# Latent Force Models with Gaussian Processes

**Neil D. Lawrence**
work with Mauricio Alvarez, David Luengo

Inspire Workshop, Imperial College

24th September 2009

## Outline

# Outline

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

$$\mathbf{X} = \mathbf{FW} + \boldsymbol{\epsilon},$$

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

- Integrate out $\mathbf{F}$, optimize with respect to $\mathbf{W}$.
- For temporal data and a particular Gaussian prior in the latent space: Kalman filter/smoother.
- More generally consider a Gaussian process (GP) prior,

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

# Dimensionality Reduction II

- Given the covariance functions for $\{f_i(t)\}$ the implied covariance functions for $\{x_i(t)\}$ — semi-parametric latent factor model (Teh et al., 2005).
- Kalman filter/smoother approach 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).

# Outline

**Zero mean Gaussian distribution**

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

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

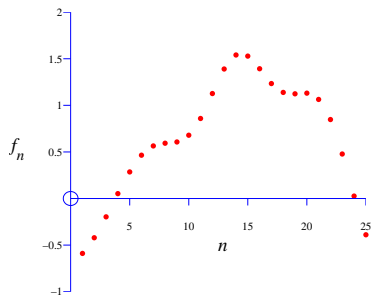▶ We will consider the special case where the mean is zero,

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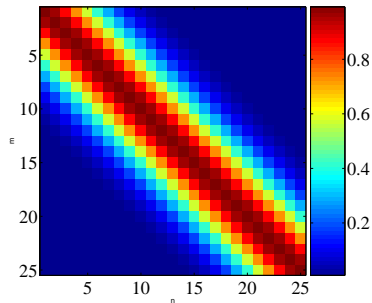
**Multi-variate Gaussians**

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

# Gaussian Distribution Sample

`demGPSample`



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

**The covariance matrix**

▶ Covariance matrix shows correlation between points $f_m$ and $f_n$ if $n$ is near to $m$.

▶ Less correlation if $n$ is distant from $m$.

▶ Our ordering of points means that the *function appears smooth*.

▶ Let's focus on the joint distribution of two points form the 25.

**The covariance matrix**

▶ Covariance matrix shows correlation between points $f_m$ and $f_n$ if $n$ is near to $m$.

▶ Less correlation if $n$ is distant from $m$.

▶ Our ordering of points means that the *function appears smooth*.

▶ Let's focus on the joint distribution of two points form the 25.

**The covariance matrix**

▶ Covariance matrix shows correlation between points $f_m$ and $f_n$ if $n$ is near to $m$.

▶ Less correlation if $n$ is distant from $m$.

▶ Our ordering of points means that the *function appears smooth*.

▶ Let's focus on the joint distribution of two points form the 25.

**The covariance matrix**

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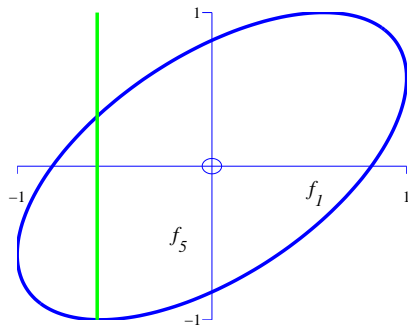- Let's focus on the joint distribution of two points form the 25.

```
demGpCov2D([1 2])
```



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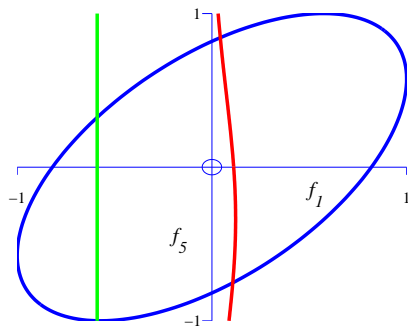
# Prediction of $f_2$ from $f_1$

```
demGpCov2D([1 2])
```



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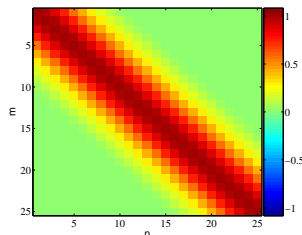
# Prediction of $f_2$ from $f_1$

```
demGpCov2D([1 2])
```



