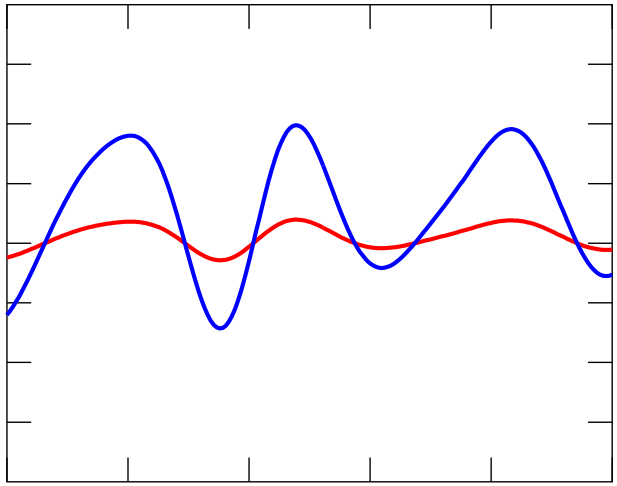


Intrinsic Coregionalization Model

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

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

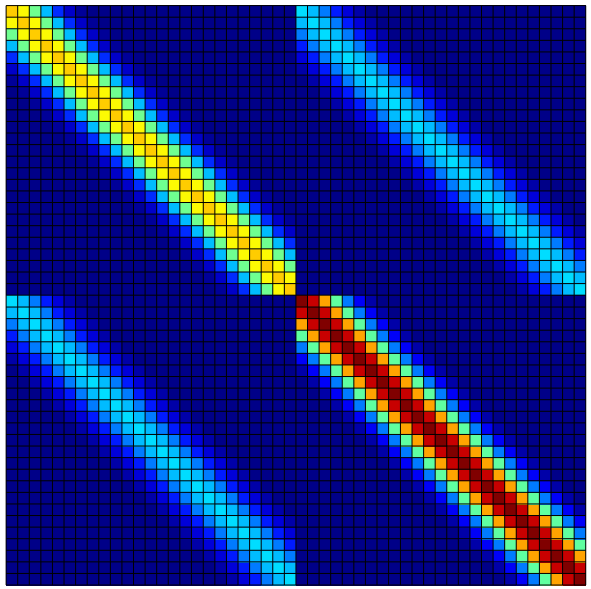
$$\mathbf{B} = \begin{bmatrix} 1 & 5 \\ 5 & 25 \end{bmatrix}$$



Intrinsic Coregionalization Model

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

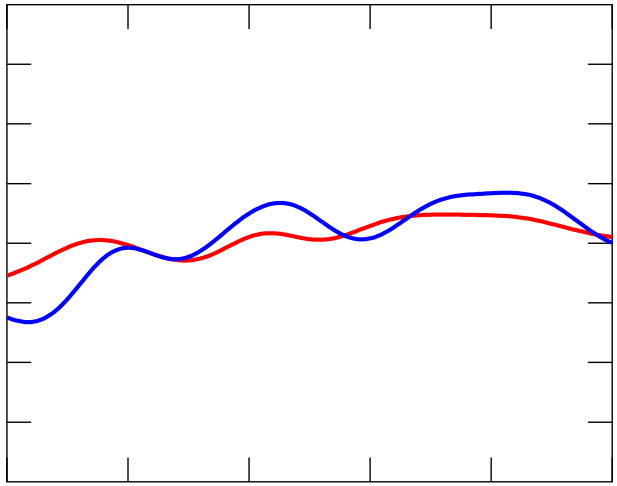
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

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

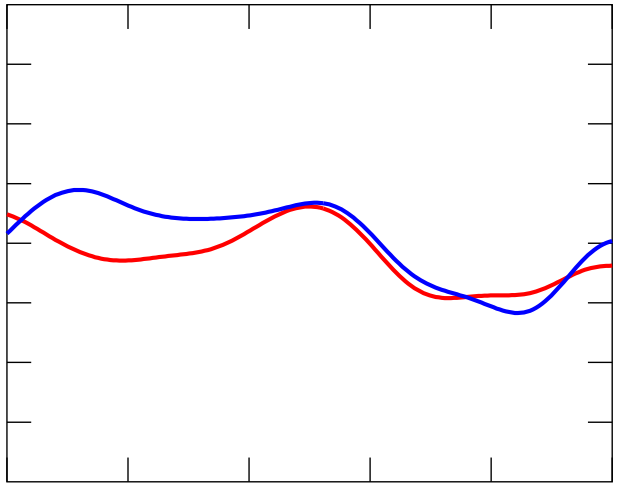
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

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

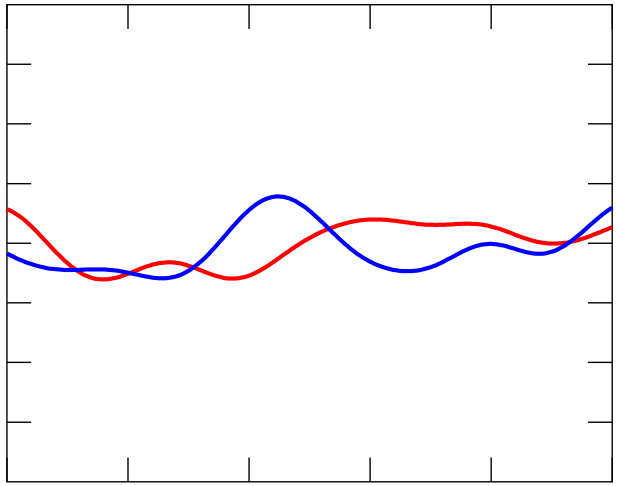
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



Intrinsic Coregionalization Model

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

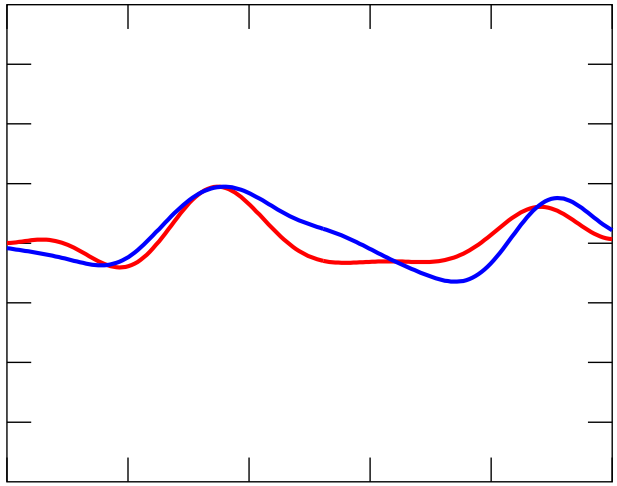
$$\mathbf{B} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1.5 \end{bmatrix}$$



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

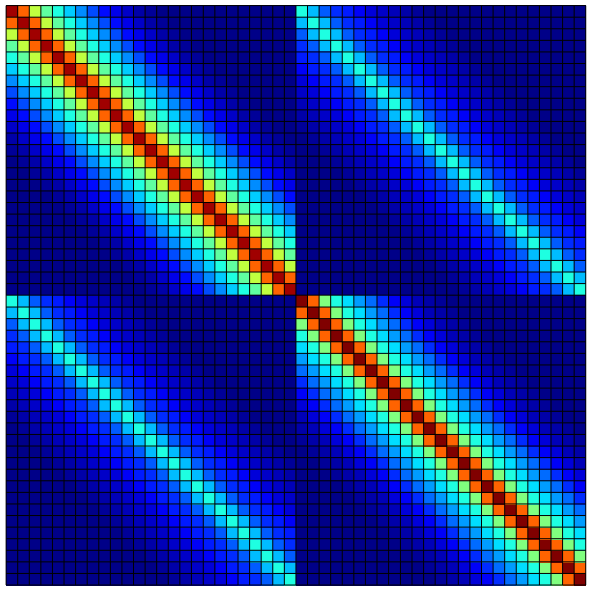
$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X}, \mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X}, \mathbf{X})$$

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

$$\ell_1 = 1$$

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

$$\ell_2 = 0.2$$



LMC Samples

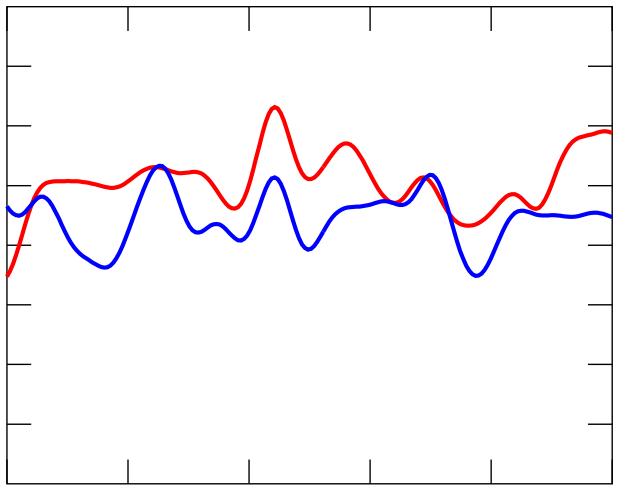
$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X}, \mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X}, \mathbf{X})$$

