Bayesian latent variable modelling with Gaussian processes

Andreas Damianou

Department of Computer Science, University of Sheffield, UK

Gaussian Processes Summer School, Sheffield, 14/09/2015

Outline

Outline

Part 1: Bayesian modelling

- Overfitting, model complexity and Occam's razor

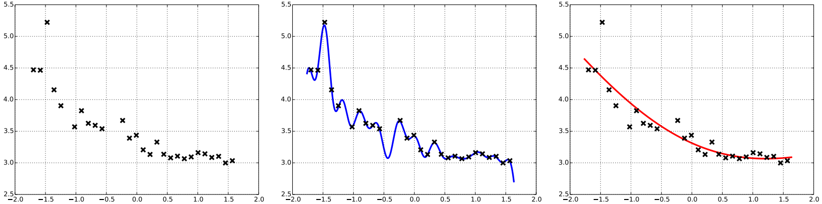
Part 2: Bayesian latent variable modelling with GPs

- Formulation
- Tractability Issues
- Advantages and Extensions

Part 3: Bayesian extensions

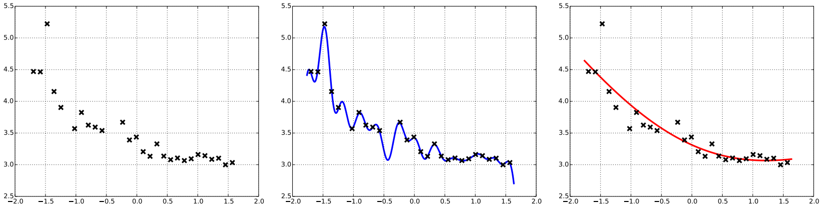
- Deep Gaussian processes
- Multi-view modelling

Curve fitting



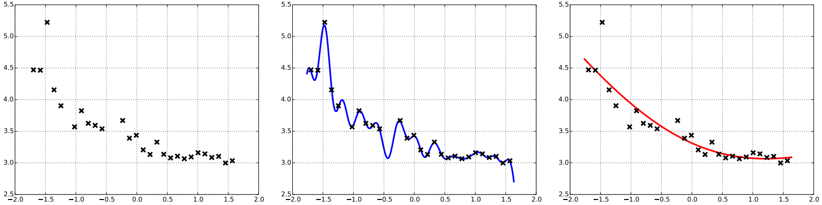
- Which curve fits the data better?

Curve fitting



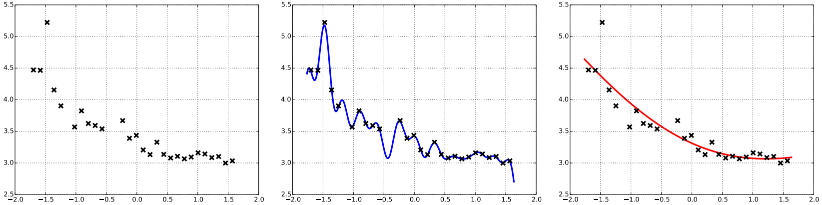
- Which curve fits the data better?
- Which curve is more “complex”?

Curve fitting



- Which curve fits the data better?
- Which curve is more “complex”?
- Which curve is better overall?

Curve fitting



- Which curve fits the data better?
- Which curve is more "complex"?
- Which curve is better overall?

Need a good balance between data fit vs overfitting!

How do GPs solve the overfitting problem?

How do GPs solve the overfitting problem?

- Answer: Integrate over the function itself!
- This is associated with the Bayesian methodology.
- So, we will average out all possible function forms, under a (GP) prior!

Recap:

$$\text{ML:} \quad \underset{\mathbf{w}}{\operatorname{argmax}}\ p(\mathbf{y}|\mathbf{w}, \phi(\mathbf{x})) \qquad \text{e.g. } \mathbf{y} = \phi(\mathbf{x})^\top \mathbf{w} + \boldsymbol{\epsilon}$$

$$\text{Bayesian:} \quad \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) \underbrace{p(\mathbf{f}|\mathbf{x}, \boldsymbol{\theta})}_{\text{GP prior}} \qquad \text{e.g. } \mathbf{y} = f(\mathbf{x}, \boldsymbol{\theta}) + \boldsymbol{\epsilon}$$

- $\boldsymbol{\theta}$ are *hyper*parameters
- The Bayesian approach (GP) automatically balances the data-fitting with the complexity penalty.

