Deep Gaussian Processes

Neil Lawrence

GPRS
14th February 2014

Outline

Deep GPs

Bayesian GP-LVM

Conclusions

Outline

Deep GPs

Bayesian GP-LVM

Conclusions

Hierarchical GP-LVM

(Lawrence and Moore, 2007)

Stacking Gaussian Processes

- Regressive dynamics provides a simple hierarchy.
  - The input space of the GP is governed by another GP.
- By stacking GPs we can consider more complex hierarchies.
- Ideally we should marginalise latent spaces
  - In practice we seek MAP solutions.

Two Correlated Subjects

(Lawrence and Moore, 2007)

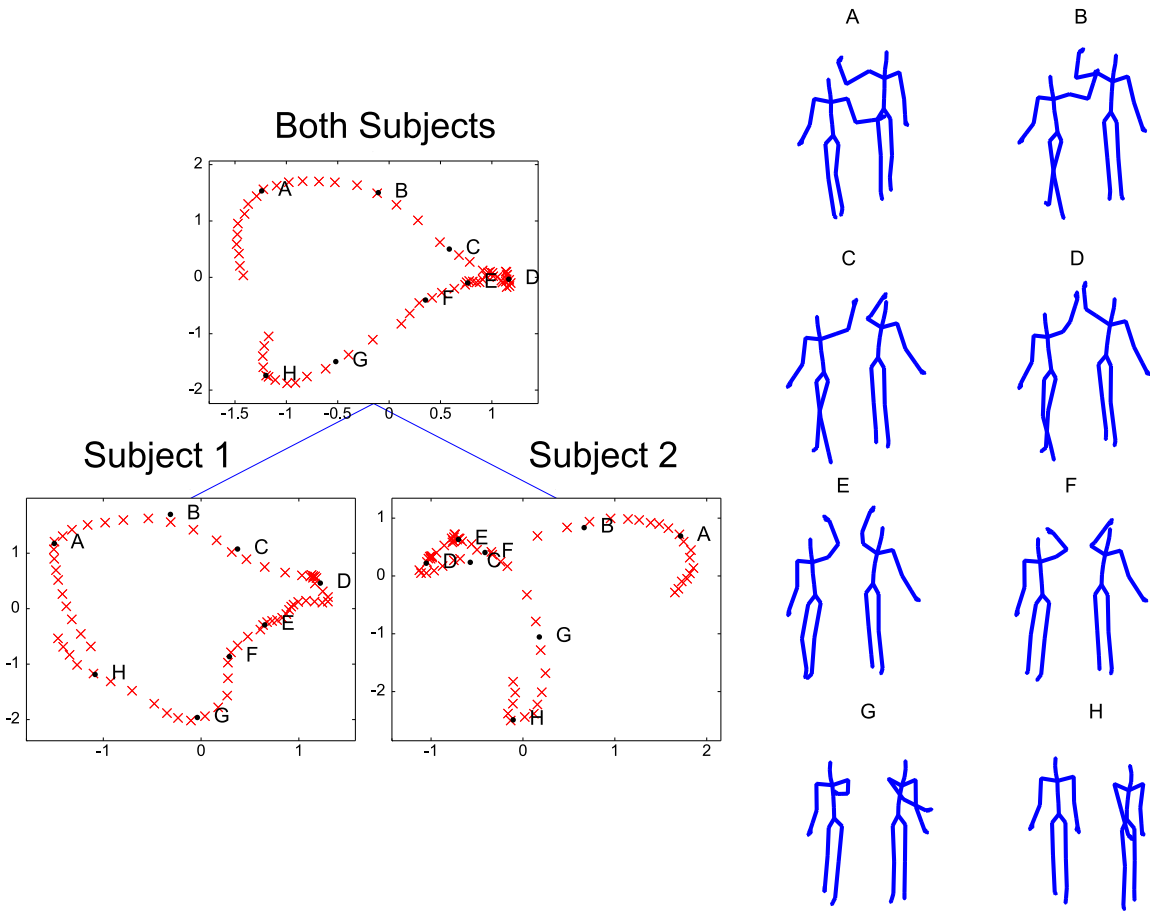


Figure : Hierarchical model of a 'high five'.

Within Subject Hierarchy

(Lawrence and Moore, 2007)

Decomposition of Body

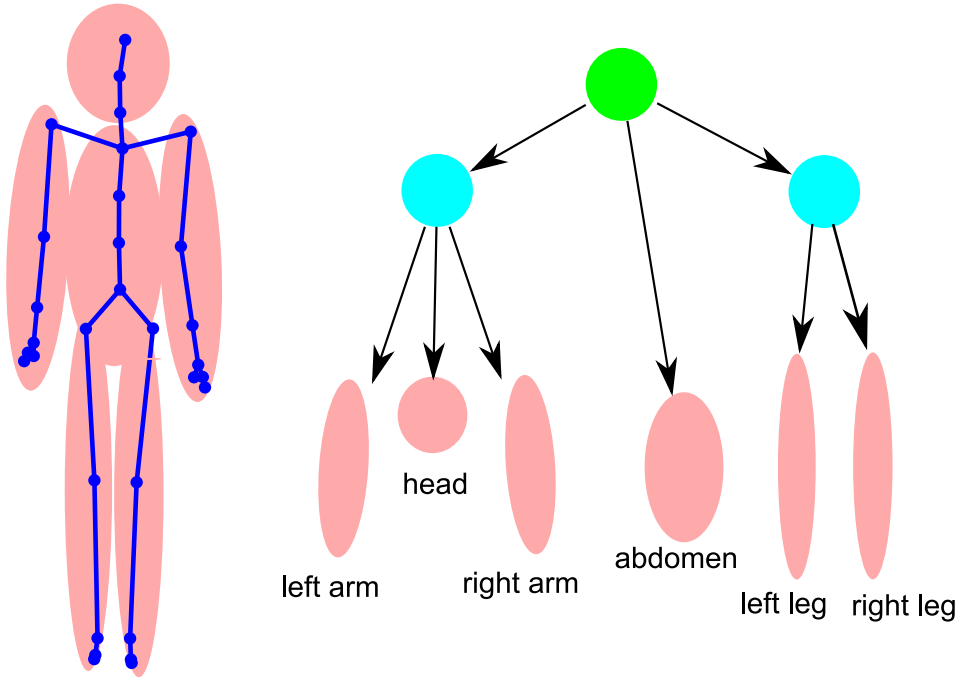


Figure : Decomposition of a subject.

Single Subject Run/Walk

(Lawrence and Moore, 2007)

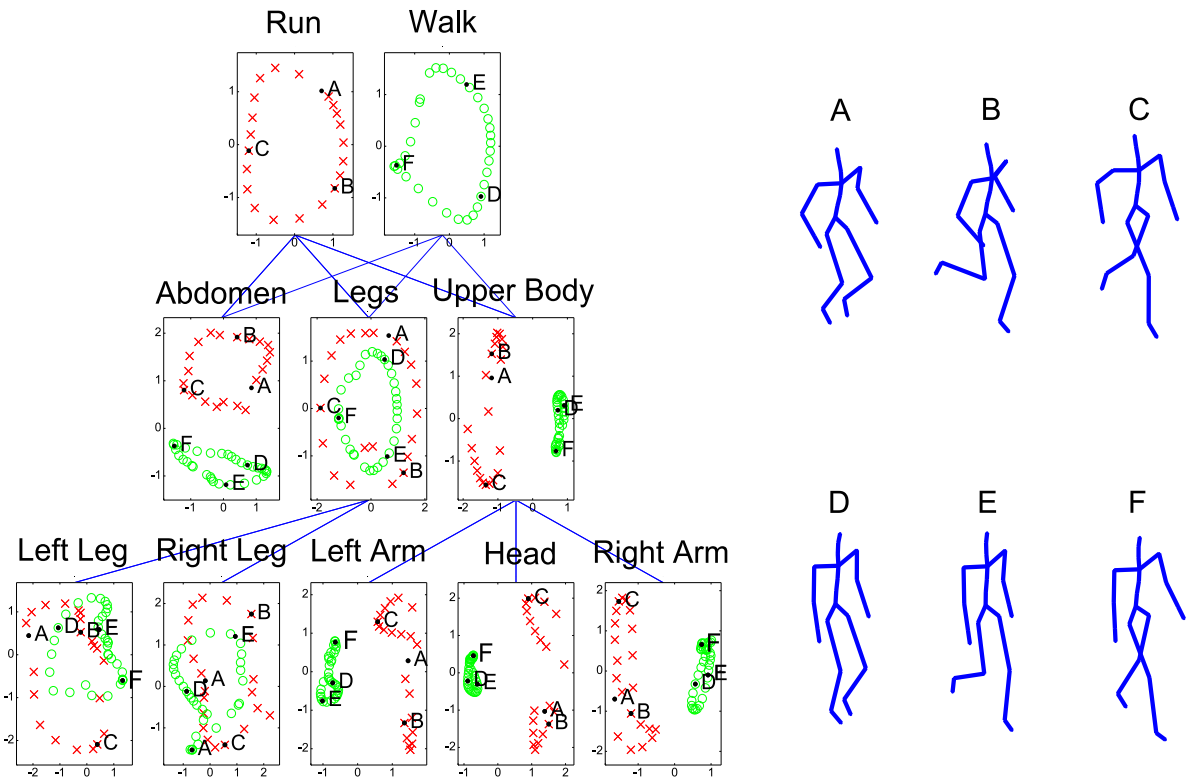


Figure : Hierarchical model of a walk and a run.

Return

Learning in Larger Datasets

(Lawrence, 2007; Titsias, 2009)

- Complexity of standard GP:
  - $O(n^3)$ in computation.
  - $O(n^2)$ in storage.
- Via low rank representations of covariance:
  - $O(nm^2)$ in computation.
  - $O(nm)$ in storage.
- Where m is user chosen number of *inducing* variables. They give the rank of the resulting covariance.

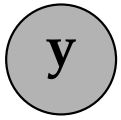
Inducing Variable Approximations

- Date back to (Williams and Seeger, 2001; Smola and Bartlett, 2001; Csató and Opper, 2002; Seeger et al., 2003; Snelson and Ghahramani, 2006). See Quiñonero Candela and Rasmussen (2005) for a review.
- We follow variational perspective of (Titsias, 2009).
- This is an augmented variable method, followed by a collapsed variational approximation (King and Lawrence, 2006; Hensman et al., 2012).

Augmented Variable Model

Augment standard GP model with a set of m new inducing variables, $\mathbf{u}$.

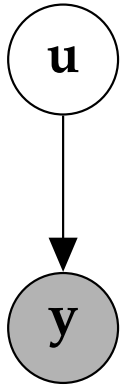
$$p(\mathbf{y}) = \int p(\mathbf{y}, \mathbf{u}) \mathrm{d}\mathbf{u}$$



Augmented Variable Model

Augment standard GP model with a set of m new inducing variables, $\mathbf{u}$.

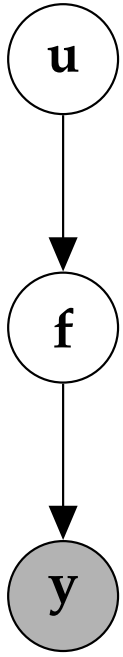
$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u}$$



Augmented Variable Model

Assume that relationship is through **f**.