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

$$\ell_1 = 1$$

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

$$\ell_2 = 0.2$$



LMC Samples

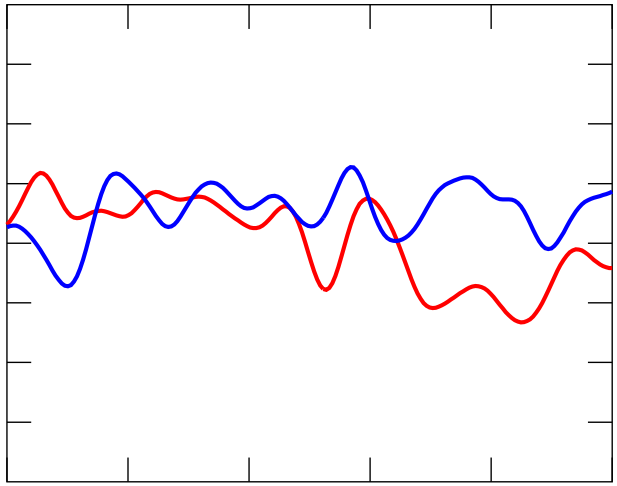
$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X}, \mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X}, \mathbf{X})$$

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

$$\ell_1 = 1$$

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

$$\ell_2 = 0.2$$



LMC Samples

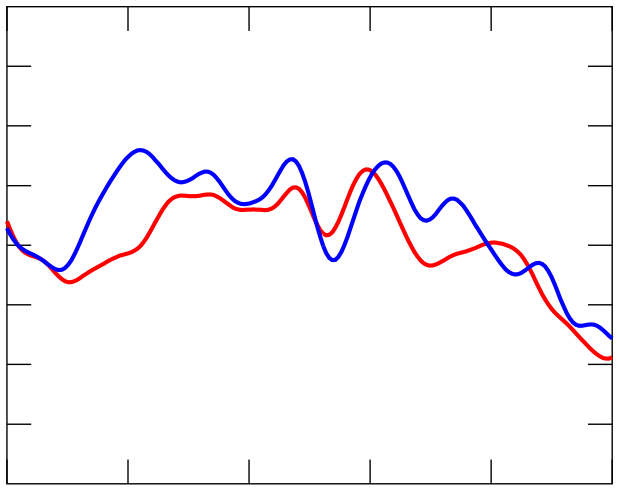
$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X}, \mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X}, \mathbf{X})$$

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

$$\ell_1 = 1$$

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

$$\ell_2 = 0.2$$



LMC Samples

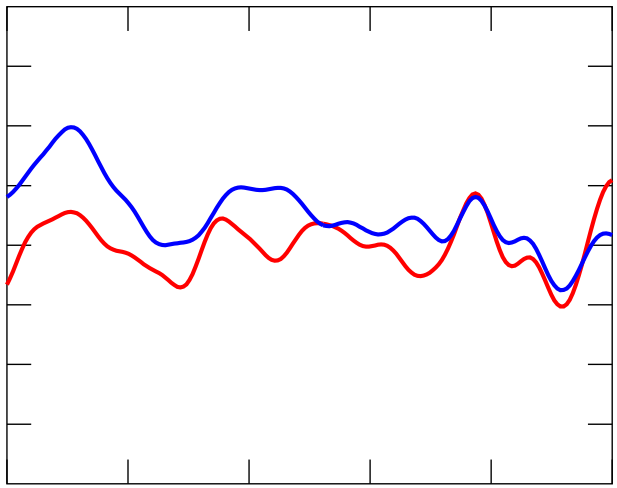
$$\mathbf{K}(\mathbf{X}, \mathbf{X}) = \mathbf{B}_1 \otimes k_1(\mathbf{X}, \mathbf{X}) + \mathbf{B}_2 \otimes k_2(\mathbf{X}, \mathbf{X})$$

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

$$\ell_1 = 1$$

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

$$\ell_2 = 0.2$$



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

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

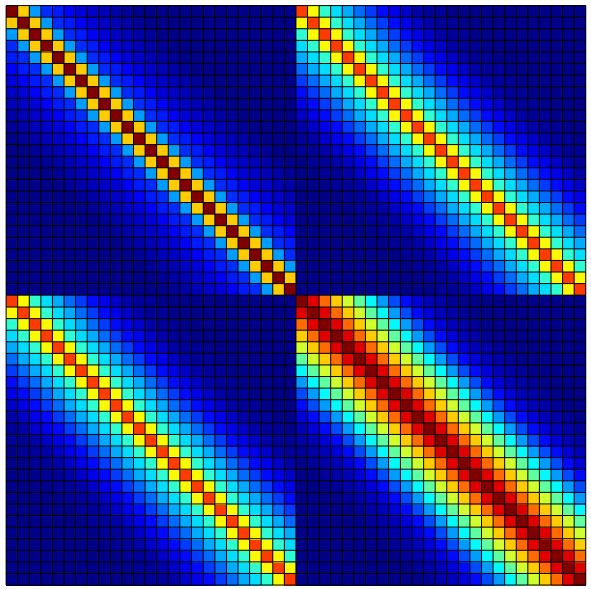
- Like the Kalman filter, but each latent function has a *different* covariance.
- Authors suggest using an exponentiated quadratic characteristic length-scale for each input dimension.

Semiparametric Latent Factor Model Samples

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

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

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

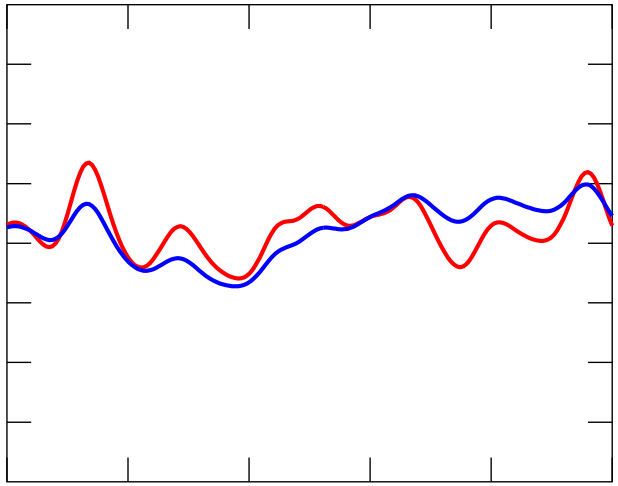


Semiparametric Latent Factor Model Samples

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

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

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

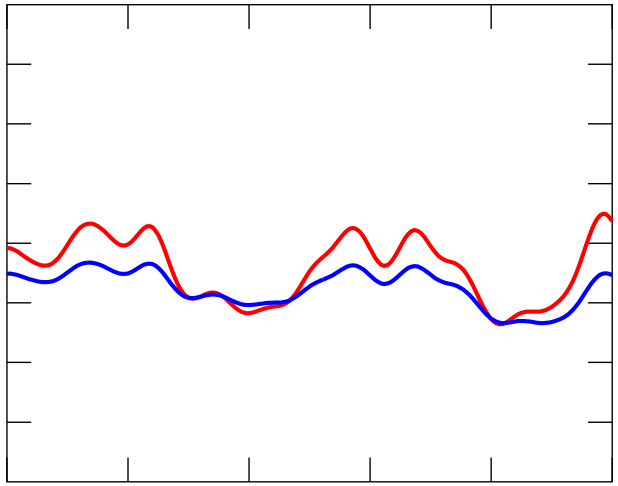


Semiparametric Latent Factor Model Samples

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

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

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

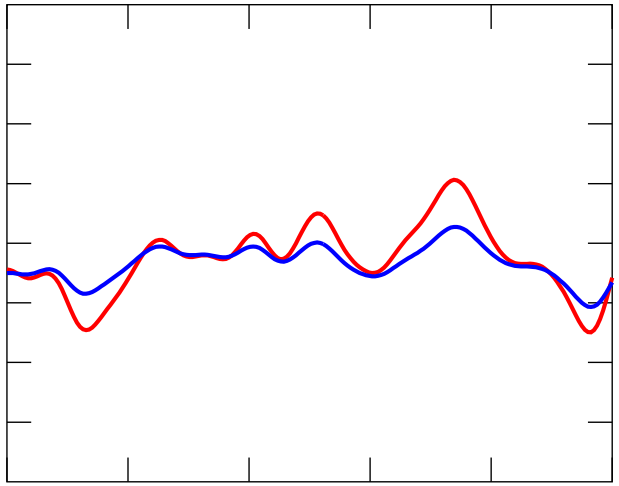


Semiparametric Latent Factor Model Samples

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

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

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

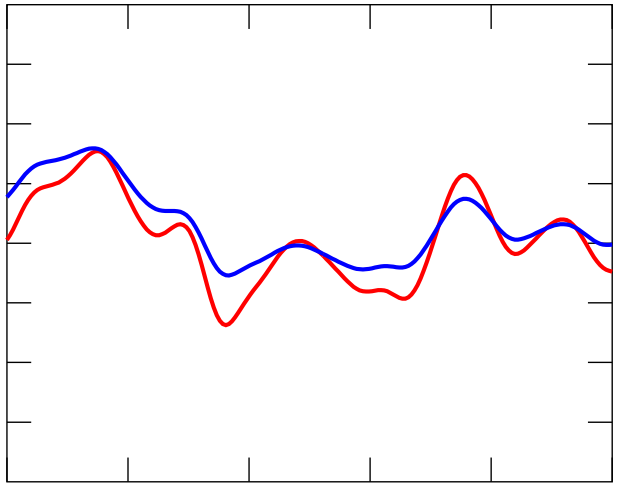


Semiparametric Latent Factor Model Samples

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

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

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



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; Williams and Barber, 1998; 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).