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

# Prediction of $f_5$ from $f_1$

`demGpCov2D([1 5])`



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

# Prediction of $f_5$ from $f_1$

```
demGpCov2D([1 5])
```



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

# Prediction of $f_5$ from $f_1$

```
demGpCov2D([1 5])
```



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

**RBF Kernel Function**

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

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

# Covariance Samples

`demCovFuncSample`



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

# Covariance Samples

`demCovFuncSample`



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

# Covariance Samples

`demCovFuncSample`



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

# Gaussian Process Regression

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.
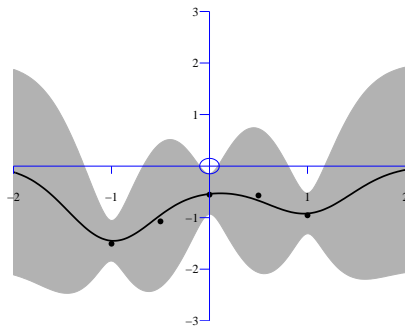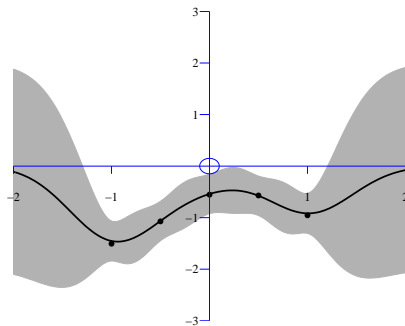
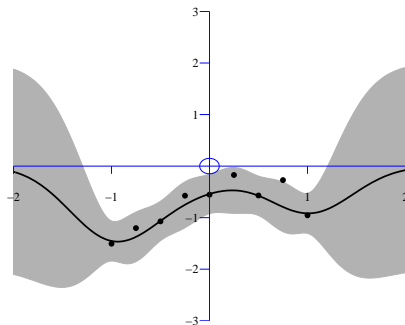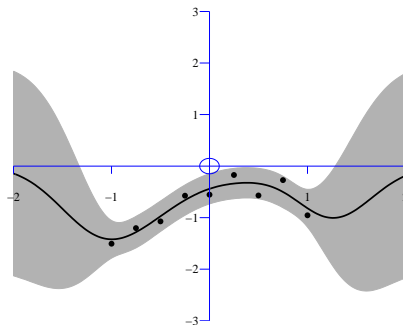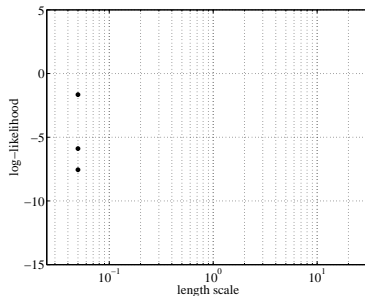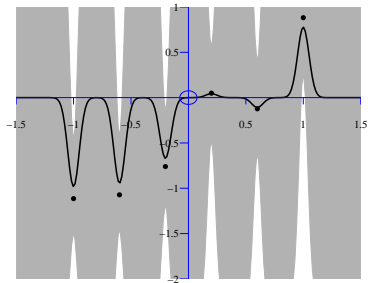# Gaussian Process Regression

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.

# Gaussian Process Regression

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.

# Gaussian Process Regression

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.

# Gaussian Process Regression

`demRegression`



Figure: Examples include WiFi localization, C14 callibration curve.

# Gaussian Process Regression

demRegression



Figure: Examples include WiFi localization, C14 callibration curve.

# Gaussian Process Regression

`demRegression`



Figure: Examples include WiFi localization, C14 callibration curve.

# Learning Kernel Parameters
Can we determine length scales and noise levels from the data?



demOptimiseGp

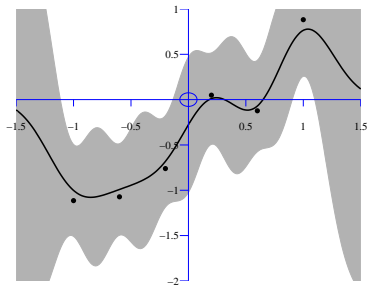$$\log N \left( \mathbf{f} | \mathbf{0}, \mathbf{K} \right) = -\frac{N}{2} \log 2\pi - \frac{1}{2} \log |\mathbf{K}| - \frac{\mathbf{f}^{\mathrm{T}} \mathbf{K}^{-1} \mathbf{f}}{2}$$