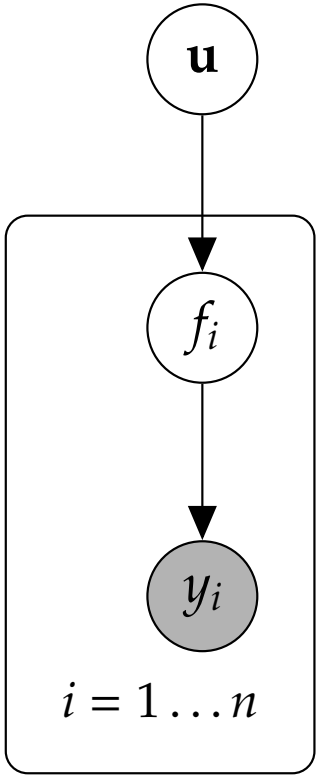
$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{u}$$



Augmented Variable Model

Very often likelihood factorizes.

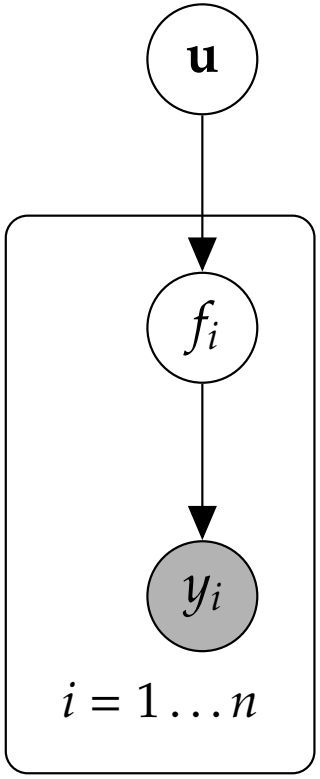
$$p(\mathbf{y}) = \int \prod_{i=1}^{n} p(y_i|f_i)p(\mathbf{f}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{u}$$



Augmented Variable Model

Focus on integral over $\mathbf{f}$.

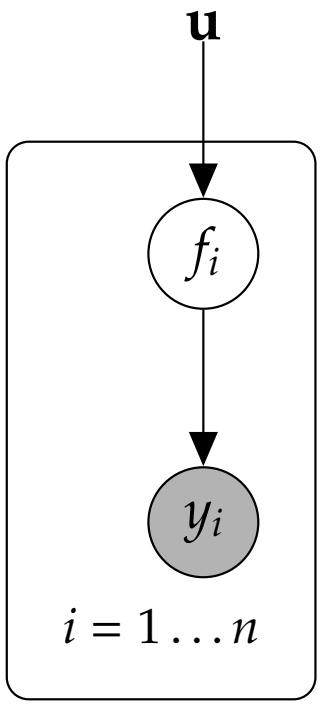
$$p(\mathbf{y}) = \int \int \prod_{i=1}^{n} p(y_i|f_i)p(\mathbf{f}|\mathbf{u})\mathrm{d}\mathbf{f}p(\mathbf{u})\mathrm{d}\mathbf{u}$$



Augmented Variable Model

Focus on integral over $\mathbf{f}$.

$$p(\mathbf{y}|\mathbf{u}) = \int \prod_{i=1}^{n} p(y_i|f_i)p(\mathbf{f}|\mathbf{u})\mathrm{d}\mathbf{f}$$



Variational Bound on $p(\mathbf{y}|\mathbf{u})$

$$\log p(\mathbf{y}|\mathbf{u}) = \log \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u})\mathrm{d}\mathbf{f}$$
$$\geq \int q(\mathbf{f}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u})}{q(\mathbf{f})}\mathrm{d}\mathbf{f}$$

- For variational approximation of (Titsias, 2009) set $q(\mathbf{f}) = p(\mathbf{f}|\mathbf{u})$,

$$\log p(\mathbf{y}|\mathbf{u}) \geq \log \int p(\mathbf{f}|\mathbf{u}) \log p(\mathbf{y}|\mathbf{f})\mathrm{d}\mathbf{f}.$$

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \int p(\mathbf{f}|\mathbf{u}) \log p(\mathbf{y}|\mathbf{f})\mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \int p(\mathbf{f}|\mathbf{u}) \log \prod_{i=1}^{n} p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \int p(\mathbf{f}|\mathbf{u}) \log \prod_{i=1}^{n} p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \int p(\mathbf{f}|\mathbf{u}) \sum_{i=1}^{n} \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \int p(\mathbf{f}|\mathbf{u}) \sum_{i=1}^{n} \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \sum_{i=1}^{n} \int p(f_i|\mathbf{u}) \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \exp \sum_{i=1}^{n} \int p(f_i|\mathbf{u}) \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \prod_{i=1}^{n} \exp \int p(f_i|\mathbf{u}) \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \prod_{i=1}^{n} \exp \int p(f_i|\mathbf{u}) \log p(y_i|f_i) \mathrm{d}\mathbf{f}.$$

- Then the bound factorizes.

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \prod_{i=1}^{n} \exp \left\langle \log p(y_i|f_i) \right\rangle_{p(f_i|\mathbf{u})}$$

- Then the bound factorizes.

Factorizing Likelihoods

- If the likelihood, $p(\mathbf{y}|\mathbf{f})$, factorizes

$$p(\mathbf{y}|\mathbf{u}) \geq \prod_{i=1}^{n} \exp \left\langle \log p(y_i|f_i) \right\rangle_{p(f_i|\mathbf{u})}$$

- Then the bound factorizes.

Gaussian $p(y_i|f_i)$

For Gaussian likelihoods:

$$\left\langle \log p(y_i|f_i) \right\rangle_{p(f_i|\mathbf{u})} = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left(y_i - \langle f_i \rangle\right)^2 - \frac{1}{2\sigma^2}\left(\left\langle f_i^2 \right\rangle - \langle f_i \rangle^2\right)$$

Gaussian $p(y_i|f_i)$

For Gaussian likelihoods:

$$\langle \log p(y_i|f_i) \rangle_{p(f_i|\mathbf{u})} = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left(y_i - \langle f_i \rangle\right)^2 - \frac{1}{2\sigma^2}\left(\left\langle f_i^2 \right\rangle - \langle f_i \rangle^2\right)$$

Implying:

$$p(y_i|\mathbf{u}) \geq \exp\langle \log c_i \rangle \, \mathcal{N}\left(y_i | \langle f_i \rangle, \sigma^2\right)$$

Gaussian Process Over $\mathbf{f}$ and $\mathbf{u}$

Define:

$$q_{i,i} = \text{var}_{p(f_i|\mathbf{u})}(f_i) = \left\langle f_i^2 \right\rangle_{p(f_i|\mathbf{u})} - \left\langle f_i \right\rangle^2_{p(f_i|\mathbf{u})}$$

We can write:

$$c_i = \exp\left(-\frac{q_{i,i}}{2\sigma^2}\right)$$

If joint distribution of $p(\mathbf{f}, \mathbf{u})$ is Gaussian then:

$$q_{i,i} = k_{i,i} - \mathbf{k}_{i,\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{k}_{i,\mathbf{u}}$$

c_i is not a function of $\mathbf{u}$ but *is* a function of $\mathbf{X}_\mathbf{u}$).

Lower Bound on Likelihood

Substitute variational bound into marginal likelihood:

$$p(\mathbf{y}) \geq \prod_{i=1}^{n} c_i \int \mathcal{N}\left(\mathbf{y} | \langle \mathbf{f} \rangle, \sigma^2 \mathbf{I}\right) p(\mathbf{u}) \mathrm{d}\mathbf{u}$$

Note that:

$$\langle \mathbf{f} \rangle_{p(\mathbf{f}|\mathbf{u})} = \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}$$

is *linearly* dependent on $\mathbf{u}$.

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \int \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \sigma^2\right)\mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)\mathrm{d}\mathbf{u}$$

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

Maximize log of the bound to find covariance function parameters,

$$L \geq \sum_{i=1}^{n} \log c_i + \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f},}\right)$$

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

Maximize log of the bound to find covariance function parameters,

$$L \geq \sum_{i=1}^{n} \log c_i + \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f},}\right)$$

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

Maximize log of the bound to find covariance function parameters,

$$L \approx \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f},}\right)$$

- If the bound is normalized, the c_i terms are removed.

Deterministic Training Conditional

Making the marginalization of $\mathbf{u}$ straightforward. In the Gaussian case:

$$p(\mathbf{u}) = \mathcal{N}\left(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right)$$

$$\int p(\mathbf{y}|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u} \geq \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \sigma^2\mathbf{I} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

Maximize log of the bound to find covariance function parameters,

- If the bound is normalized, the c_i terms are removed.
- This results in the projected process approximation (Rasmussen and Williams, 2006) or DTC (Quiñonero Candela and Rasmussen, 2005). Proposed by (Smola and Bartlett, 2001; Seeger et al., 2003; Csató and Opper, 2002; Csató, 2002).

Outline

Deep GPs

Bayesian GP-LVM

Conclusions

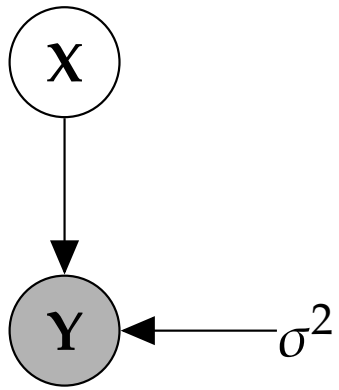
Selecting Data Dimensionality

- GP-LVM Provides probabilistic non-linear dimensionality reduction.
- How to select the dimensionality?
- Need to estimate marginal likelihood.
- In standard GP-LVM it increases with increasing q.

Integrate Mapping Function and Latent Variables

Bayesian GP-LVM

- Start with a standard GP-LVM.

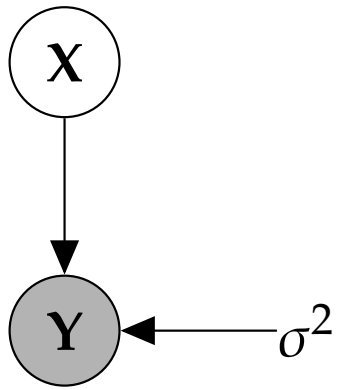


$$p\left(\mathbf{Y}|\mathbf{X}\right) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

Integrate Mapping Function and Latent Variables

Bayesian GP-LVM

- Start with a standard GP-LVM.
- Apply standard latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.

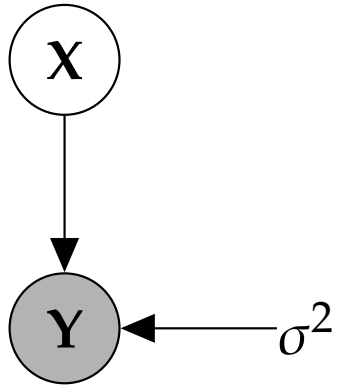


$$p\left(\mathbf{Y}|\mathbf{X}\right) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

Integrate Mapping Function and Latent Variables

Bayesian GP-LVM

- Start with a standard GP-LVM.
- Apply standard latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out *latent variables*.



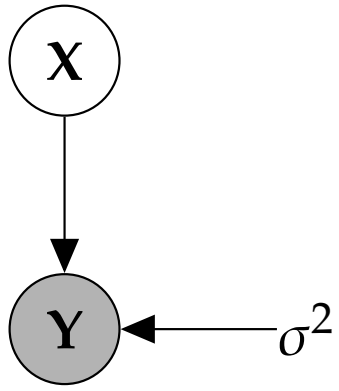
$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

$$p(\mathbf{X}) = \prod_{j=1}^{q} \mathcal{N}\left(\mathbf{x}_{:,j}|\mathbf{0}, \alpha_i^{-2}\mathbf{I}\right)$$

Integrate Mapping Function and Latent Variables

Bayesian GP-LVM

- Start with a standard GP-LVM.
- Apply standard latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out *latent variables*.
  - Unfortunately integration is intractable.