Outline

1 Background

2 Convolution Processes

3 Motion Capture Example

Convolution Process

- A convolution process is a moving-average construction that guarantees a valid covariance function.
- Consider a set of functions $\{f_j(\mathbf{x})\}_{j=1}^{p}$.
- Each function can be expressed as

$$f_j(\mathbf{x}) = \int_{\mathcal{X}} G_j(\mathbf{x} - \mathbf{z}) u(\mathbf{z}) \mathrm{d}\mathbf{z} = G_j(\mathbf{x}) * u(\mathbf{x}).$$

- Influence of more than one latent function, $\{u_i(\mathbf{z})\}_{i=1}^{q}$ and inclusion of an independent process $w_j(\mathbf{x})$

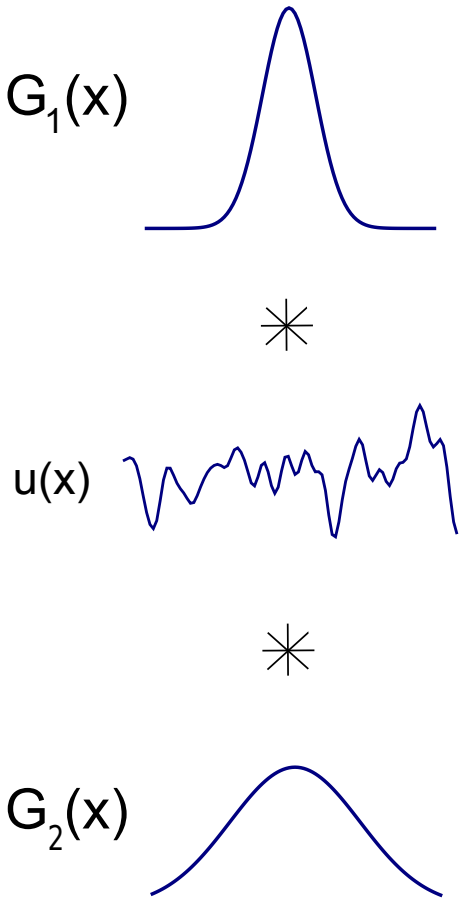
$$y_j(\mathbf{x}) = f_j(\mathbf{x}) + w_j(\mathbf{x}) = \sum_{i=1}^{q} \int_{\mathcal{X}} G_{j,i}(\mathbf{x} - \mathbf{z}) u_i(\mathbf{z}) \mathrm{d}\mathbf{z} + w_j(\mathbf{x}).$$

A pictorial representation

u(x) 

u(x): latent function.

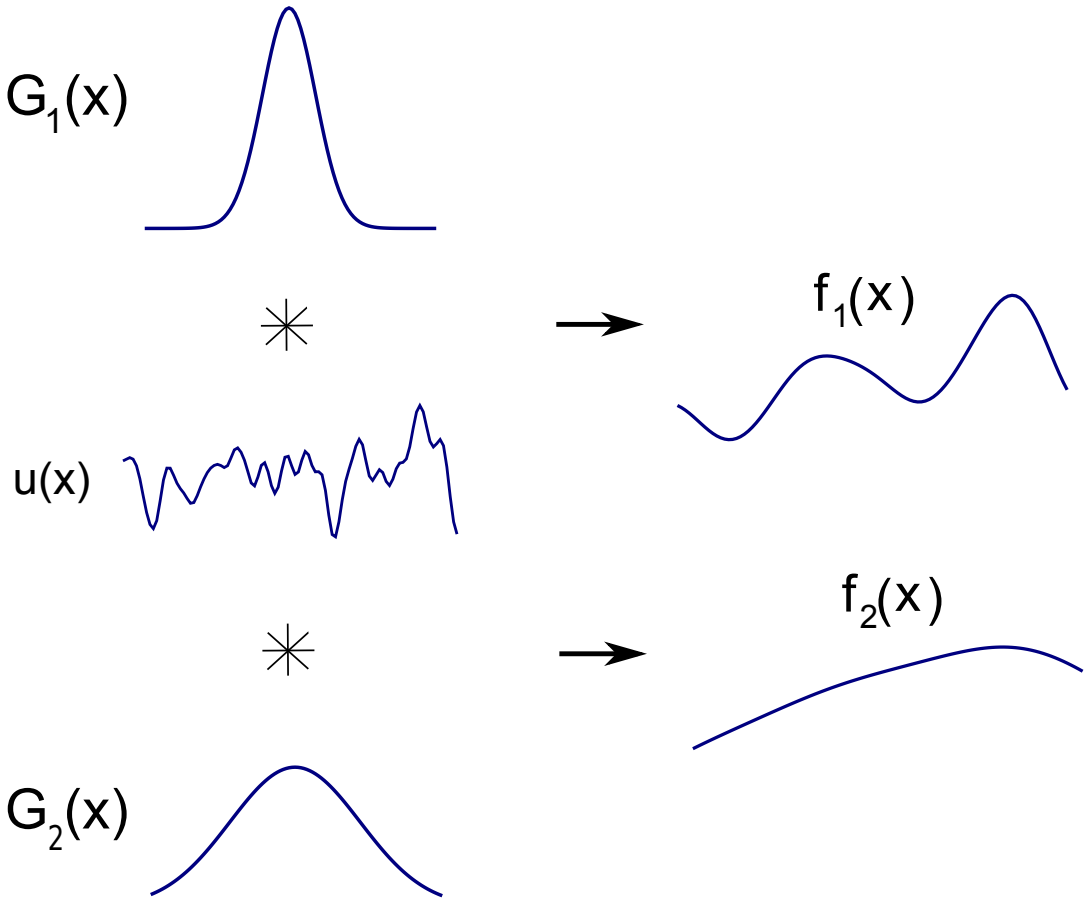
A pictorial representation



u(x): latent function.

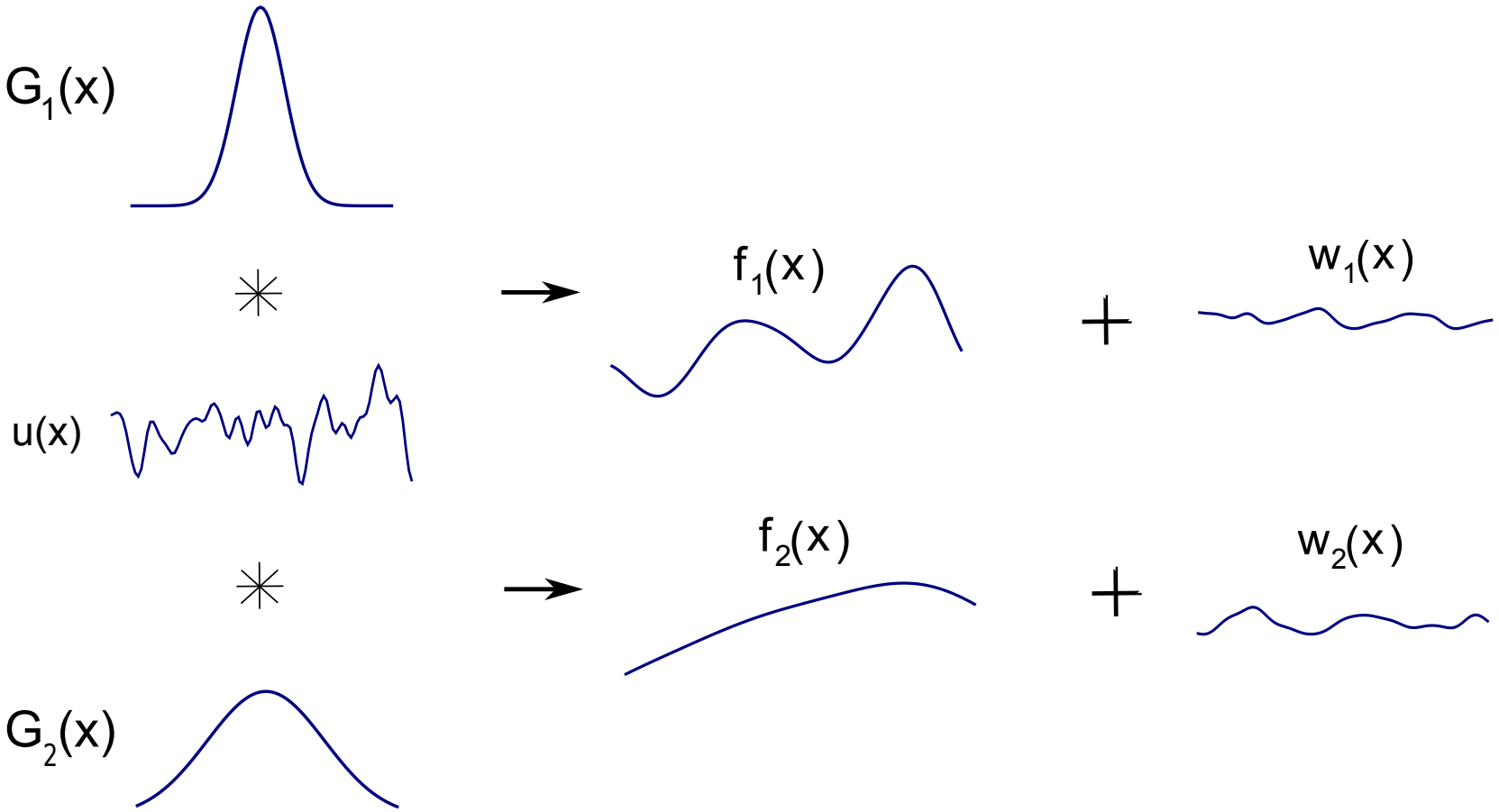
G(x): smoothing kernel.