How do GPs solve the overfitting problem?

- Answer: Integrate over the function itself!
- This is associated with the Bayesian methodology.
- So, we will average out all possible function forms, under a (GP) prior!

Recap:

$$\begin{array}{rll} \text{ML:} & \underset{\mathbf{w}}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{w}, \phi(\mathbf{x})) & \text{e.g. } \mathbf{y} = \phi(\mathbf{x})^\top \mathbf{w} + \boldsymbol{\epsilon} \\ \text{Bayesian:} & \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f})\, \underbrace{p(\mathbf{f}|\mathbf{x}, \boldsymbol{\theta})}_{\text{GP prior}} & \text{e.g. } \mathbf{y} = f(\mathbf{x}, \boldsymbol{\theta}) + \boldsymbol{\epsilon} \end{array}$$

- $\boldsymbol{\theta}$ are *hyper*parameters
- The Bayesian approach (GP) automatically balances the data-fitting with the complexity penalty.

Next: More intuition on...

- What does it mean to follow a Bayesian approach?
- What does it have to do with (avoiding) overfitting and controlling model complexity?

Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.

Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.

Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.

Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.

Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.

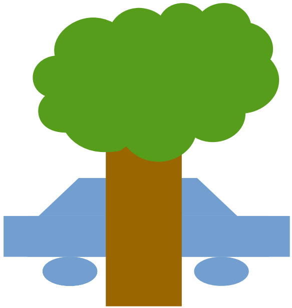
Bayesian approach

Assume a hypothesis (model) $\mathcal{M}$ and a distribution for its parameters, θ.

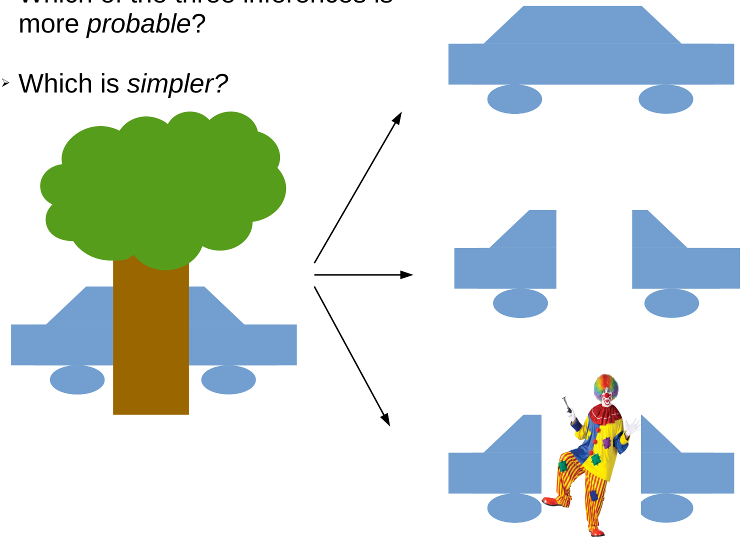
- Assume a prior distribution for our parameters, θ.
- Assume a likelihood for the observed data, D, *given* the parameters.
- Find the posterior of the parameters, given the data.
- The normaliser of the posterior is the model evidence.
- All linked through *Bayes' rule*:

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

- *Next*: See how this relates to model complexity.