# Learning Kernel Parameters

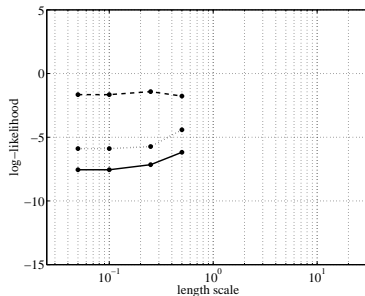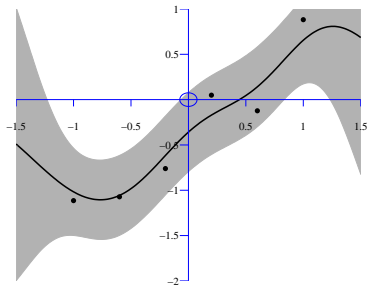Can we determine length scales and noise levels from the data?

`demOptimiseGp`



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

# Learning Kernel Parameters
Can we determine length scales and noise levels from the data?
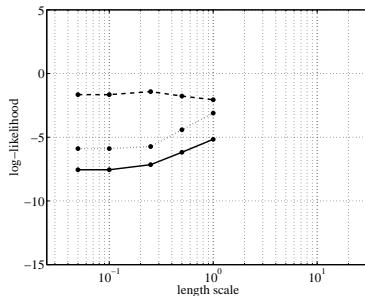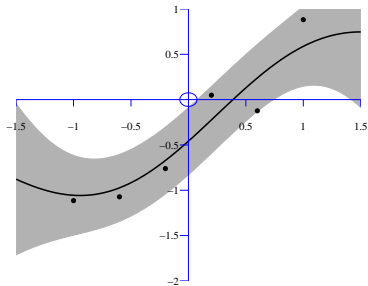
`demOptimiseGp`



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

# Learning Kernel Parameters

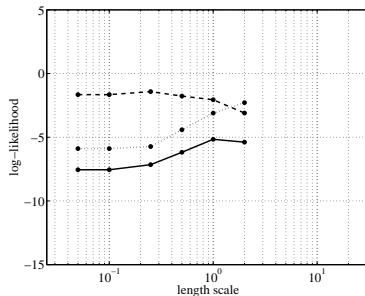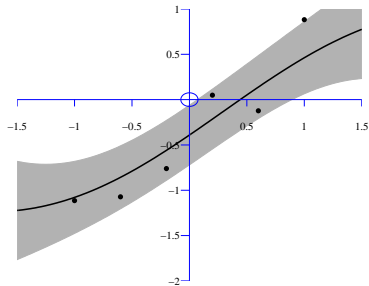Can we determine length scales and noise levels from the data?



demOptimiseGp

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

# Learning Kernel Parameters
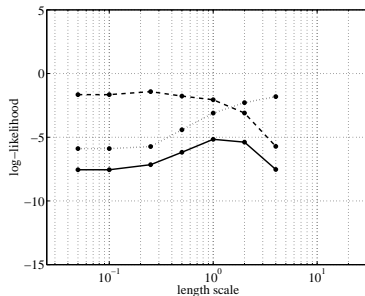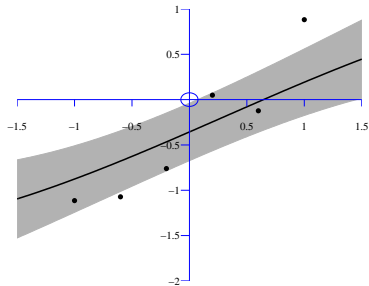Can we determine length scales and noise levels from the data?

demOptimiseGp



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

# Learning Kernel Parameters
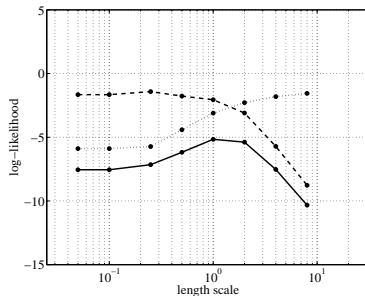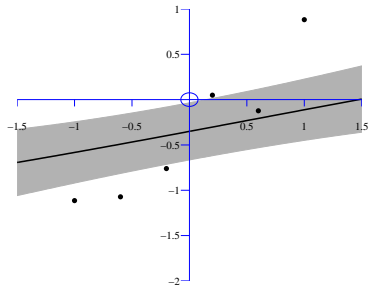Can we determine length scales and noise levels from the data?