A pictorial representation



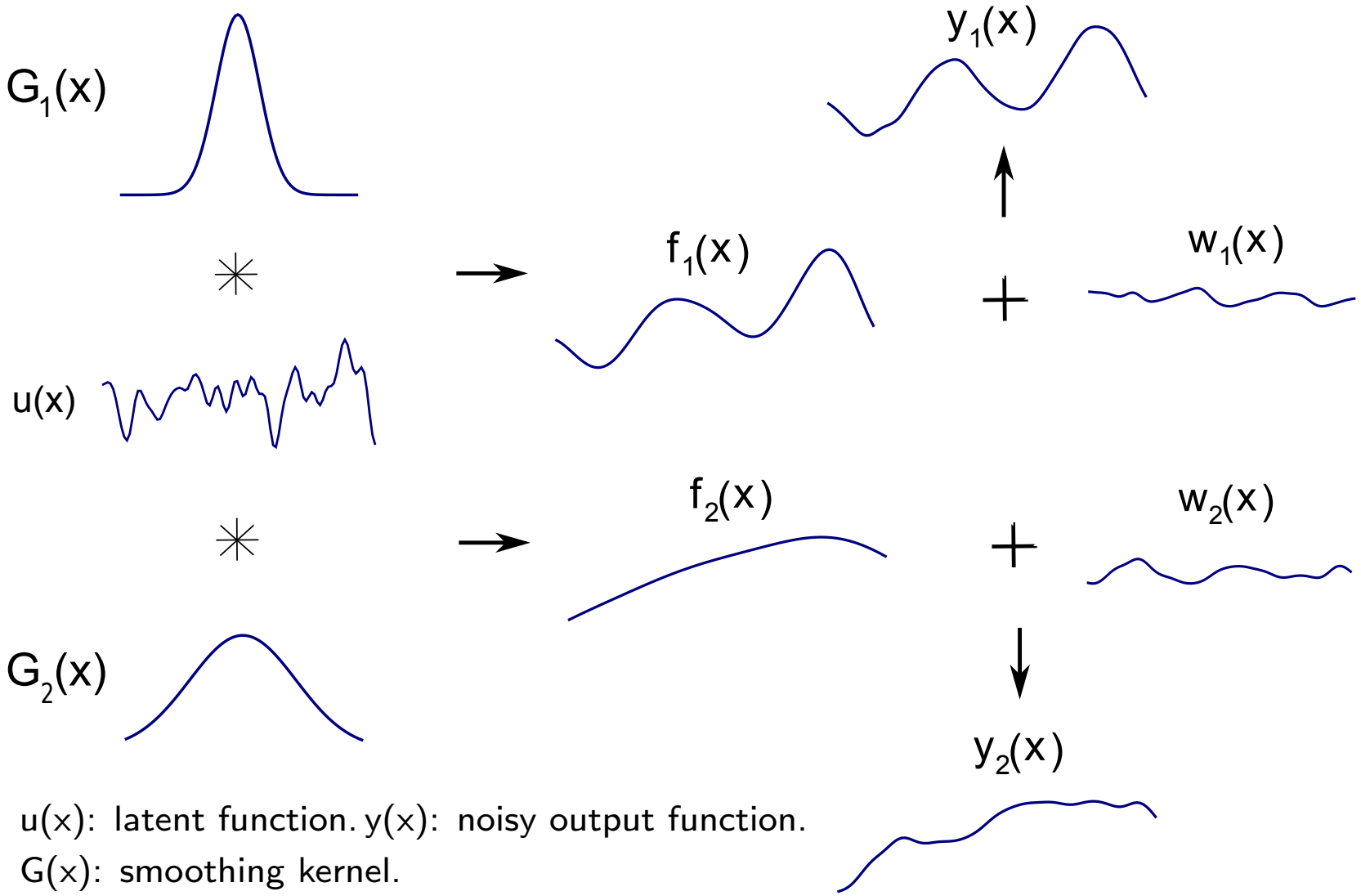
u(x): latent function.
G(x): smoothing kernel.
f(x): output function.

A pictorial representation



u(x): latent function.
G(x): smoothing kernel.
f(x): output function.
w(x): independent process.

A pictorial representation



u(x): latent function. y(x): noisy output function.
G(x): smoothing kernel.
f(x): output function.
w(x): independent process.

Covariance of the output functions.

The covariance between $y_j(\mathbf{x})$ and $y_{j'}(\mathbf{x}')$ is given as

$$\text{cov}\left[y_j(\mathbf{x}), y_{j'}(\mathbf{x}')\right] = \text{cov}\left[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')\right] + \text{cov}\left[w_j(\mathbf{x}), w_{j'}(\mathbf{x}')\right]\delta_{j,j'}$$

where

$$\text{cov}\left[f_j(\mathbf{x}), f_{j'}(\mathbf{x}')\right] = \int_{\mathcal{X}} G_j(\mathbf{x}-\mathbf{z}) \int_{\mathcal{X}} G_{j'}(\mathbf{x}'-\mathbf{z}')\text{cov}\left[u(\mathbf{z}), u(\mathbf{z}')\right] \text{d}\mathbf{z}'\text{d}\mathbf{z}$$

Different forms of covariance for the output functions.

- Input *Gaussian process*

$$\text{cov}\left[f_j, f_{j'}\right] = \int_{\mathcal{X}} G_j(\mathbf{x} - \mathbf{z}) \int_{\mathcal{X}} G_{j'}(\mathbf{x}' - \mathbf{z}') k_{u,u}(\mathbf{z}, \mathbf{z}') \mathrm{d}\mathbf{z}' \mathrm{d}\mathbf{z}$$

- Input *white noise process*

$$\text{cov}\left[f_j, f_{j'}\right] = \int_{\mathcal{X}} G_j(\mathbf{x} - \mathbf{z}) G_{j'}(\mathbf{x}' - \mathbf{z}) \mathrm{d}\mathbf{z}$$

- Covariance between output functions and latent functions

$$\text{cov}\left[f_j, u\right] = \int_{\mathcal{X}} G_j(\mathbf{x} - \mathbf{z}') k_{u,u}(\mathbf{z}', \mathbf{z}) \mathrm{d}\mathbf{z}'$$

Dimensionality Reduction

- Linear relationship between the data, $\mathbf{X} \in \Re^{n \times p}$, and a reduced dimensional representation, $\mathbf{F} \in \Re^{n \times q}$, where $q \ll p$.

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

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

- Integrate out $\mathbf{F}$, optimize with respect to $\mathbf{W}$.
- For Gaussian prior, $\mathbf{F} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - and $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ we have probabilistic PCA (Tipping and Bishop, 1999; Roweis, 1998).
  - and $\boldsymbol{\Sigma}$ constrained to be diagonal, we have factor analysis.

Dimensionality Reduction: Temporal Data

- Deal with temporal data with a temporal latent prior.
- Independent Gauss-Markov priors over each $f_i(t)$ leads to : Rauch-Tung-Striebel (RTS) smoother (Kalman filter).
- More generally consider a Gaussian process (GP) prior,

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

Joint Gaussian Process

- Given the covariance functions for $\{f_i(t)\}$ we have an implied covariance function across all $\{x_i(t)\}$—(ML: semi-parametric latent factor model (Teh et al., 2005), Geostatistics: linear model of coregionalization).
- Rauch-Tung-Striebel smoother has been preferred
  - linear computational complexity in n.
  - Advances in sparse approximations have made the general GP framework practical. (Titsias, 2009; Snelson and Ghahramani, 2006; Quiñonero Candela and Rasmussen, 2005).

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, $\mathbf{XD} = \mathbf{FS} + \boldsymbol{\epsilon}$.
  - 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: $\mathbf{W} = \mathbf{SD}^{-1}$.