- Which of the three inferences is more *probable*?
- Which is *simpler?*



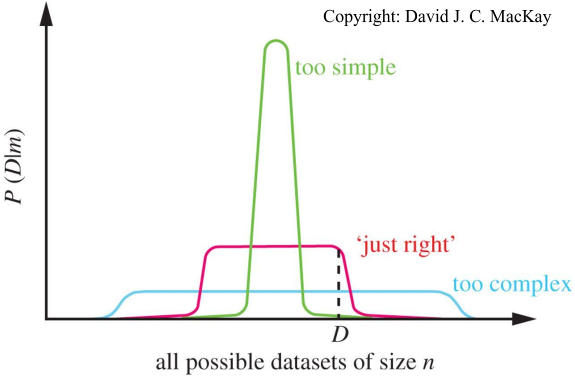
Bayes' rule again

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

(Bayesian) Occam's Razor

"A plurality is not to be posited without necessity". *W. of Ockham*

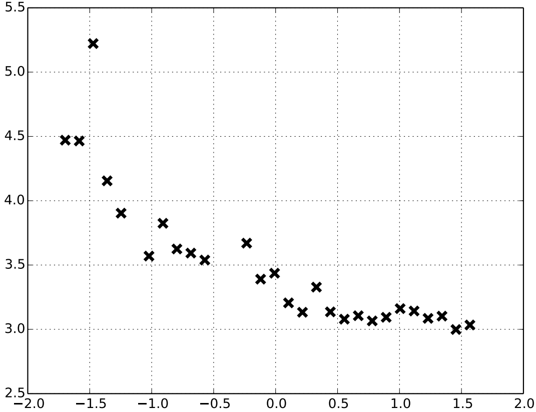
"Everything should be made as simple as possible, but not simpler". *A. Einstein*



Evidence is higher for the model that is not "unnecessarily complex" but still "explains" the data D.

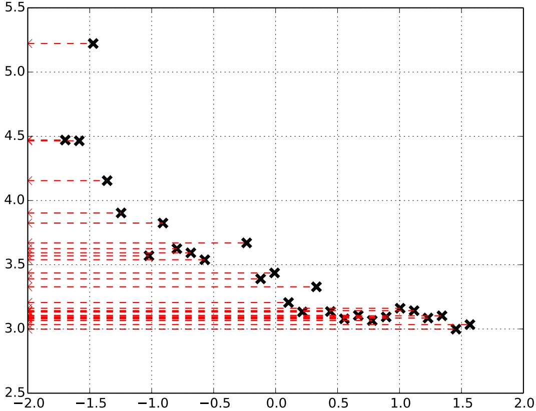
Outline

Fitting the GP-LVM



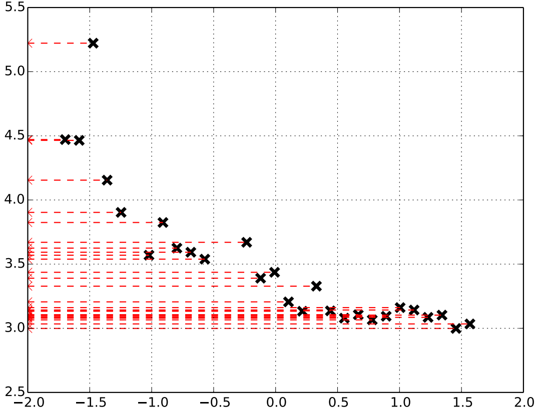
Fitting the GP-LVM

Figure credits: C. H. Ek



Fitting the GP-LVM

Figure credits: C. H. Ek



- Additional difficulty: x's are also missing!
- Improvement: Invoke the Bayesian methodology to find x's too.

Bayesian GP-LVM

GP-LVM objective:

- $\underset{\boldsymbol{\theta},\mathbf{x}}{\operatorname{argmax}}\, p(\mathbf{y}|\mathbf{x},\boldsymbol{\theta})$, where $p(\mathbf{y}|\mathbf{x},\boldsymbol{\theta}) = \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{x},\boldsymbol{\theta})$
- Bayesian w.r.t f, MAP/ML w.r.t $\mathbf{x}$.

Bayesian GP-LVM objective:

- $\underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\mathbf{y}|\boldsymbol{\theta})$, where $p(\mathbf{y}|\boldsymbol{\theta}) = \int_{\mathbf{x}} \left[\int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{x},\boldsymbol{\theta}) \right] p(\mathbf{x})$
- *fully* Bayesian.