demOptimiseGp

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

# Learning Kernel Parameters

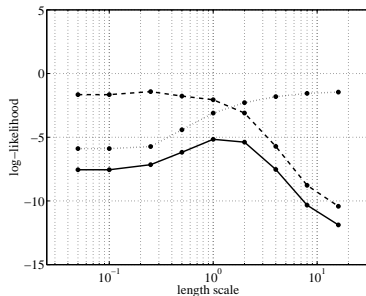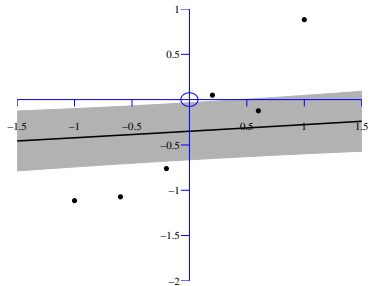Can we determine length scales and noise levels from the data?

demOptimiseGp



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

# Learning Kernel Parameters
Can we determine length scales and noise levels from the data?

demOptimiseGp



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

# Learning Kernel Parameters

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

`demOptimiseGp`



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

# Outline

**Back to Latent Force 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 $d$ 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, $\mathbf{S} \in \Re^{q \times d}$.
  - ▶ Diagonal matrix of spring constants, $\mathbf{D} \in \Re^{d \times d}$.
  - ▶ Original System: $\mathbf{W} = \mathbf{SD}^{-1}$.

- Add a damper and give the system mass.

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

- 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*.

# Outline

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

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

where, $m_k$ is the $k$th diagonal element from **M** and similarly for $c_k$ and $d_k$. $s_{ik}$ is the $i$, $k$th element of **S**.

- Model the latent forces as $q$ independent, GPs with RBF covariances

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

# Covariance for ODE Model

► RBF Kernel function for $f(t)$

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

► Joint distribution for $x_1(t)$, $x_2(t)$, $x_3(t)$ and $f(t)$. Damping ratios:
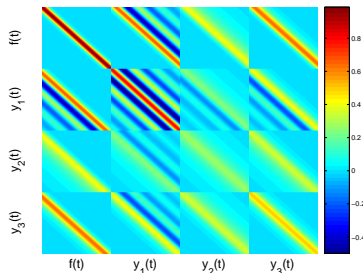
| $\zeta_1$ | $\zeta_2$ | $\zeta_3$ |
|-----------|-----------|-----------|
| 0.125 | 2 | 1 |

▶ `demLfmSample`



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

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

▶ `demLfmSample`



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

▶ `demLfmSample`



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

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

▶ `demLfmSample`



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

# Covariance for ODE

- RBF Kernel function for $f(t)$

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

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

| $\zeta_1$ | $\zeta_2$ | $\zeta_3$ |
|---|---|---|
| 0.125 | 2 | 1 |

# Outline

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

▶ 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.

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

▶ 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.

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

▶ 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)**

- ▶ 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.

- 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).

# Outline

## Partial Differential Equations and Latent Forces

- ▶ 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

▶ 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

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

with $T_q = \kappa_q t$.

## Covariance Function

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

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

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

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

# Prediction of Metal Concentrations

- ▶ 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).[1]
- ▶ 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).

---

[1]Data available at http://www.ai-geostats.org/.

## 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 | $0.5823 \pm 0.0133$ | $0.4505 \pm 0.0126$ | 0.5 |
| Cu | $15.9357 \pm 0.0907$ | $7.1677 \pm 0.2266$ | 7.8 |
| Pb | $22.9141 \pm 0.6076$ | $10.1097 \pm 0.2842$ | 10.7 |
| Co | $2.0735 \pm 0.1070$ | $1.7546 \pm 0.0895$ | 1.5 |

# Outline

- Integration of probabilistic inference with mechanistic models.
- Ongoing/other work:
    - Biological systems.
    - Non linear response and non linear differential equations.
    - Scaling up to larger systems
    - Applications to other types of system, *e.g.* computational biology, spatial systems.
    - Stochastic differential equations

## Acknowledgements

# References I

M. Álvarez and N. D. Lawrence. Sparse convolved Gaussian processes for multi-output regression. In Koller et al. (2009). [PDF]. To appear.

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

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

P. Boyle and M. Frean. Dependent Gaussian processes. In L. Saul, Y. Weiss, and L. Bouttou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 217–224, Cambridge, MA, 2005. MIT Press.