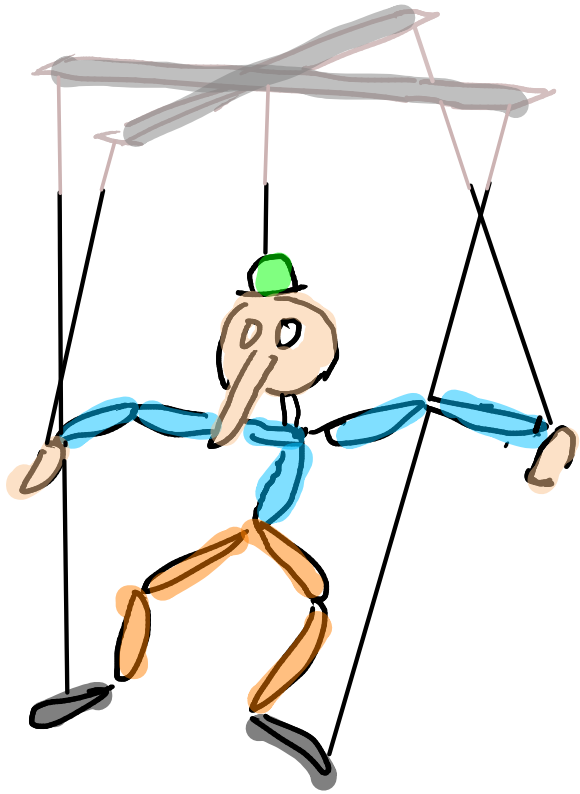
Extend Model

- 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*.

Marionette



Mass Spring Damper Analogy

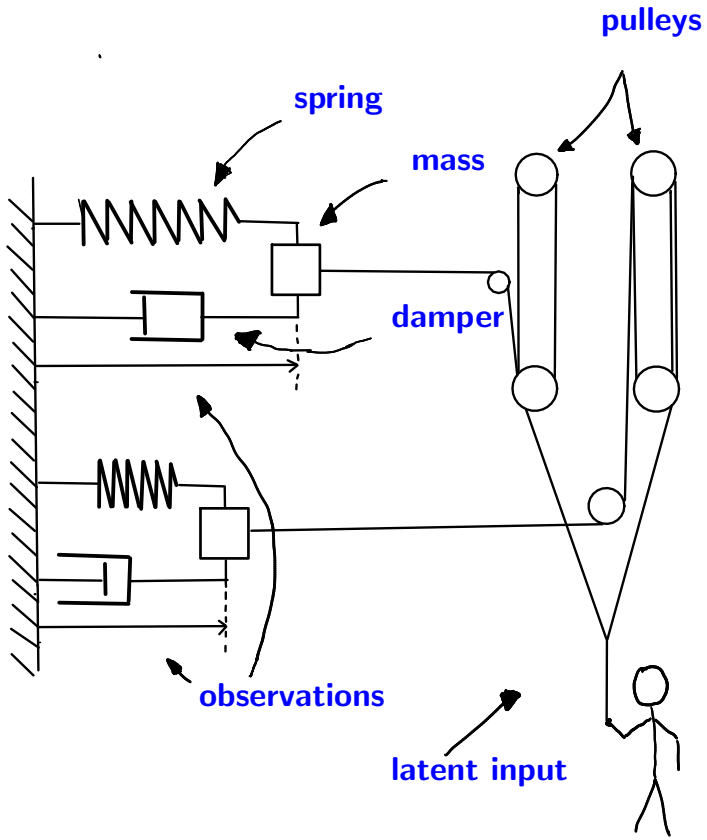


Figure: Mass spring damper analogy, an unobserved force drives multiple oscillators.

Mass Spring Damper Analogy

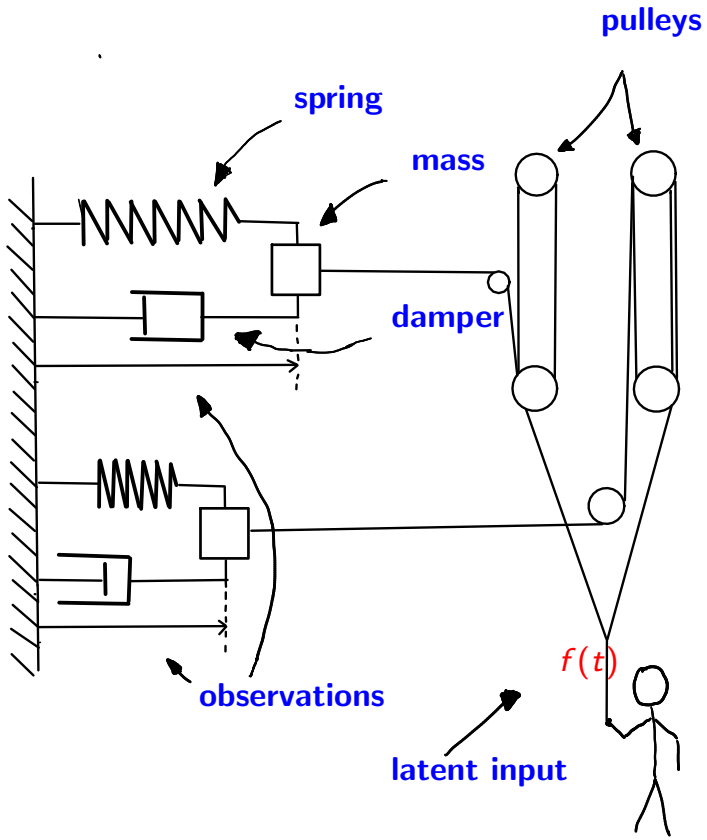


Figure: Mass spring damper analogy, an unobserved force drives multiple oscillators.

Mass Spring Damper Analogy

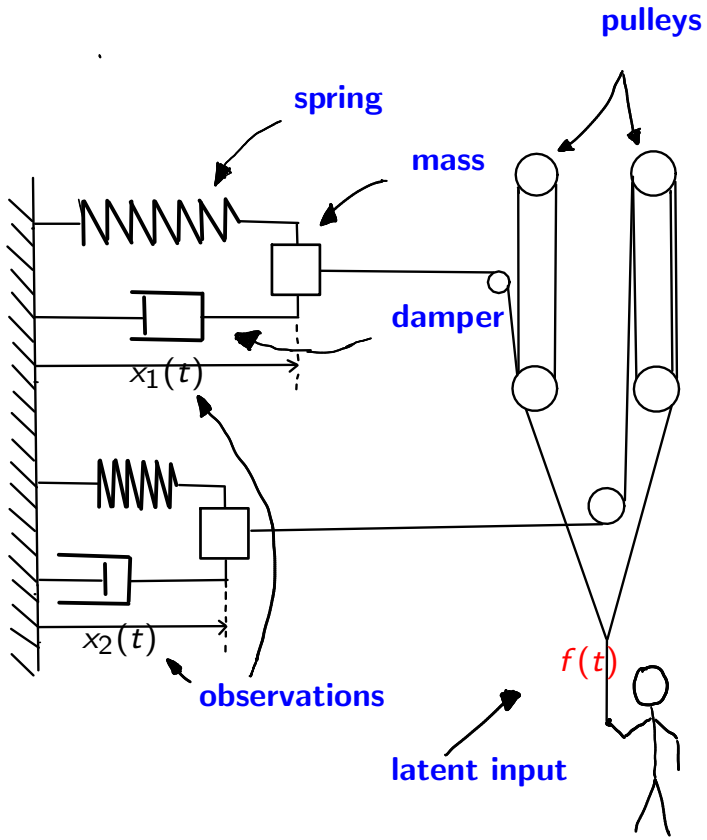


Figure: Mass spring damper analogy, an unobserved force drives multiple oscillators.

Mass Spring Damper Analogy

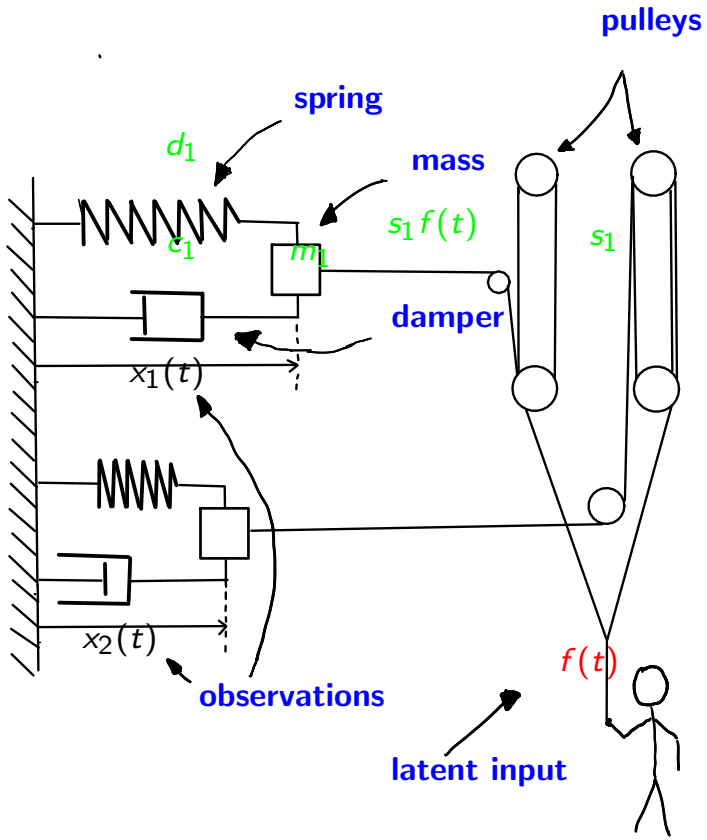


Figure: Mass spring damper analogy, an unobserved force drives multiple oscillators.