[Titsias & Lawrence. "Bayesian GP-LVM", AISTATS 2010]

Access to $p(\mathbf{y})$ also gives us the posterior:

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$

Bayesian GP-LVM

GP-LVM objective:

- $\underset{\boldsymbol{\theta},\mathbf{x}}{\operatorname{argmax}}\ p(\mathbf{y}|\mathbf{x},\boldsymbol{\theta})$, where $p(\mathbf{y}|\mathbf{x},\boldsymbol{\theta}) = \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{x},\boldsymbol{\theta})$
- Bayesian w.r.t f, MAP/ML w.r.t $\mathbf{x}$.

Bayesian GP-LVM objective:

- $\underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p(\mathbf{y}|\boldsymbol{\theta})$, where $p(\mathbf{y}|\boldsymbol{\theta}) = \int_{\mathbf{x}} \left[\int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{x},\boldsymbol{\theta})\right] p(\mathbf{x})$
- *fully* Bayesian.

[Titsias & Lawrence. "Bayesian GP-LVM", AISTATS 2010]

Access to $p(\mathbf{y})$ also gives us the posterior:

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$

Evidence computation is intractable for the GP-LVM

$$
\begin{aligned}
p(\mathbf{y}) &= \int p(\mathbf{y}|\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{x} \\
&= \int\int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{f}\mathbf{x} \\
&= \int p(\mathbf{y}|\mathbf{f})\Big[\underbrace{\int p(\mathbf{f}|\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{x}}_{\text{Intractable!}}\Big]\mathrm{d}\mathbf{f}
\end{aligned}
$$

Intractability: $\mathbf{x}$ appears non-linearly in $p(\mathbf{f}|\mathbf{x})$, inside $\mathbf{K}^{-1}$ (and also the determinant term), where $\mathbf{K} = k(\mathbf{x}, \mathbf{x})$.

Solution to intractability: Variational approximation

- Solution: Construct an approximation of the form of a *variational lower bound* (conditioning on $\mathcal{M}, \boldsymbol{\theta}$ dropped):

$$p(\mathbf{y}) \geq \mathcal{F}.$$

- $\mathcal{F}$ is the new objective; in maximum $\rightarrow p(\mathbf{y})$.
- Since $p(\mathbf{y})$ is approximated, then $p(\mathbf{x}|\mathbf{y})$ is also approximate:

$$q(\mathbf{x}) \approx p(\mathbf{x}|\mathbf{y})$$

- Having a *posterior* for $\mathbf{x}$ is very important!
- More on these approximations in Alan's and James' talk tomorrow.

Advantages and extensions

- Training robust to overfitting (Occam's razor)
- More natural handling of missing data (*semi-described* and *semi-supervised* learning)
- Automatic detection of $\mathbf{X}$'s dimensionality
- More natural incorporation of priors for $\mathbf{X}$, e.g. *dynamics*
- Structural extensions:
  - Deep models
  - Multi-view models

Advantages and extensions

- Training robust to overfitting (Occam's razor)
- More natural handling of missing data (*semi-described* and *semi-supervised* learning)
- Automatic detection of $\mathbf{X}$'s dimensionality
- More natural incorporation of priors for $\mathbf{X}$, e.g. *dynamics*
- Structural extensions:
  - Deep models
  - Multi-view models

Automatic Relevance Determination

$\mathbf{X}$ is multidimensional: $\mathbf{X} = \{\mathbf{x}_j\}_{j=1}^q$

The EQ cov. function

$$k_{EQ}(x, x') = \alpha \exp\left(-\sum_{j=1}^{q} \frac{(x_j - x'_j)^2}{2l^2}\right)$$

The ARD cov. function

$$k_{ARD}(x, x') = \alpha \exp\left(-\sum_{j=1}^{q} \frac{(x_j - x'_j)^2}{2l_j^2}\right)$$

$$= \alpha \exp\left(-\frac{1}{2}\sum_{j=1}^{q} w_j (x_j - x'_j)^2\right)$$

Automatic Relevance Determination

$\mathbf{X}$ is multidimensional: $\mathbf{X} = \{\mathbf{x}_j\}_{j=1}^{q}$

The EQ cov. function

$$k_{EQ}(x, x') = \alpha \exp\left(-\sum_{j=1}^{q} \frac{(x_j - x'_j)^2}{2l^2}\right)$$