$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{p} \mathcal{N}\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

$$p(\mathbf{X}) = \prod_{j=1}^{q} \mathcal{N}\left(\mathbf{x}_{:,j}|\mathbf{0}, \alpha_i^{-2}\mathbf{I}\right)$$

$$p(\mathbf{Y}|\boldsymbol{\alpha}) = ??$$

Standard Variational Approach Fails

- Standard variational bound has the form:

$$\mathcal{L} = \langle \log p(\mathbf{y}|\mathbf{X}) \rangle_{q(\mathbf{X})} + \mathrm{KL}\left(q(\mathbf{X}) \| p(\mathbf{X})\right)$$

Standard Variational Approach Fails

- Standard variational bound has the form:

$$\mathcal{L} = \left\langle \log p(\mathbf{y}|\mathbf{X}) \right\rangle_{q(\mathbf{X})} + \mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X})\right)$$

- Requires expectation of $\log p(\mathbf{y}|\mathbf{X})$ under $q(\mathbf{X})$.

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^\top \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}\right| - \frac{n}{2}\log 2\pi$$

Standard Variational Approach Fails

- Standard variational bound has the form:

$$\mathcal{L} = \left\langle \log p(\mathbf{y}|\mathbf{X}) \right\rangle_{q(\mathbf{X})} + \mathrm{KL}\left(q(\mathbf{X}) \,\|\, p(\mathbf{X})\right)$$

- Requires expectation of $\log p(\mathbf{y}|\mathbf{X})$ under $q(\mathbf{X})$.

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^\top \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}\right| - \frac{n}{2}\log 2\pi$$

- Extremely difficult to compute because $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is dependent on $\mathbf{X}$ and appears in the inverse.

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$p(\mathbf{y}) \geq \prod_{i=1}^{n} c_i \int \mathcal{N}\left(\mathbf{y}_i \middle| \langle \mathbf{f} \rangle, \sigma^2 \mathbf{I}\right) p(\mathbf{u}) \mathrm{d}\mathbf{u}$$

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$p(\mathbf{y}|\mathbf{X}) \geq \prod_{i=1}^{n} c_i \int \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{u})\mathrm{d}\mathbf{u}$$

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$\int p(\mathbf{y}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X} \geq \int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}p(\mathbf{u})\mathrm{d}\mathbf{u}$$

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$\int p(\mathbf{y}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X} \geq \int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}| \left\langle \mathbf{f} \right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}p(\mathbf{u})\mathrm{d}\mathbf{u}$$

- Apply variational lower bound to the inner integral.

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$\int p(\mathbf{y}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X} \geq \int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}p(\mathbf{u})\mathrm{d}\mathbf{u}$$

- Apply variational lower bound to the inner integral.

$$\int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}$$

$$\geq \left\langle \sum_{i=1}^{n} \log c_i \right\rangle_{q(\mathbf{X})} + \left\langle \log \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right)\right\rangle_{q(\mathbf{X})} + \mathrm{KL}\left(q(\mathbf{X})\,||\,p(\mathbf{X})\right)$$

Variational Bayesian GP-LVM

- Consider collapsed variational bound,

$$\int p(\mathbf{y}|\mathbf{X})p(\mathbf{X})\mathrm{d}\mathbf{X} \geq \int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}p(\mathbf{u})\mathrm{d}\mathbf{u}$$

- Apply variational lower bound to the inner integral.

$$\int \prod_{i=1}^{n} c_i \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right) p(\mathbf{X})\mathrm{d}\mathbf{X}$$

$$\geq \left\langle \sum_{i=1}^{n} \log c_i \right\rangle_{q(\mathbf{X})} + \left\langle \log \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X})}, \sigma^2\mathbf{I}\right)\right\rangle_{q(\mathbf{X})} + \mathrm{KL}\left(q(\mathbf{X})\,||\,p(\mathbf{X})\right)$$

- Which is analytically tractable for Gaussian $q(\mathbf{X})$ and some covariance functions.

Required Expectations

- Need expectations under $q(\mathbf{X})$ of:

$$\log c_i = \frac{1}{2\sigma^2}\left[k_{i,i} - \mathbf{k}_{i,\mathbf{u}}^\top \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{k}_{i,\mathbf{u}}\right]$$

and

$$\log \mathcal{N}\left(\mathbf{y}|\left\langle \mathbf{f}\right\rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{Y})}, \sigma^2\mathbf{I}\right) = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left(y_i - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}\right)^2$$

- This requires the expectations

$$\left\langle \mathbf{K}_{\mathbf{f},\mathbf{u}}\right\rangle_{q(\mathbf{X})}$$

and

$$\left\langle \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right\rangle_{q(\mathbf{X})}$$

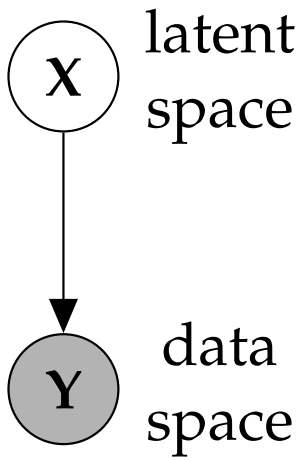
which can be computed analytically for some covariance functions.

Priors for Latent Space

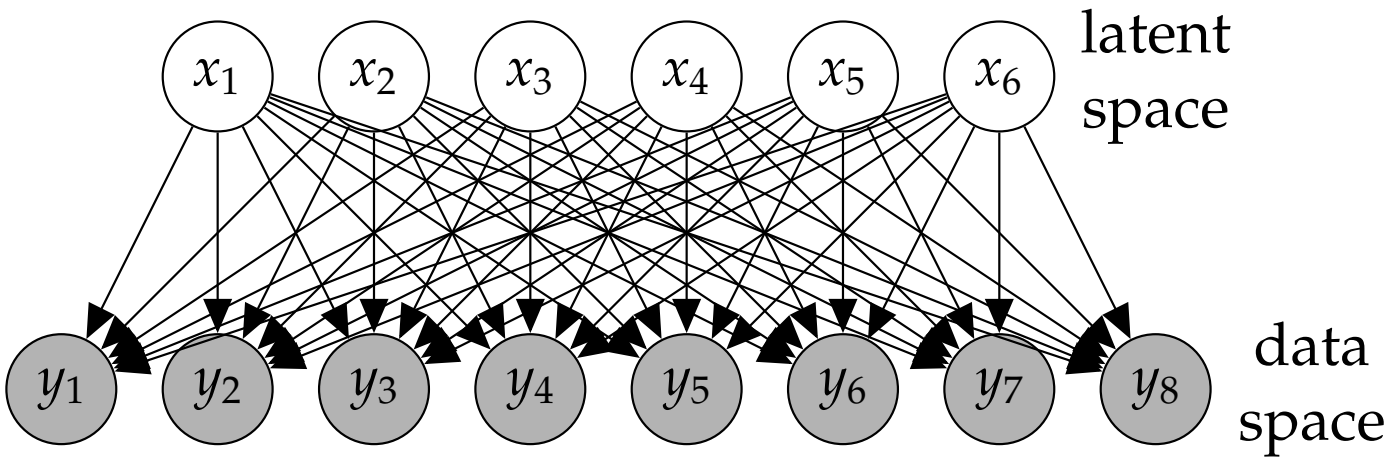
Titsias and Lawrence (2010)

- Variational marginalization of $\mathbf{X}$ allows us to learn parameters of $p(\mathbf{X})$.
- Standard GP-LVM where $\mathbf{X}$ learnt by MAP, this is not possible (see e.g. Wang et al., 2008).
- First example: learn the dimensionality of latent space.