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

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

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

D. M. Higdon. Space and space-time modelling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

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

D. Koller, Y. Bengio, D. Schuurmans, and L. Bottou, editors. *Advances in Neural Information Processing Systems*, volume 21, Cambridge, MA, 2009. MIT Press. To appear.

P. S. Laplace. Mémoire sur la probabilité des causes par les évènemens. In *Mémoires de mathèmatique et de physique, presentés à lAcadémie Royale des Sciences, par divers savans, & lù dans ses assemblées 6*, pages 621–656, 1774. Translated in Stigler (1986).

M. A. Osborne, A. Rogers, S. D. Ramchurn, S. J. Roberts, and N. R. Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN 2008)*, 2008.

# References II

A. D. Polyanin. *Handbook of Linear Partial Differential Equations for Engineers and Scientists*. Chapman & Hall/CRC, 1 edition, 2002.

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.

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

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.

S. M. Stigler. Laplace's 1774 memoir on inverse probability. *Statistical Science*, 1:359–378, 1986.

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. Titsias, N. D. Lawrence, and M. Rattray. Efficient sampling for Gaussian process inference using control variables. In Koller et al. (2009). [PDF]. To appear.

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.

# Example: Transcriptional Regulation

▶ First Order Differential Equation

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

▶ Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.

▶ $x_j(t)$ – concentration of gene $j$'s mRNA

▶ $f(t)$ – concentration of active transcription factor

▶ Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$

▶ Application: identifying co-regulated genes (targets)

▶ Problem: how do we fit the model when $f(t)$ is not observed?

## Example: Transcriptional Regulation

- ▶ First Order Differential Equation

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

- ▶ Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- ▶ $x_j(t)$ – concentration of gene $j$'s mRNA
- ▶ $f(t)$ – concentration of active transcription factor
- ▶ Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$
- ▶ Application: identifying co-regulated genes (targets)
- ▶ Problem: how do we fit the model when $f(t)$ is not observed?

## Example: Transcriptional Regulation

▶ First Order Differential Equation

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

▶ Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.

▶ $x_j(t)$ – concentration of gene $j$'s mRNA

▶ $f(t)$ – concentration of active transcription factor

▶ Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$

▶ Application: identifying co-regulated genes (targets)

▶ Problem: how do we fit the model when $f(t)$ is not observed?

# Example: Transcriptional Regulation

- First Order Differential Equation

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

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.

- $x_j(t)$ – concentration of gene $j$'s mRNA

- $f(t)$ – concentration of active transcription factor

- Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$

- Application: identifying co-regulated genes (targets)

- Problem: how do we fit the model when $f(t)$ is not observed?

# Example: Transcriptional Regulation

- First Order Differential Equation

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

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $x_j(t)$ – concentration of gene $j$'s mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

## Example: Transcriptional Regulation

- First Order Differential Equation

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

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $x_j(t)$ – concentration of gene $j$'s mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

- First Order Differential Equation

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

- Can be used as a model of gene transcription: Barenco et al., 2006; Gao et al., 2008.
- $x_j(t)$ – concentration of gene $j$'s mRNA
- $f(t)$ – concentration of active transcription factor
- Model parameters: baseline $B_j$, sensitivity $S_j$ and decay $D_j$
- Application: identifying co-regulated genes (targets)
- Problem: how do we fit the model when $f(t)$ is not observed?

**RBF covariance function for** $f(t)$

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

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

    | $D_1$ | $S_1$ | $D_2$ | $S_2$ |
    |-------|-------|-------|-------|
    | 5     | 5     | 0.5   | 0.5   |

## 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.
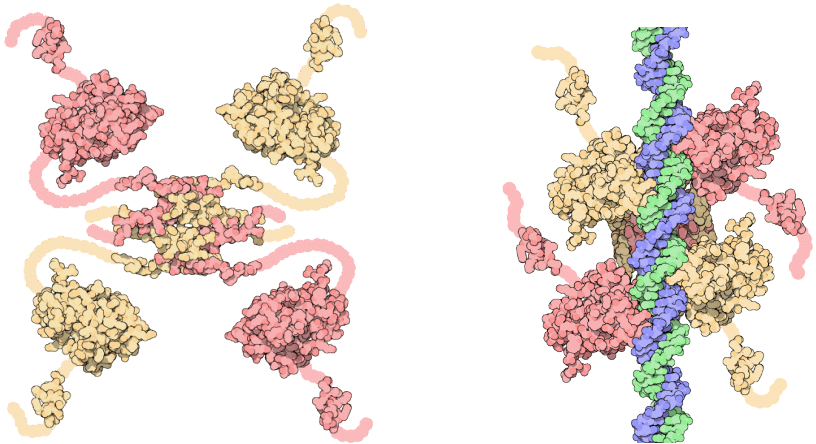- ▶ Large scale feeback loop with NF-$\kappa$B.