The ARD cov. function

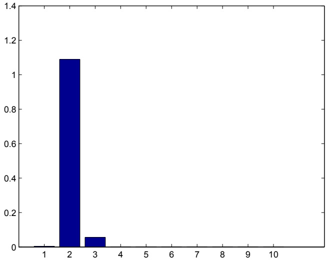
$$\begin{aligned} k_{ARD}(x, x') &= \alpha \exp\left(-\sum_{j=1}^{q} \frac{(x_j - x'_j)^2}{2l_j^2}\right) \\ &= \alpha \exp\left(-\frac{1}{2}\sum_{j=1}^{q} w_j (x_j - x'_j)^2\right) \end{aligned}$$

Automatic Relevance Determination

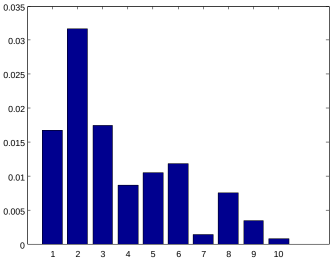
$$k_{ARD}(x, x') = \alpha e^{-\sum_{j=1}^{q} \frac{(x_j - x'_j)^2}{2l_j^2}} = \alpha e^{-\frac{1}{2}\sum_{j=1}^{q} w_j (x_j - x'_j)^2}$$

- The lengthscale l_j along input dimension j tells us how big $|x_j - x'_j|$ has to be for $|f(\mathbf{x}) - f(\mathbf{x}')|$ to be significant.
- So, when $l_j \to \infty,$ i.e. $(w_j \to 0)$, then f varies very little as a function of x_j (i.e. dimension j becomes irrelevant).
- By optimising the whole vector $\mathbf{w} = [w_1, w_2, \cdots, w_q]$ we perform automatic selection of the input features.
- In the GP-LVM case, the input features (columns of $\mathbf{X}$) correspond to *dimensions*, hence we perform automatic dimensionality detection.

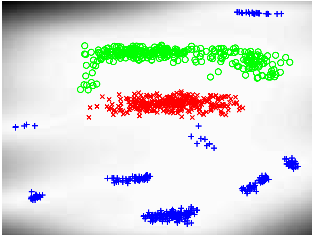
Automatic Relevance Determination



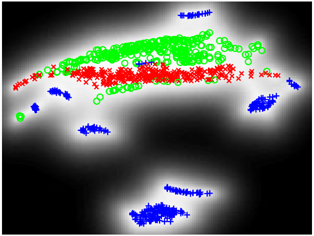
Bayesian GP-LVM with ARD



GP-LVM with ARD



Bayesian GP-LVM, $q = 10$ (2D projection)



GP-LVM, no ARD, $q = 2$

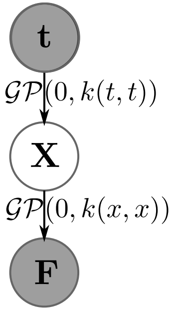
Advantages and extensions

- Training robust to overfitting (Occam's razor)
- More natural handling of missing data (*semi-described* and *semi-supervised* learning)
- Automatic detection of $\mathbf{X}$'s dimensionality
- More natural incorporation of priors for $\mathbf{X}$, e.g. *dynamics*
- Structural extensions:
  - Deep models
  - Multi-view models

Bayes' rule again

$$p(\theta|D,\mathcal{M}) = \frac{p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}{p(D|\mathcal{M}) = \int_\theta p(D|\theta,\mathcal{M})p(\theta|\mathcal{M})}$$

Latent space priors



$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$

- In general, we have $p(\mathbf{x}|\boldsymbol{\theta}_x)$
- If $\mathbf{y}$ is a timeseries, then we might want to make $\mathbf{x}$ to be also a timeseries
- We can even make $\mathbf{x}$ to be a function: $\mathbf{x} = x(\mathbf{t})$
- Then we can put a GP prior on it: $x(t) \sim \mathcal{GP}(0, k(t,t))$