Mass Spring Damper Analogy

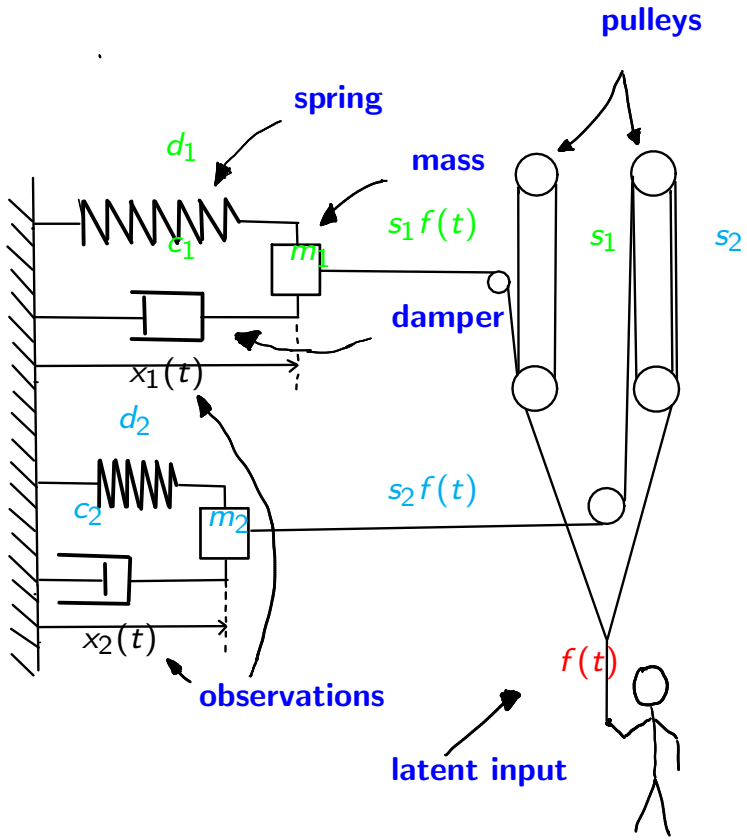


Figure: Mass spring damper analogy, an unobserved force drives multiple oscillators.

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

$$m_k \ddot{x}_k(t) + c_k \dot{x}_k(t) + d_k x_k(t) = b_k + \sum_{i=0}^{q} s_{ik} f_i(t), \tag{1}$$