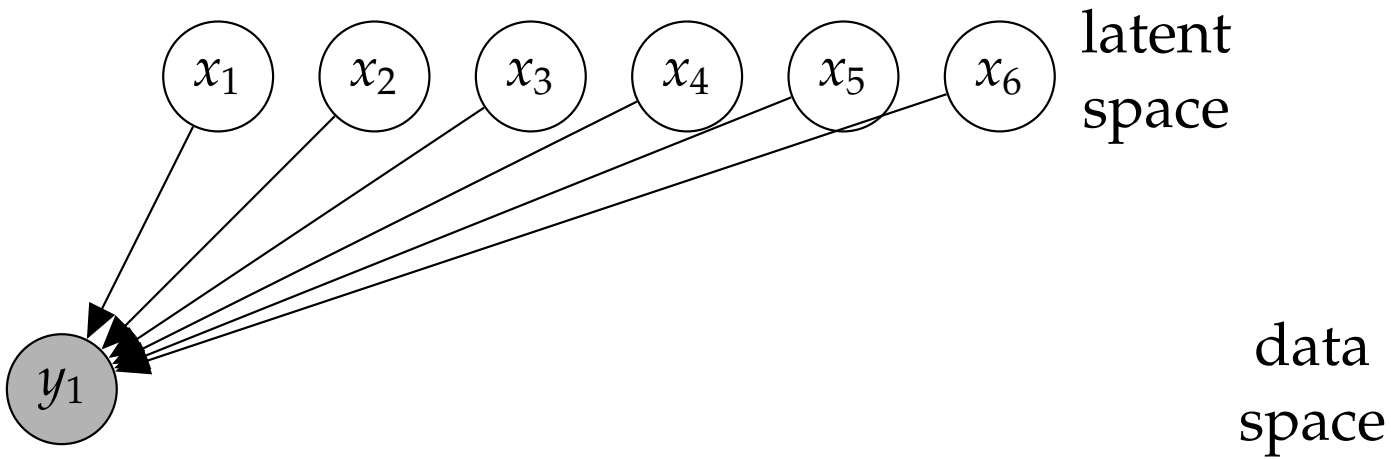
Graphical Representations of GP-LVM



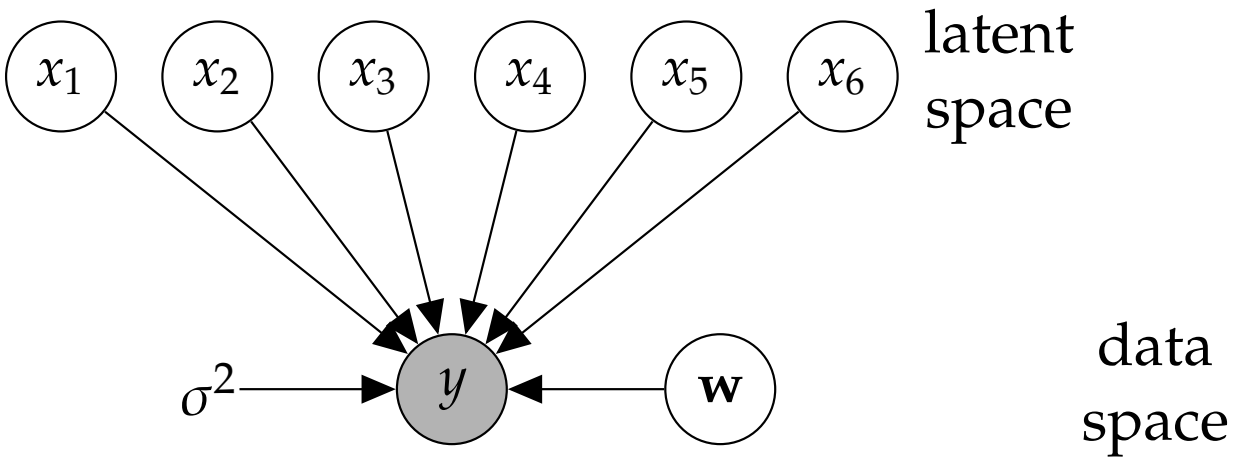
Graphical Representations of GP-LVM



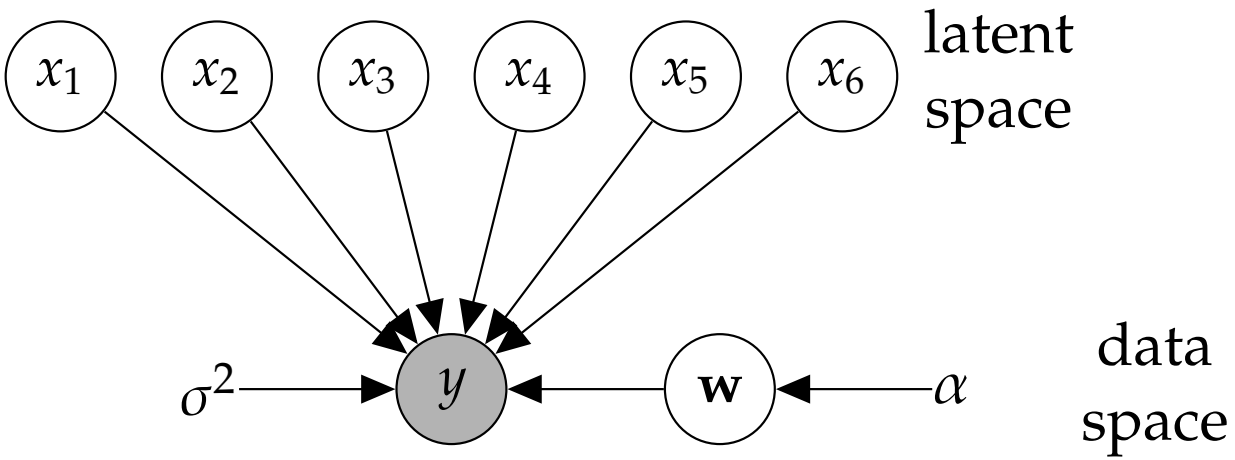
Graphical Representations of GP-LVM



Graphical Representations of GP-LVM



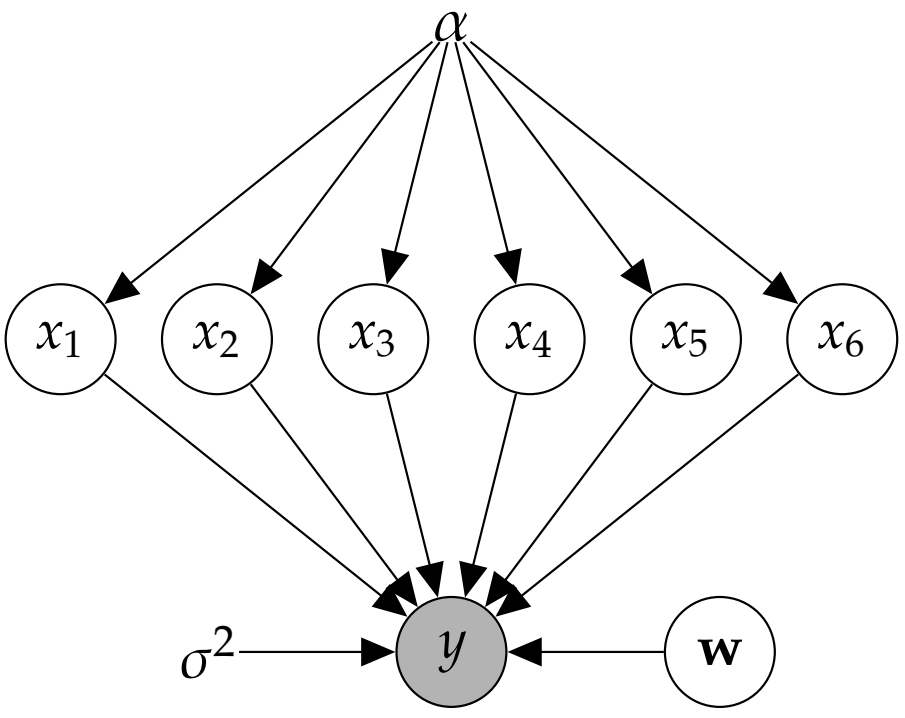
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I}) \quad \mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$$

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

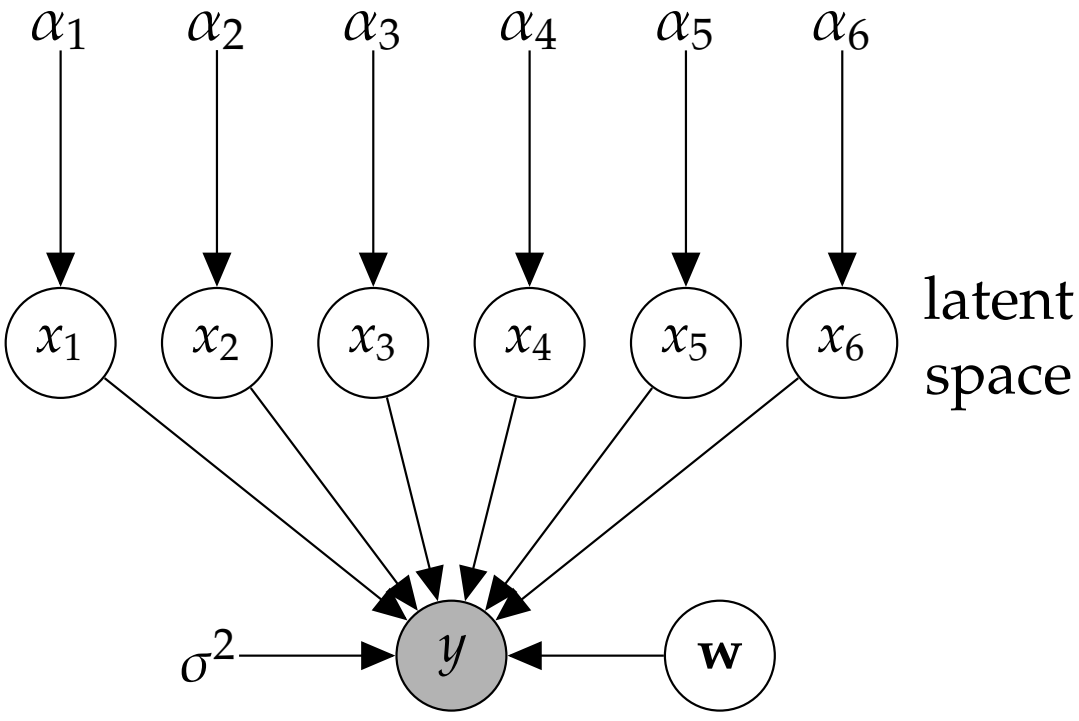
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad \mathbf{x} \sim \mathcal{N}(0, \alpha\mathbf{I})$$

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

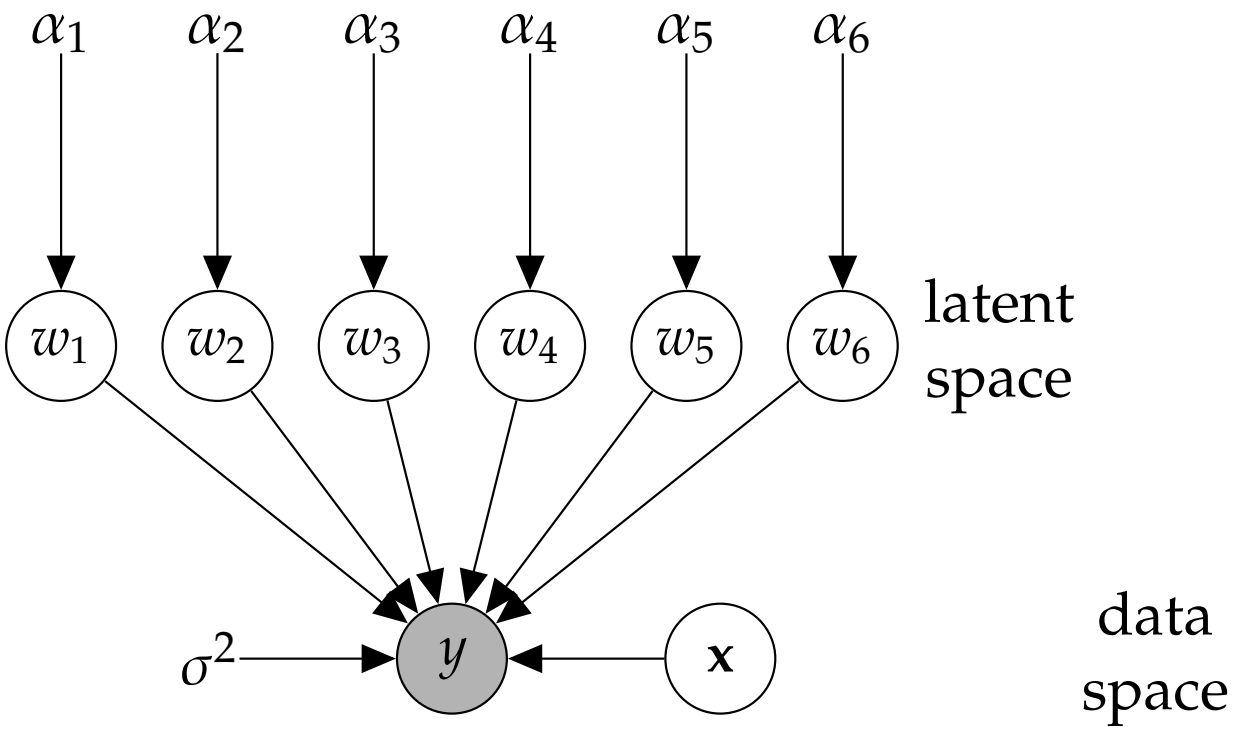
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad x_i \sim \mathcal{N}(0, \alpha_i)$$

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

Graphical Representations of GP-LVM



$$w_i \sim \mathcal{N}(0, \alpha_i) \quad \mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

Non-linear $f(\mathbf{x})$

- In linear case equivalence because $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$

$$p(w_i) \sim \mathcal{N}(\mathbf{0}, \alpha_i)$$

- In non linear case, need to scale columns of $\mathbf{X}$ in prior for $f(\mathbf{x})$.
- This implies scaling columns of $\mathbf{X}$ in covariance function

$$k(\mathbf{x}_{i,:}, \mathbf{x}_{j,:}) = \exp\left(-\frac{1}{2}(\mathbf{x}_{:,i} - \mathbf{x}_{:,j})^\top \mathbf{A}(\mathbf{x}_{:,i} - \mathbf{x}_{:,j})\right)$$

$\mathbf{A}$ is diagonal with elements α_i^2. Now keep prior spherical

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

- Covariance functions of this type are known as ARD (see e.g. Neal, 1996; MacKay, 2003; Rasmussen and Williams, 2006).