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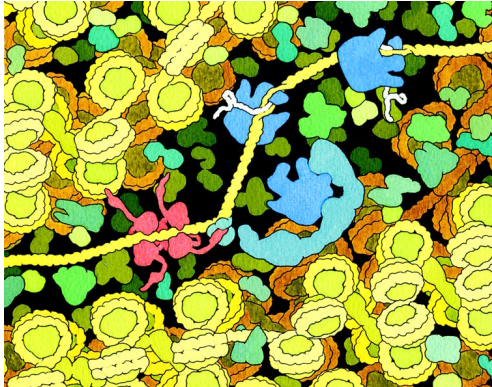
Figure: Repair of DNA damage by p53. Image from Goodsell (1999).

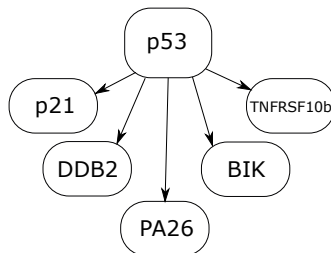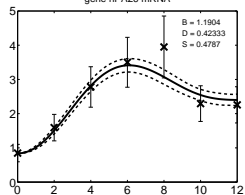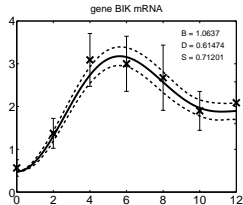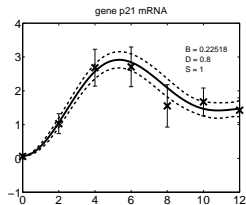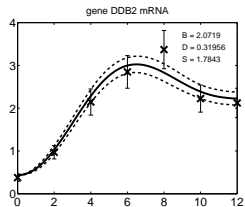- Assume p53 affects targets as a single input module network motif (SIM).
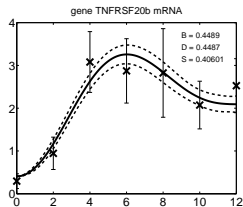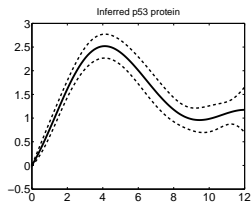


Figure: p53 SIM network motif as modelled by Barenco et al. 2006.

# p53 (RBF covariance)
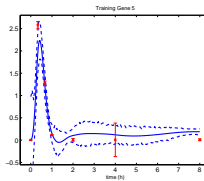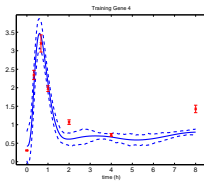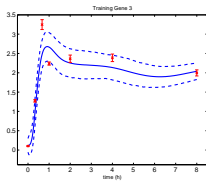
**Pei Gao**

- 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**

Fitted model used to rank potential targets of Elk-1

# 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 $O(N^3 d^3)$ complexity, $O(N^2 d^2)$ 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 $O(N^3 dk^2)$ complexity, $O(N^2 dk)$ storage.
  - Can also do a FITC style approximation (Snelson and Ghahramani, 2006).
  - Reduces to $O(Ndk^2)$ complexity, $O(Ndk)$ storage.

**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

**Mauricio Alvarez**

- Jura dataset — concentrations of several heavy metals.
- 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.

# Outline

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i f(t)}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Khanin et al., 2006, PNAS 103

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i f(t)}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Khanin et al., 2006, PNAS 103

# Models of non-linear regulation

- Non-linear Activation: Michaelis-Menten Kinetics

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i f(t)}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Rogers and Girolami (2006)