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

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

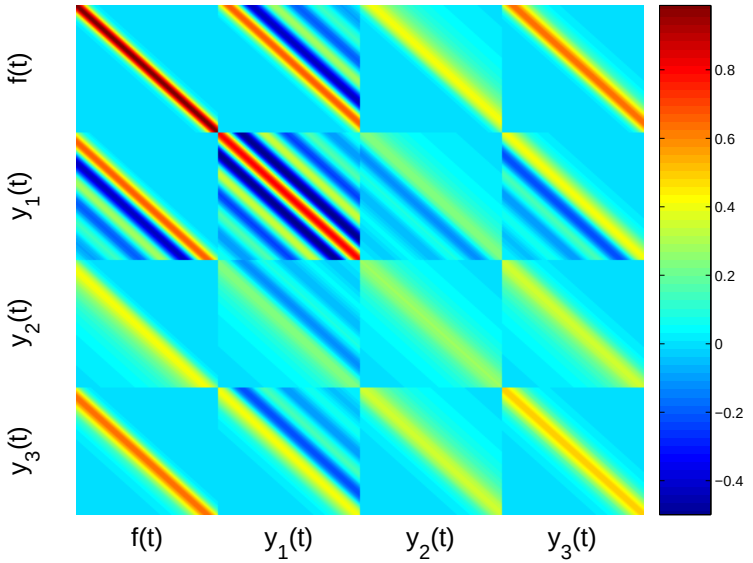
Covariance for ODE Model

- 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



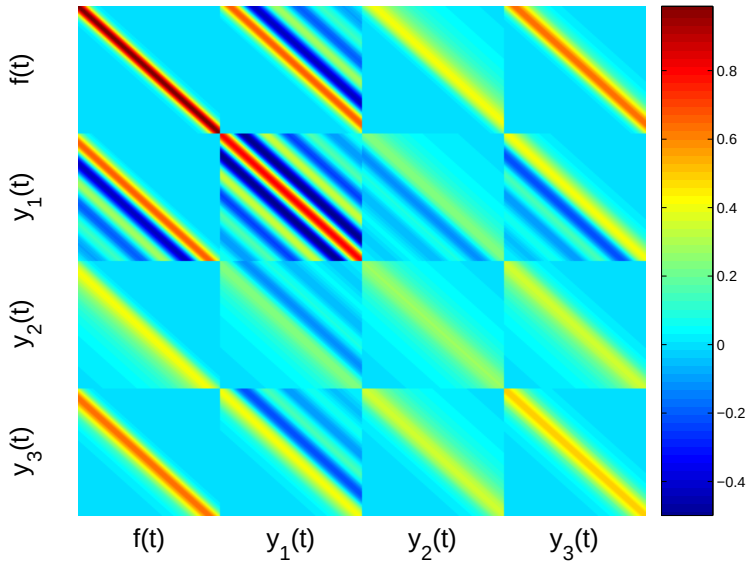
Covariance for ODE Model

- 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



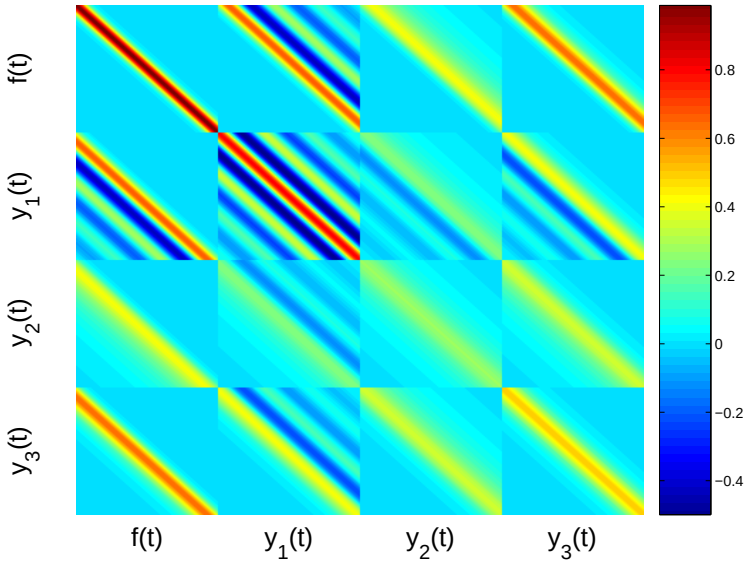
Covariance for ODE Model

- 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`

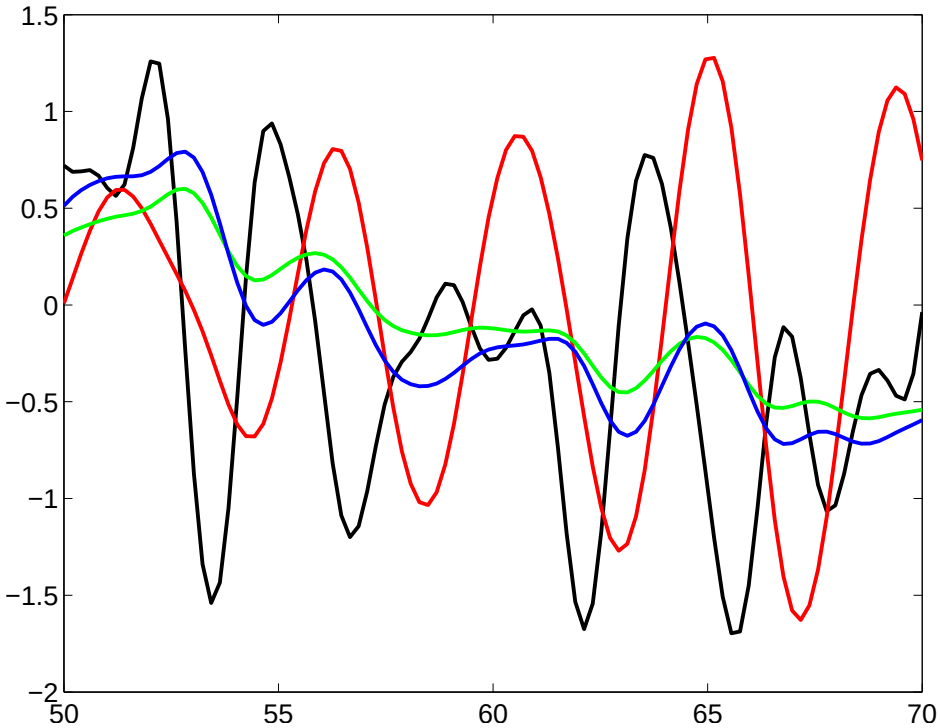


Figure: Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

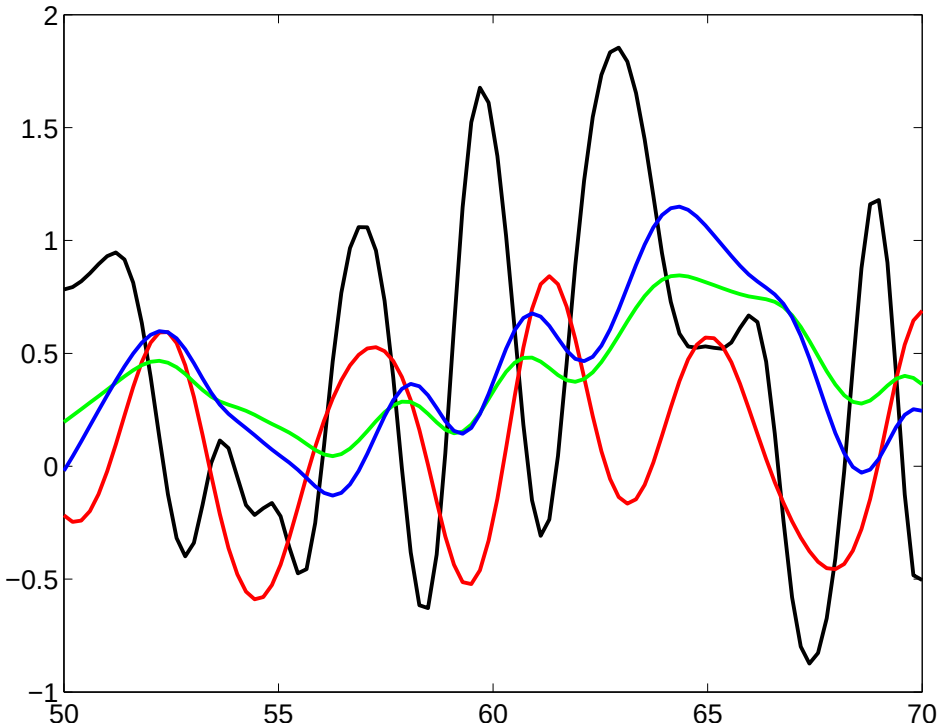


Figure: Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

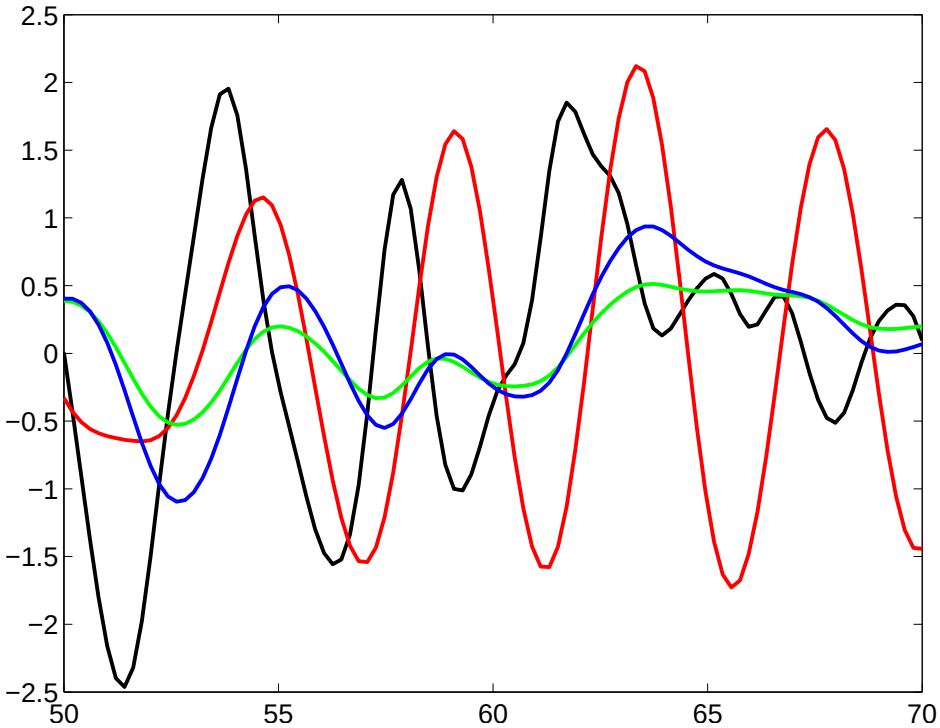


Figure: Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

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

- `lfmSample`

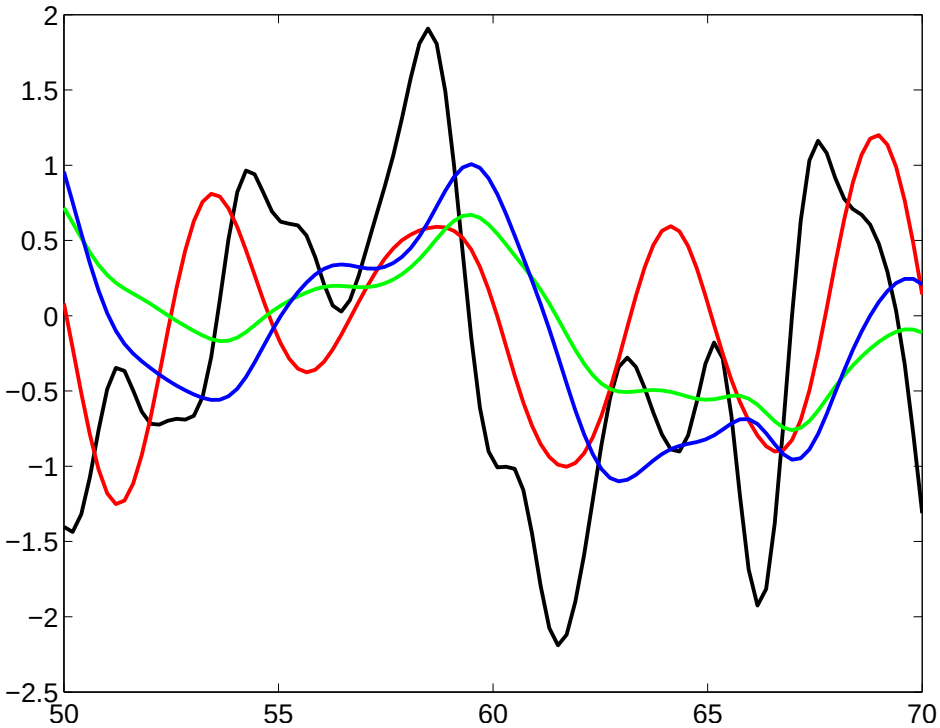


Figure: Joint samples from the ODE covariance, *black*: $f(t)$, *red*: $x_1(t)$ (underdamped), *green*: $x_2(t)$ (overdamped), and *blue*: $x_3(t)$ (critically damped).

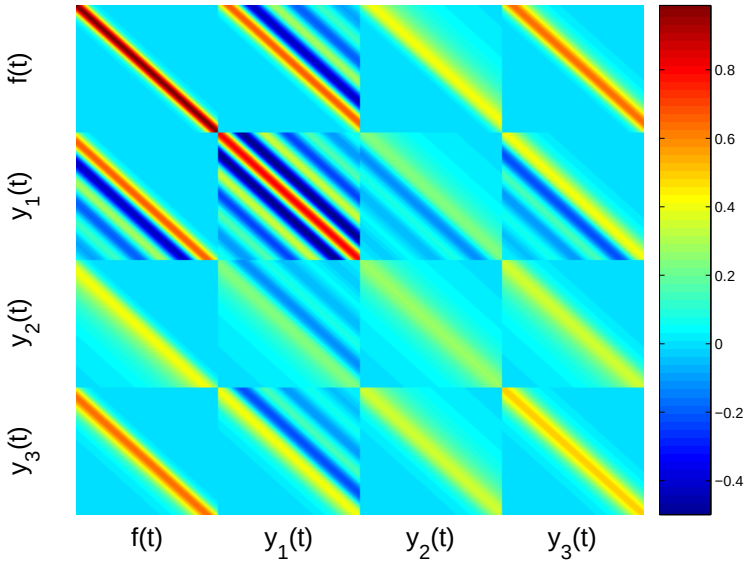
Covariance for ODE

- 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



Outline

1 Background

2 Convolution Processes

3 Motion Capture Example

Example: Motion Capture

Mauricio Alvarez and David Luengo (Álvarez et al., 2009, 2011)

- 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, 2011)

- 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, 2011)

- 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, 2011)