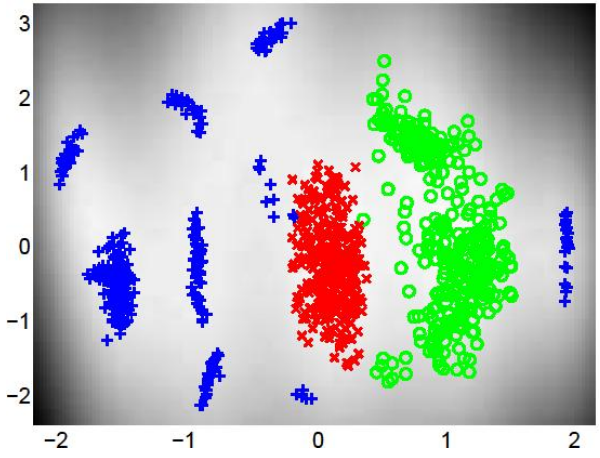
Automatic dimensionality detection

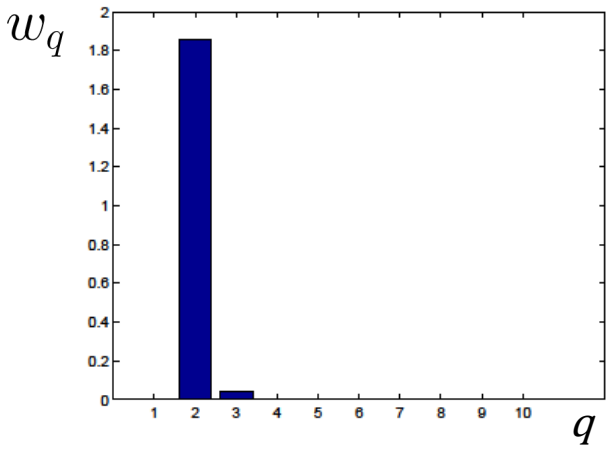
- Achieved by employing an *Automatic Relevance Determination (ARD)* covariance function for the prior on the GP mapping

- $f \sim GP(\mathbf{0}, k_f)$ with

$$k_f(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\frac{1}{2}\sum_{q=1}^{Q} w_q (x_{i,q} - x_{j,q})^2\right)$$

- Example



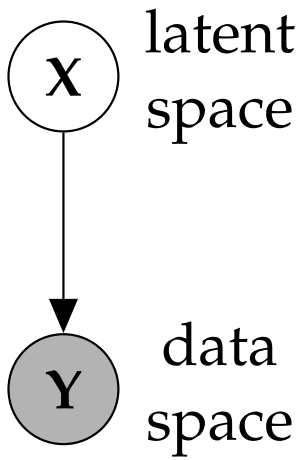


Priors for Latent Space

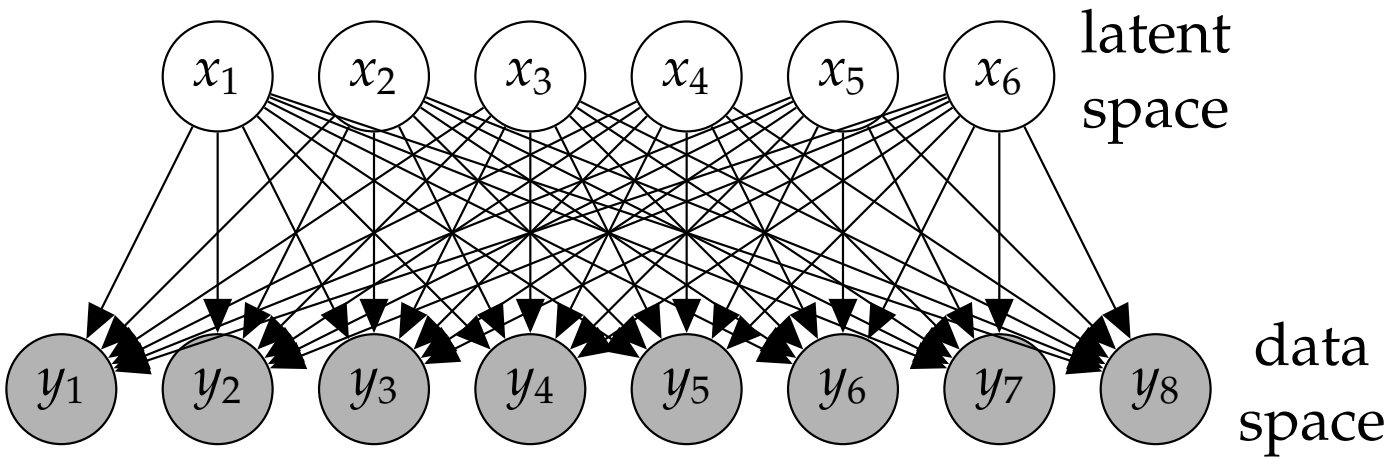
Titsias and Lawrence (2010)

- Variational marginalization of $\mathbf{X}$ allows us to learn parameters of $p(\mathbf{X})$.
- Standard GP-LVM where $\mathbf{X}$ learnt by MAP, this is not possible (see e.g. Wang et al., 2008).
- First example: learn the dimensionality of latent space.

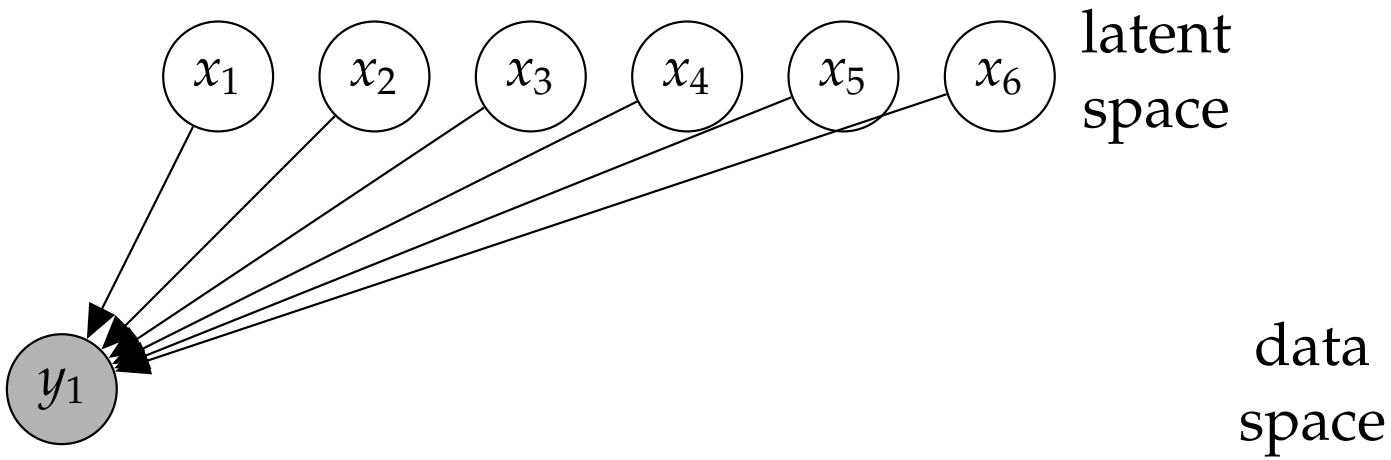
Graphical Representations of GP-LVM



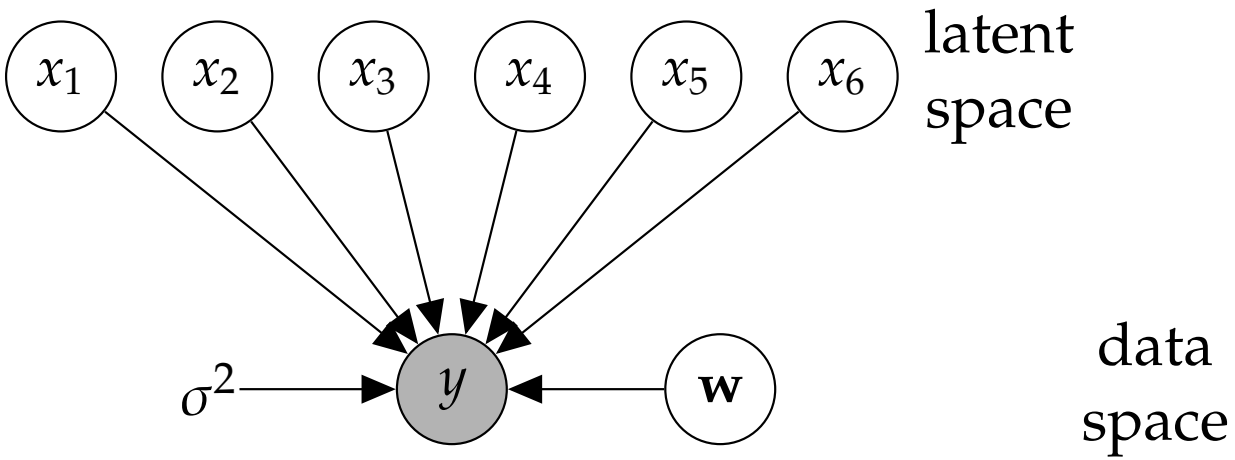
Graphical Representations of GP-LVM



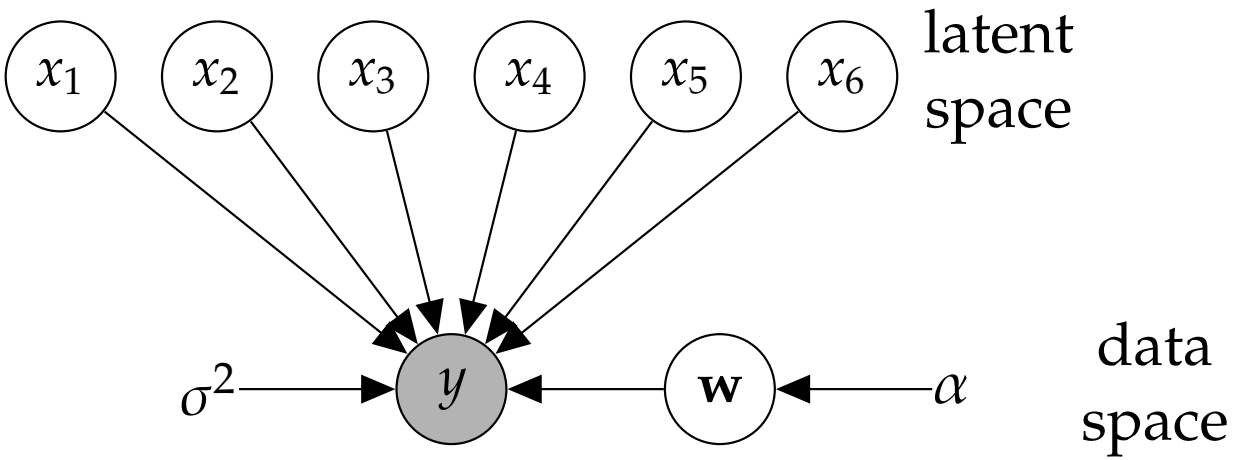
Graphical Representations of GP-LVM



Graphical Representations of GP-LVM



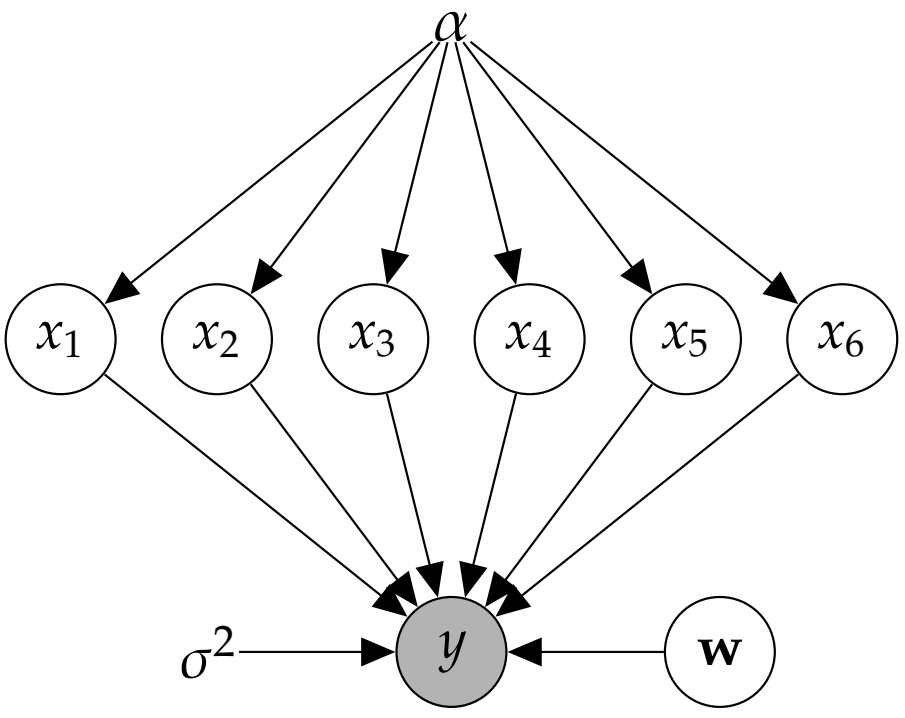
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha\mathbf{I}) \quad \mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$$