- Non-linear Repression

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = B_i + \frac{S_i}{\gamma_i + f(t)} - D_i x_i(t)$$

used by Khanin et al., 2006, PNAS 103

## MAP Laplace Approximation

Consider the following modification to the model,

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

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

$$x_j(t) = \frac{B_j}{D_j} + S_j \int_0^t e^{-D_j(t-u)} g_j\left(f(u)\right) \mathrm{d}u$$
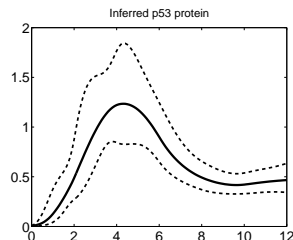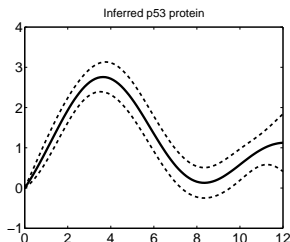
Use Laplace's method (Laplace, 1774),

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

where $\hat{\mathbf{f}} = \operatorname{argmax} p(\mathbf{f} \mid \mathbf{x})$ and $\mathbf{A} = -\nabla\nabla \log p\left(\mathbf{f} \mid \mathbf{y}\right)\big|_{\mathbf{f}=\hat{\mathbf{f}}}$ is the Hessian of the negative posterior at that point.

# p53 and Michaelis-Menten Kinetics

**Pei Gao**

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

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

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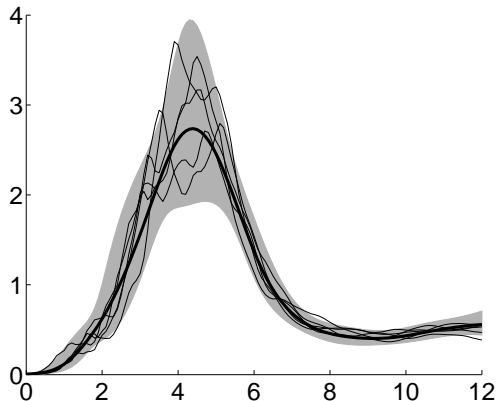


(a)

**Michalis Titsias**



Figure: Laplace approximation error bars along with samples from the true posterior distribution.

**Michalis Titsias**

▶ Sample in Gaussian processes

$$p(\mathbf{f}|\mathbf{x}) \propto p(\mathbf{x}|\mathbf{f}) \, p(\mathbf{f})$$

▶ Likelihood relates GP to data through

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

▶ We use *control points* for fast sampling. (Titsias et al., 2009)

## Sampling using control points

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

$$Q\left(\mathbf{f}^{(t+1)}|\mathbf{f}^{(t)}\right) = p\left(\mathbf{f}_\rho^{(t+1)}|\mathbf{f}_c^{(t+1)}\right) q\left(\mathbf{f}_c^{(t+1)}|\mathbf{f}_c^{(t)}\right)$$
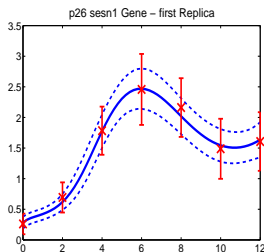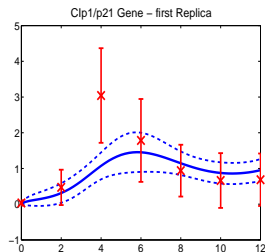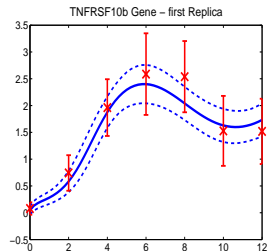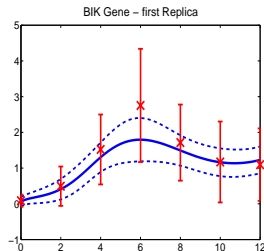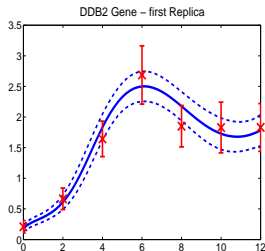
- Its like sampling from the prior $p(\mathbf{f})$ but imposing random walk behaviour through the control points.

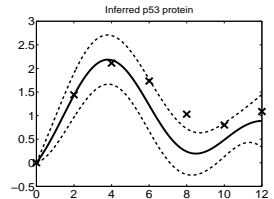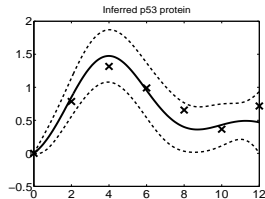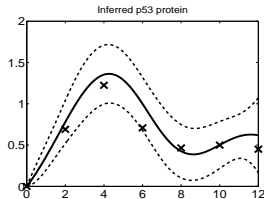- One transcription factor (p53) that acts as an activator. We consider the Michaelis-Menten kinetic equation