- 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

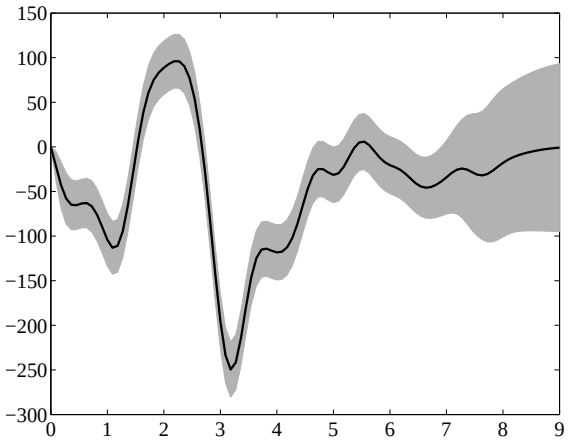
- Model left arm only.
- 3 balancing motions (18, 19, 20) from subject 49.
- 18 and 19 are similar, 20 contains more dramatic movements.
- Train on 18 and 19 and testing on 20
- Data was down-sampled by 32 (from 120 fps).
- Reconstruct motion of left arm for 20 given other movements.
- Compare with GP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

Mocap Results

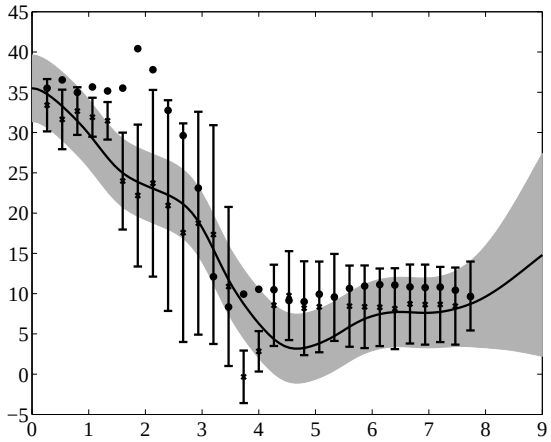
Table: Root mean squared (RMS) angle error for prediction of the left arm's configuration in the motion capture data. Prediction with the latent force model outperforms the prediction with regression for all apart from the radius's angle.

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

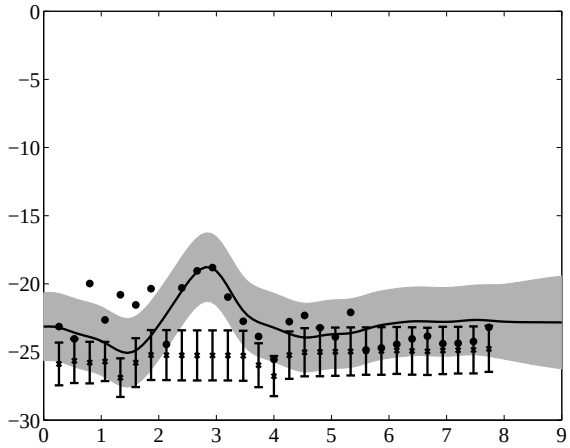
Mocap Results II



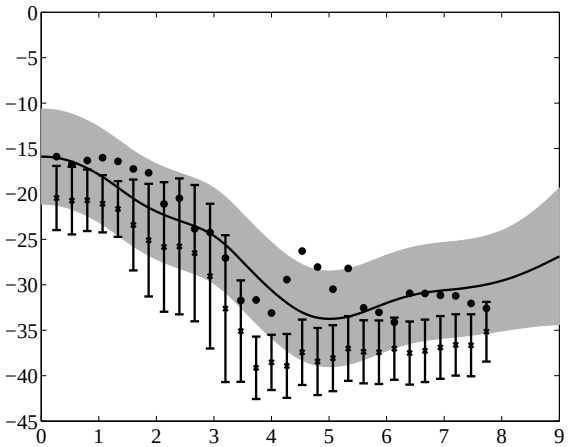
(a) Inferred Latent Force



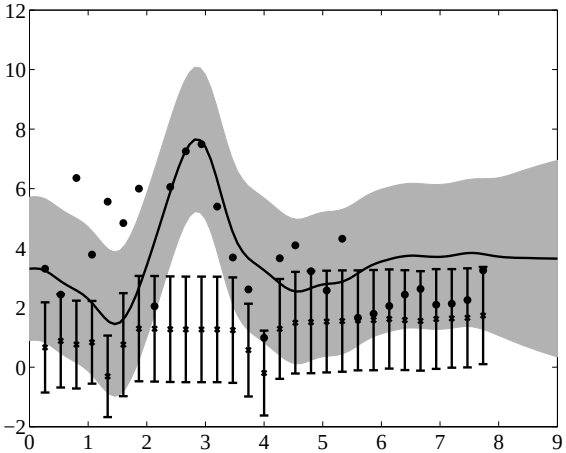
(b) Wrist



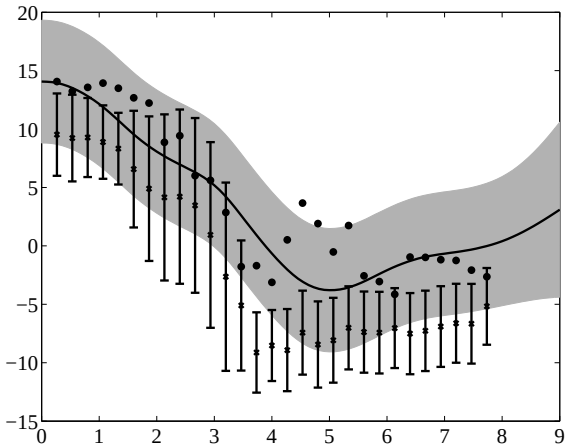
(c) Hand X Rotation



(d) Hand Z Rotation



(e) Thumb X Rotation



(f) Thumb Z Rotation

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

References I

M. A. Álvarez, D. Luengo, and N. D. Lawrence. Latent force models. In van Dyk and Welling (2009), pages 9–16. [PDF].

M. A. Álvarez, D. Luengo, and N. D. Lawrence. Linear latent force models using Gaussian processes. Technical report, University of Sheffield, [PDF].

L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. Multi-output learning via spectral filtering. *Machine Learning*, 87(3): 259–301, 2012. [DOI].

E. V. Bonilla, K. M. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, Cambridge, MA, 2008. MIT Press.

S. Conti and A. O'Hagan. Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640–651, 2009. [DOI].

G. Della Gatta, M. Bansal, A. Ambesi-Impiombato, D. Antonini, C. Missero, and D. di Bernardo. Direct targets of the trp63 transcription factor revealed by a combination of gene expression profiling and reverse engineering. *Genome Research*, 18(6): 939–948, Jun 2008. [URL]. [DOI].

P. Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, 1997. [Google Books] .

J. D. Helterbrand and N. A. C. Cressie. Universal cokriging under intrinsic coregionalization. *Mathematical Geology*, 26(2): 205–226, 1994.

D. M. Higdon, J. Gattiker, B. Williams, and M. Rightley. Computer model calibration using high dimensional output. *Journal of the American Statistical Association*, 103(482):570–583, 2008.

A. G. Journel and C. J. Huijbregts. *Mining Geostatistics*. Academic Press, London, 1978. [Google Books] .

A. A. Kalaitzis and N. D. Lawrence. A simple approach to ranking differentially expressed gene expression time courses through Gaussian process regression. *BMC Bioinformatics*, 12(180), 2011. [DOI].

N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In R. Greiner and D. Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21, pages 512–519. Omnipress, 2004. [PDF].

T. P. Minka and R. W. Picard. Learning how to learn is learning with point sets. Available on-line., 1997. [URL]. Revised 1999, available at `http://www.stat.cmu.edu/~{}minka/`.

References II

J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. [Google Books] .

S. T. Roweis. EM algorithms for PCA and SPCA. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, pages 626–632, Cambridge, MA, 1998. MIT Press.

M. Seeger and M. I. Jordan. Sparse Gaussian Process Classification With Multiple Classes. Technical Report 661, Department of Statistics, University of California at Berkeley,

G. Skolidis and G. Sanguinetti. Bayesian multitask classification with Gaussian process priors. *IEEE Transactions on Neural Networks*, 22(12):2011 – 2021, 2011.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, Cambridge, MA, 2006. MIT Press.

O. Stegle, C. Lippert, J. Mooij, N. Lawrence, and K. Borgwardt. Efficient inference in matrix-variate Gaussian models with i.i.d. observation noise. In *Neural Information Processing Systems*, 2011.

Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 333–340, Barbados, 6-8 January 2005. Society for Artificial Intelligence and Statistics.

M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3): 611–622, 1999. [PDF]. [DOI].

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In van Dyk and Welling (2009), pages 567–574.

D. van Dyk and M. Welling, editors. *Artificial Intelligence and Statistics*, volume 5, Clearwater Beach, FL, 16-18 April 2009. JMLR W&CP 5.

H. Wackernagel. *Multivariate Geostatistics: An Introduction With Applications*. Springer-Verlag, 3rd edition, 2003. [Google Books] .

C. K. Williams and D. Barber. Bayesian Classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

I. Woodward, M. R. Lomas, and R. A. Betts. Vegetation-climate feedbacks in a greenhouse world. *Philosophical Transactions: Biological Sciences*, 353(1365):29–39, 1998.

K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 1012–1019, 2005.