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

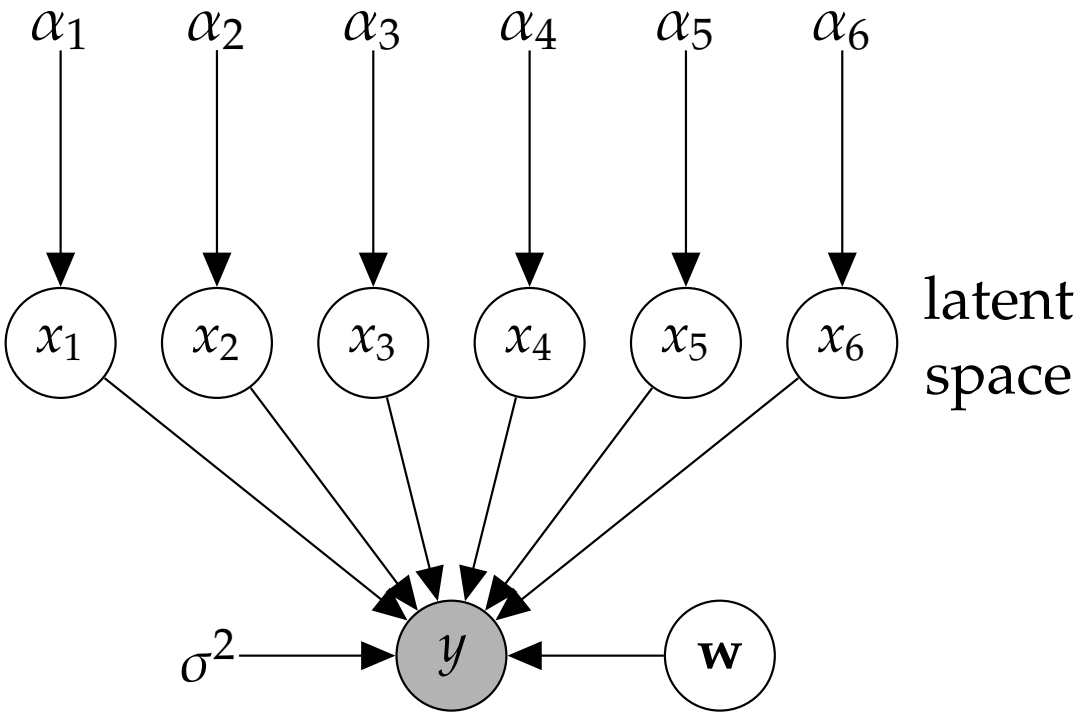
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad \mathbf{x} \sim \mathcal{N}(0, \alpha\mathbf{I})$$

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

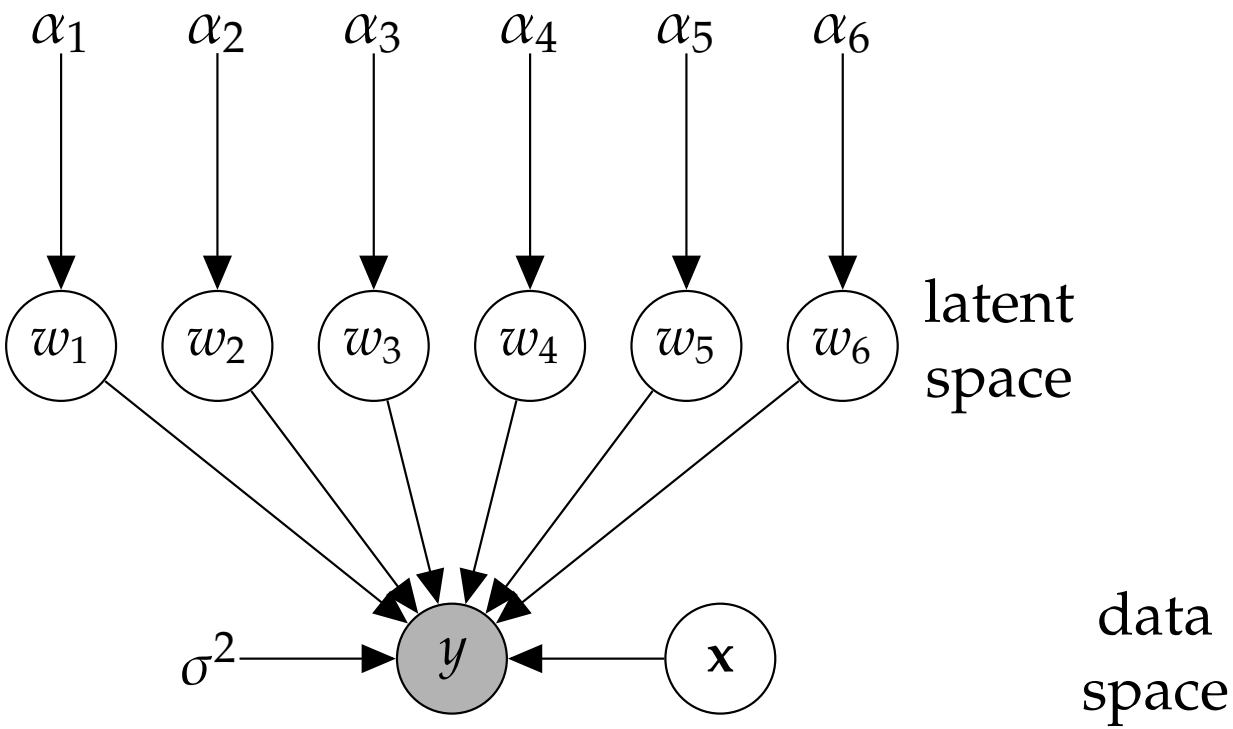
Graphical Representations of GP-LVM



$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad x_i \sim \mathcal{N}(0, \alpha_i)$$

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

Graphical Representations of GP-LVM



$$w_i \sim \mathcal{N}(0, \alpha_i) \quad \mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

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

Non-linear $f(\mathbf{x})$

- In linear case equivalence because $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$

$$p(w_i) \sim \mathcal{N}(\mathbf{0}, \alpha_i)$$

- In non linear case, need to scale columns of $\mathbf{X}$ in prior for $f(\mathbf{x})$.
- This implies scaling columns of $\mathbf{X}$ in covariance function

$$k(\mathbf{x}_{i,:}, \mathbf{x}_{j,:}) = \exp\left(-\frac{1}{2}(\mathbf{x}_{:,i} - \mathbf{x}_{:,j})^\top \mathbf{A}(\mathbf{x}_{:,i} - \mathbf{x}_{:,j})\right)$$

$\mathbf{A}$ is diagonal with elements α_i^2. Now keep prior spherical

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

- Covariance functions of this type are known as ARD (see e.g. Neal, 1996; MacKay, 2003; Rasmussen and Williams, 2006).

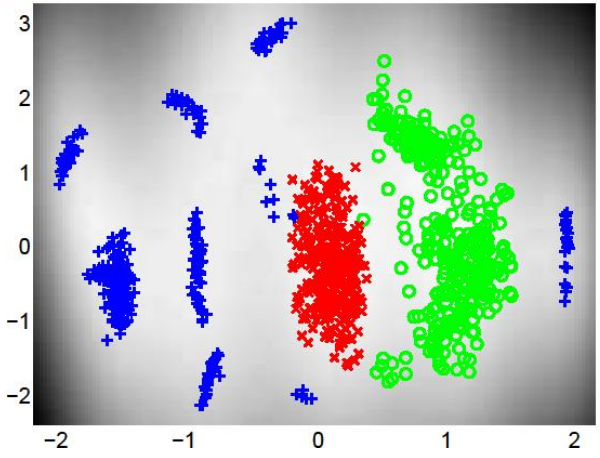
Automatic dimensionality detection

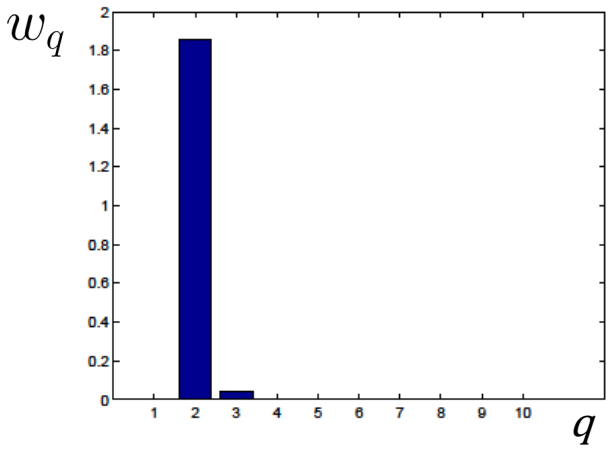
- Achieved by employing an *Automatic Relevance Determination (ARD)* covariance function for the prior on the GP mapping

- $f \sim GP(\mathbf{0}, k_f)$ with

$$k_f(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\frac{1}{2}\sum_{q=1}^{Q} w_q (x_{i,q} - x_{j,q})^2\right)$$

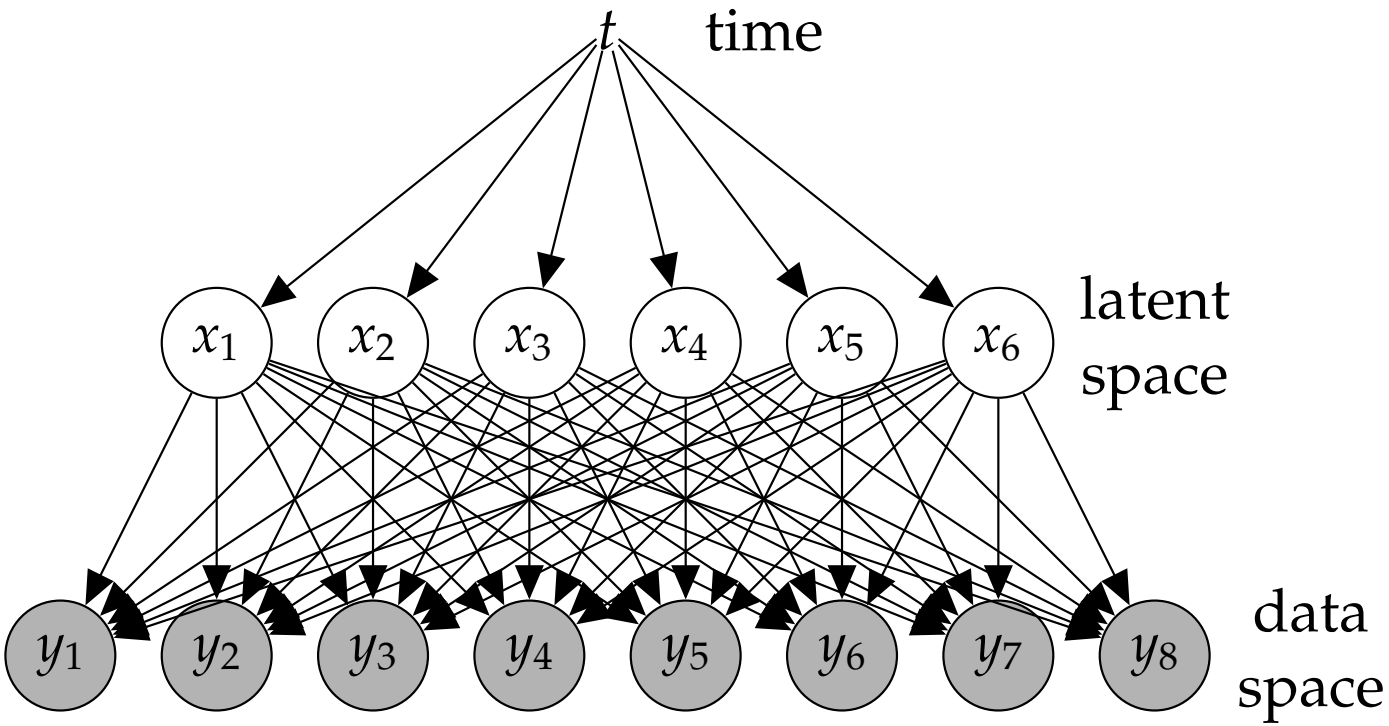
- Example





Gaussian Process Dynamical Systems

(Damianou et al., 2011)