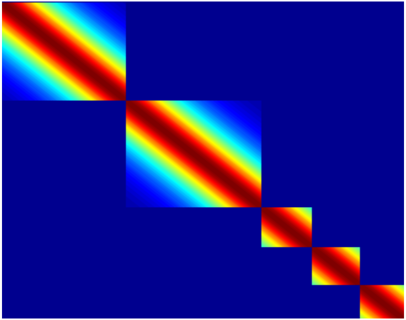
Latent space priors

Video modelling examples...

- https://youtu.be/i9TEoYxaBxQ
- https://youtu.be/mUY1XHPnoCU

Dynamics with multiple sequences

- If $\mathbf{Y}$ consists of multiple independent sequences, $\mathbf{Y} = \left[\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \cdots \mathbf{Y}^{(s)}\right]$, then the time-stamp vector will also be something like $\mathbf{t} = \left[\mathbf{t}^{(1)}, \mathbf{t}^{(2)}, \cdots \mathbf{t}^{(s)}\right]$.
- Then, the covariance matrix from $k_x(\mathbf{t}, \mathbf{t})$ on the dynamics will look something like this:

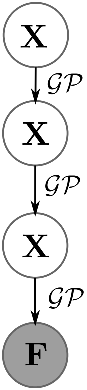


- Mocap demo: https://youtu.be/fHDWloJtgk8

Outline

- Deep GPs
- Multi-view: MRD
- Missing Data (uncertainty)

Deep Gaussian processes

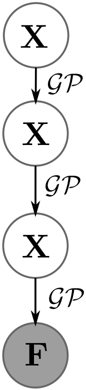


- Now recurse the stacked construction

$$
\begin{aligned}
f(\mathbf{x}) &\rightarrow \text{GP} \\
f(x(\mathbf{t})) &\rightarrow \text{stacked GP} \\
f(x_2(\mathbf{x}_1)) &\rightarrow \text{stacked GP} \\
f(x(x(x\cdots(\mathbf{x}_1)))) &\rightarrow \text{deep GP}
\end{aligned}
$$

- The variational approximation changes only a little
- Uncertainty modelled "everywhere"!

Deep Gaussian processes

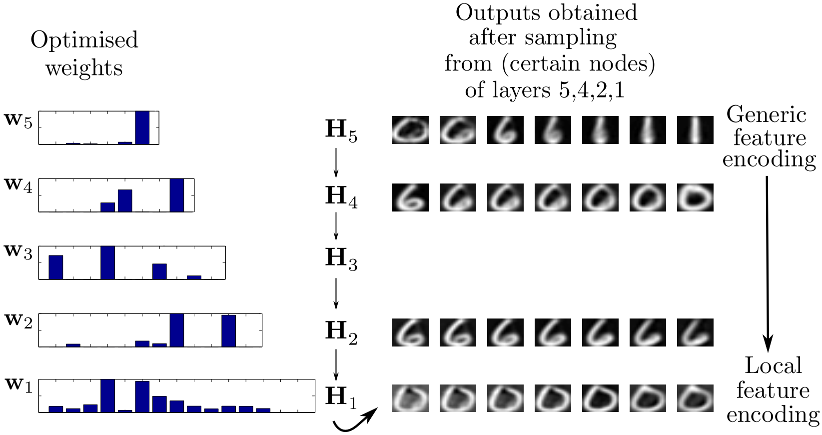


- Now recurse the stacked construction

$$
\begin{aligned}
f(\mathbf{x}) &\rightarrow \text{GP} \\
f(x(\mathbf{t})) &\rightarrow \text{stacked GP} \\
f(x_2(\mathbf{x}_1)) &\rightarrow \text{stacked GP} \\
f(x(x(x\cdots(\mathbf{x}_1)))) &\rightarrow \text{deep GP}
\end{aligned}
$$

- The variational approximation changes only a little
- Uncertainty modelled "everywhere"!