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

- 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
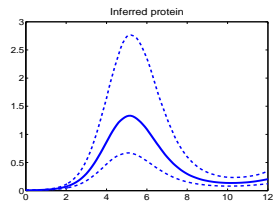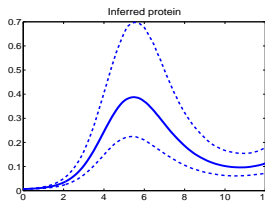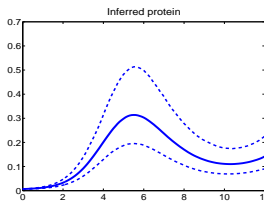  - Acceptance rate for **f** after burn in was between $15\% - 25\%$
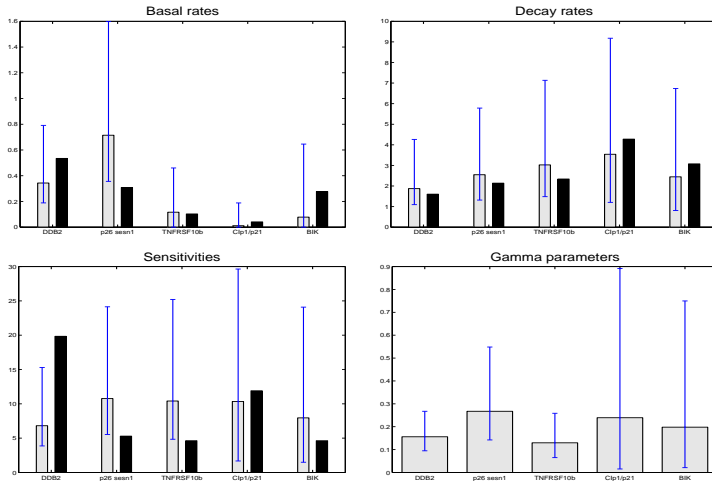
Linear model (Barenco et al. predictions are shown as crosses)



Nonlinear (Michaelis-Menten kinetic equation)

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

# Cascaded Differential Equations

**Antti Honkela**

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

# Drosophila *Mesoderm* Development

**Data from Furlong Lab in EMBL Heidelberg.**

- Describe mesoderm development.

# Cascaded Differential Equations

**Antti Honkela**

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

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = \sigma y(t) - \delta f(t)$$

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

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

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

**RBF covariance function for** $y(t)$
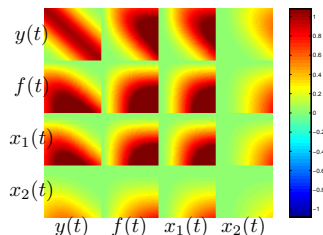
$$
\begin{aligned}
f(t) &= \sigma \exp(-\delta t) \int_0^t y(u) \exp(\delta u)\, \mathrm{d}u \\
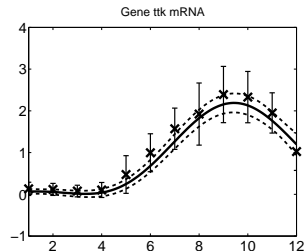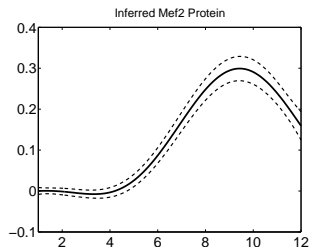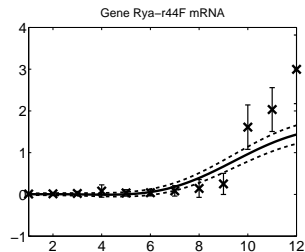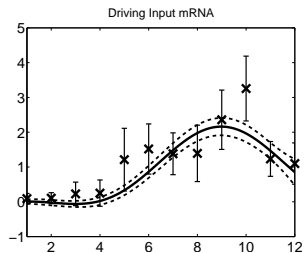x_i(t) &= \frac{B_i}{D_i} + S_i \exp(-D_i t) \int_0^t f(u) \exp(D_i u)\, \mathrm{d}u.
\end{aligned}
$$

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

| $\delta$ | $D_1$ | $S_1$ | $D_2$ | $S_2$ |
|----------|-------|-------|-------|-------|
| 0.1      | 5     | 5     | 0.5   | 0.5   |

# Results for Mef2 using the Cascade model



[c]