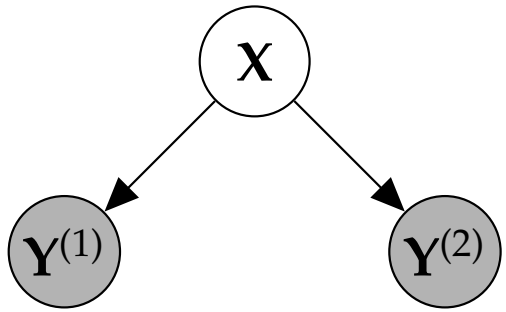
Gaussian Process over Latent Space

- Assume a GP prior for $p(\mathbf{X})$.
- Input to the process is time, $p(\mathbf{X}|t)$.

Interpolation of HD Video

Modeling Multiple 'Views'

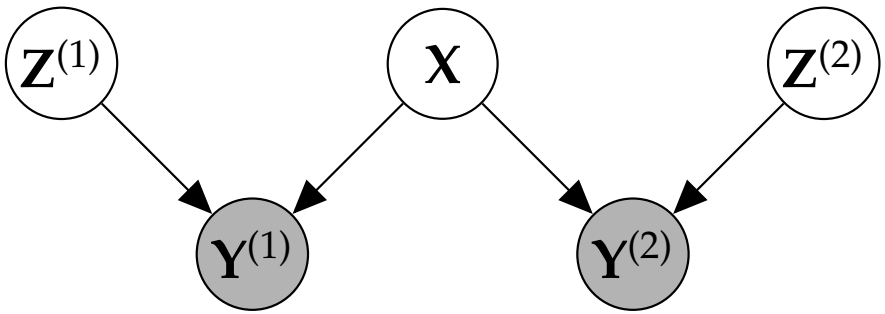
- Single space to model correlations between two different data sources, e.g., images & text, image & pose.
- Shared latent spaces: (Shon et al., 2006; Navaratnam et al., 2007; Ek et al., 2008b)



- Effective when the 'views' are correlated.
- But not all information is shared between both 'views'.
- PCA applied to concatenated data vs CCA applied to data.

Shared-Private Factorization

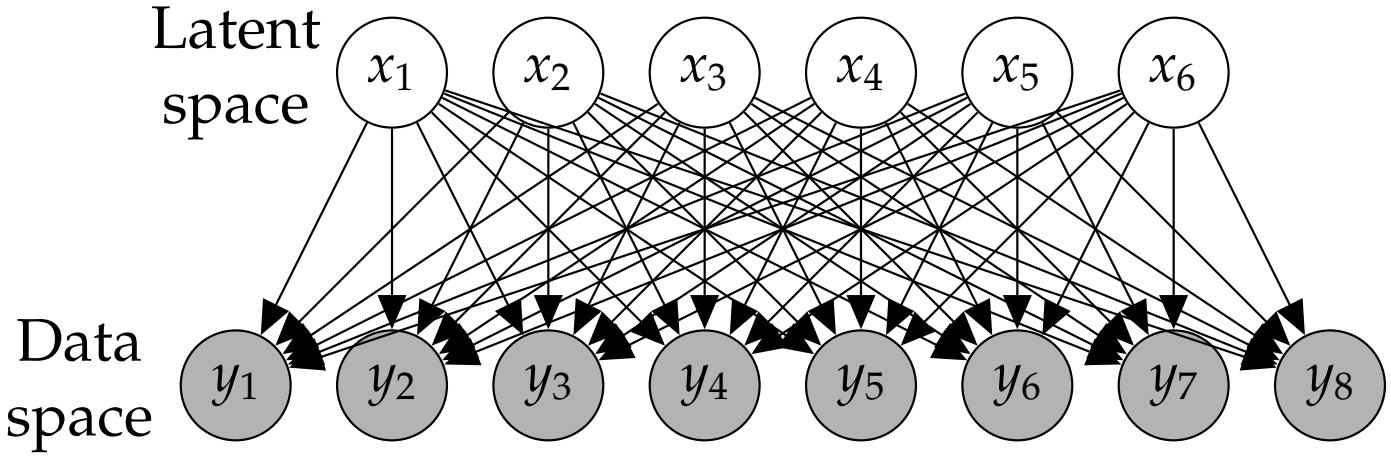
- In real scenarios, the 'views' are neither fully independent, nor fully correlated.
- Shared models
  - either allow information relevant to a single view to be mixed in the shared signal,
  - or are unable to model such private information.
- Solution: Model shared and private information (Virtanen et al., 2011; Ek et al., 2008a; Leen and Fyfe, 2006; Klami and Kaski, 2007, 2008; Tucker, 1958)



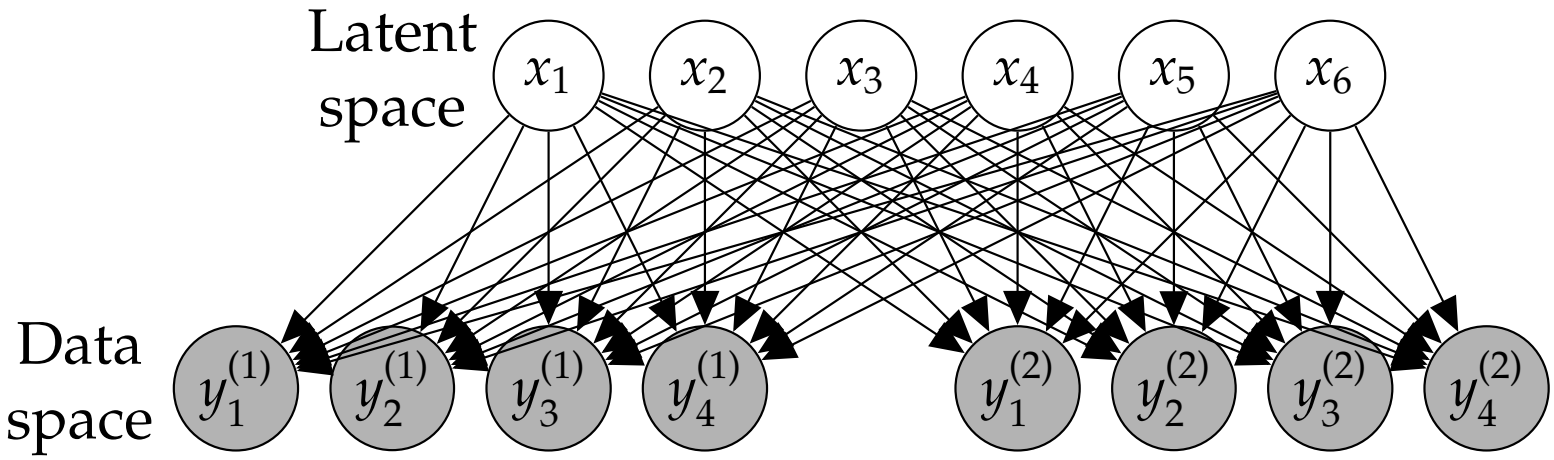
- Probabilistic CCA is case when dimensionality of $\mathbf{Z}$ matches $\mathbf{Y}^{(i)}$ (cf Inter Battery Factor Analysis (Tucker, 1958)).

Manifold Relevance Determination

Damianou et al. (2012)



Shared GP-LVM



Separate ARD parameters for mappings to $\mathbf{Y}^{(1)}$ and $\mathbf{Y}^{(2)}$.

Example: Yale faces

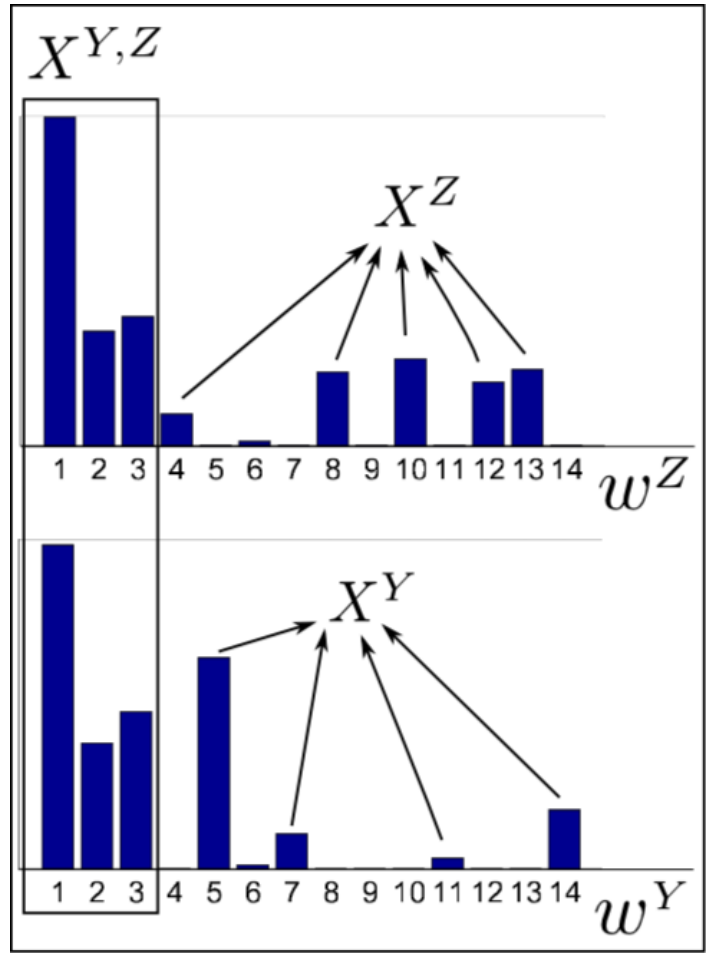
- Dataset Y: 3 persons under all illumination conditions
- Dataset Z: As above for 3 different persons
- Align datapoints $\mathbf{x}_n$ and $\mathbf{z}_n$ only based on the lighting direction

Results

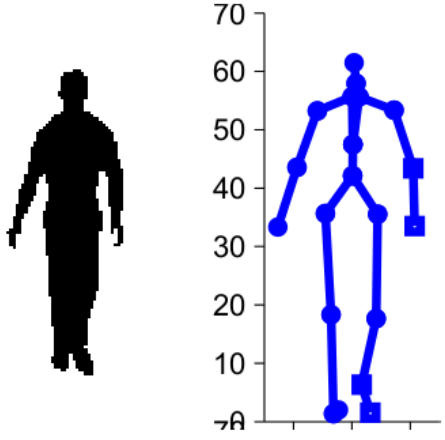
- Latent space X initialised with 14 dimensions