Deep GP: Digits example

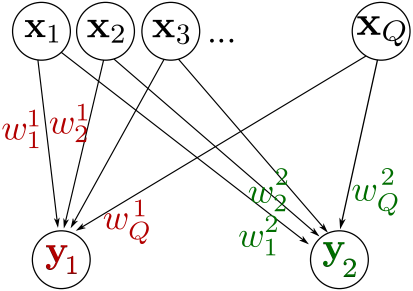


Demo: https://youtu.be/E8-vxt8wxBU

- Deep GPs
- Multi-view: MRD
- Missing Data (uncertainty)

Multi-view modelling (Expand the model "horizontally")

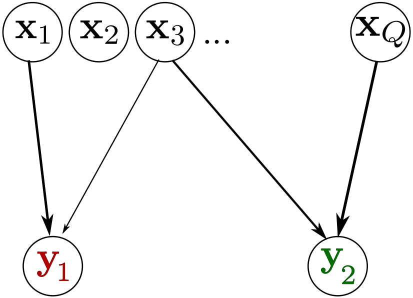
- Multi-view data arise from multiple information sources. These sources naturally contain some overlapping, or *shared* signal (since they describe the same "phenomenon"), but also have some *private* signal.
- Idea: Model such data via latent variable models



Demo: https://youtu.be/rIPX3CIOhKY

Multi-view modelling (Expand the model "horizontally")

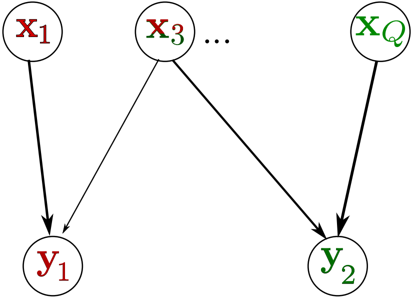
- Multi-view data arise from multiple information sources. These sources naturally contain some overlapping, or *shared* signal (since they describe the same "phenomenon"), but also have some *private* signal.
- Idea: Model such data via latent variable models



Demo: https://youtu.be/rIPX3CIOhKY

Multi-view modelling (Expand the model "horizontally")

- Multi-view data arise from multiple information sources. These sources naturally contain some overlapping, or *shared* signal (since they describe the same "phenomenon"), but also have some *private* signal.
- Idea: Model such data via latent variable models



Demo: https://youtu.be/rIPX3CIOhKY

Summary

- Bayesian modelling automatically balances fitting with complexity
- Latent variables are a powerful addition to our GP modelling toolbox (Neil's talk)
- Similarly to how the mapping $f : x \mapsto f(x)$ is treated in a Bayesian way in GPs, we can treat x also in a Bayesian way in GP-LVM
- Many advantages and extensions arise.
- The key to obtaining those is the principled *propagation of uncertainty* across all stages of the graphical model.

Thanks!

Questions?

References:

- N. D. Lawrence (2006) “The Gaussian process latent variable model” Technical Report no CS-06-03, The University of Sheffield, Department of Computer Science
- N. D. Lawrence (2006) “Probabilistic dimensional reduction with the Gaussian process latent variable model” (talk)
- C. E. Rasmussen (2008), “Learning with Gaussian Processes”, Max Planck Institute for Biological Cybernetics, Published: Feb. 5, 2008 (Videolectures.net)
- Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning. MIT Press, Cambridge, MA, 2006. ISBN 026218253X.
- M. K. Titsias (2010), “Bayesian Gaussian process latent variable model”, AISTATS 2010
- A. Damianou, M. K. Titsias and N. D. Lawrence (2011), “Variational Gaussian process dynamical systems”, NIPS 2011
- A. Damianou, C. H. Ek, M. K. Titsias and N. D. Lawrence (2012), “Manifold Relevance Determination”, ICML 2012
- A. Damianou and N. D. Lawrence (2013), “Deep Gaussian processes”, AISTATS 2013
- A. Damianou (2015), “Deep Gaussian processes and variational propagation of uncertainty”, PhD Thesis