- Weights define a segmentation of X

- Video / demo…

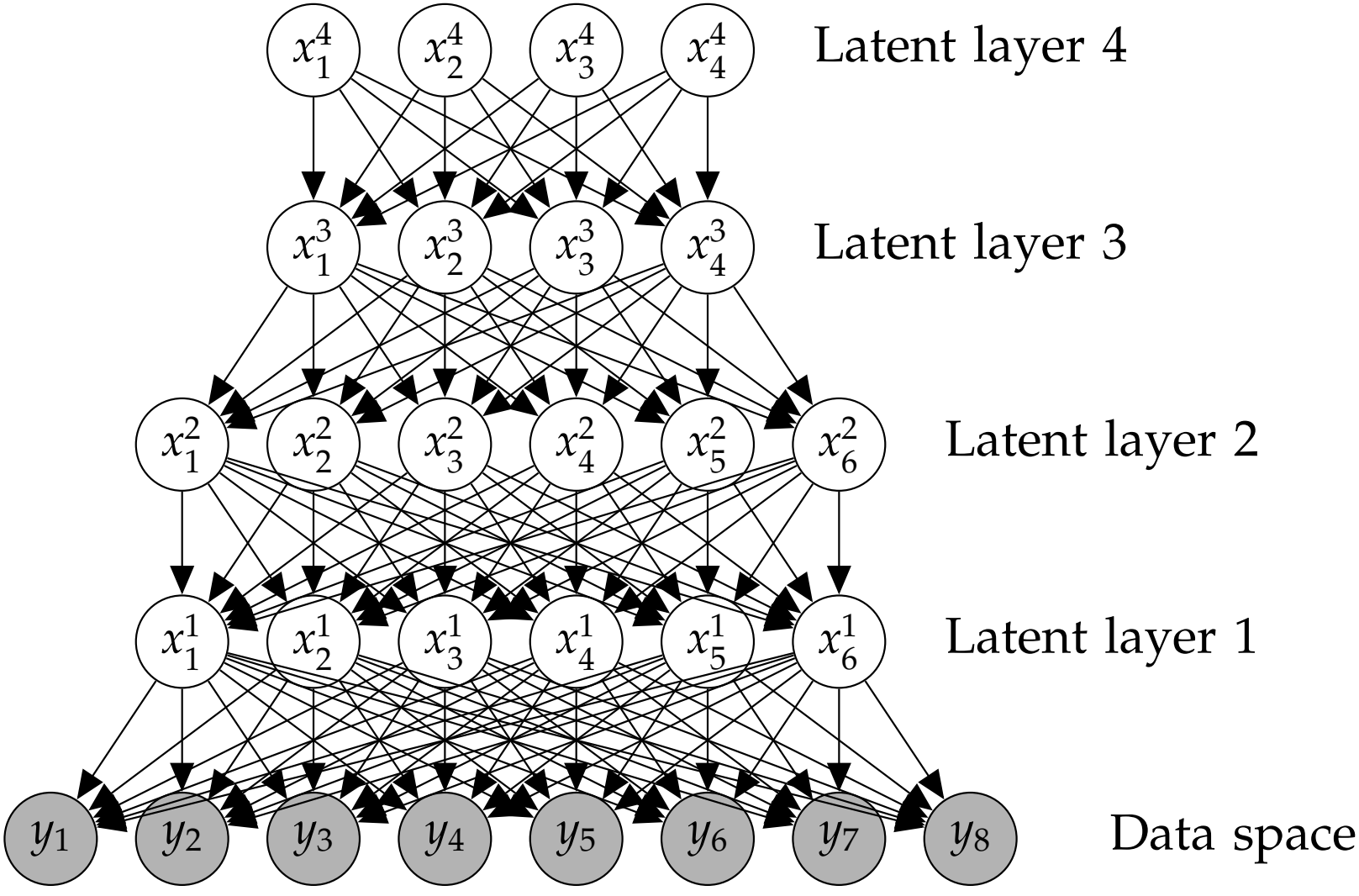


Potential applications..?

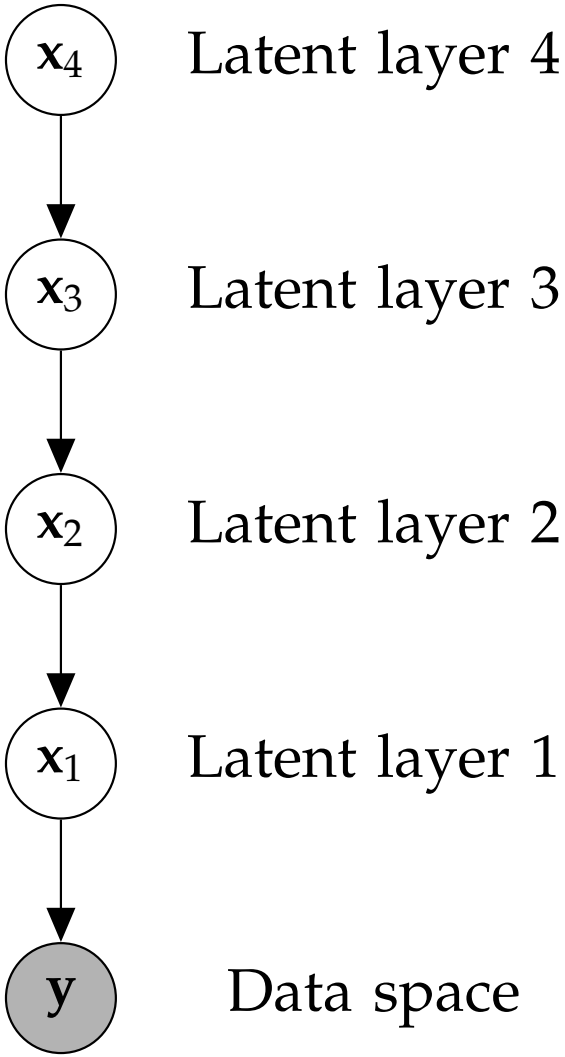


Manifold Relevance Determination

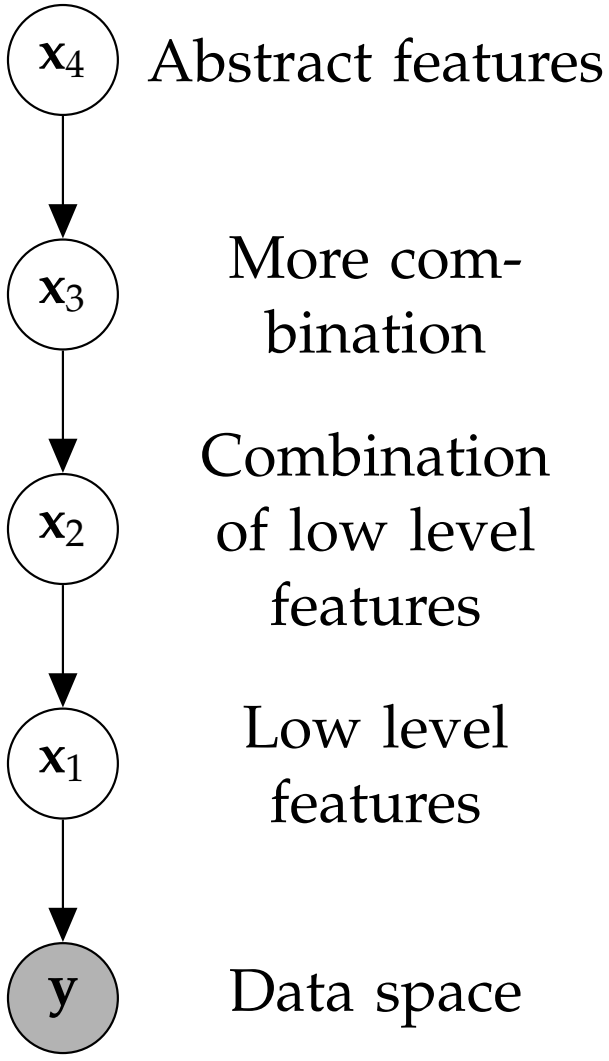
Deep Models


x_1^4 x_2^4 x_3^4 x_4^4
Latent layer 4
x_1^3 x_2^3 x_3^3 x_4^3
Latent layer 3
x_1^2 x_2^2 x_3^2 x_4^2 x_5^2 x_6^2
Latent layer 2
x_1^1 x_2^1 x_3^1 x_4^1 x_5^1 x_6^1
Latent layer 1
y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8
Data space

Deep Models


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Deep Models

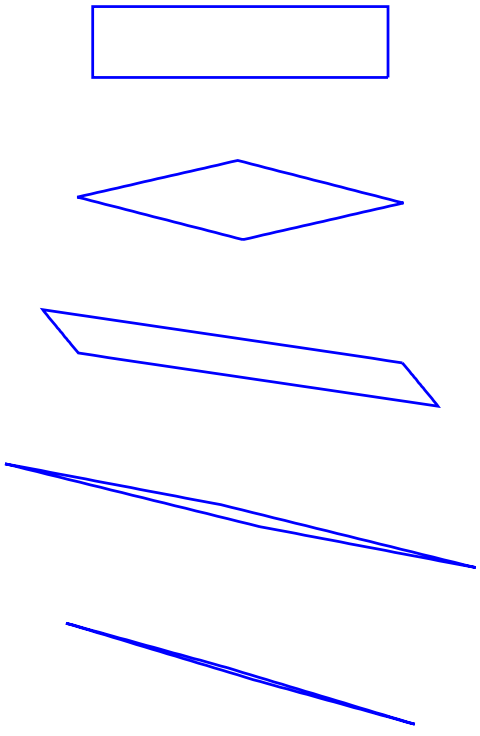

$\mathbf{x}_4$
Abstract features
$\mathbf{x}_3$
More combination
$\mathbf{x}_2$
Combination of low level features
$\mathbf{x}_1$
Low level features
$\mathbf{y}$
Data space

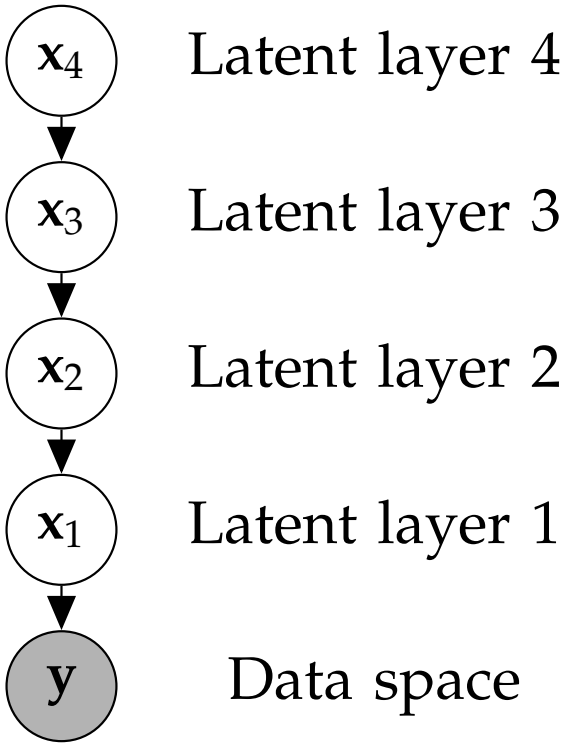
Deep Gaussian Processes

Damianou and Lawrence (2013)

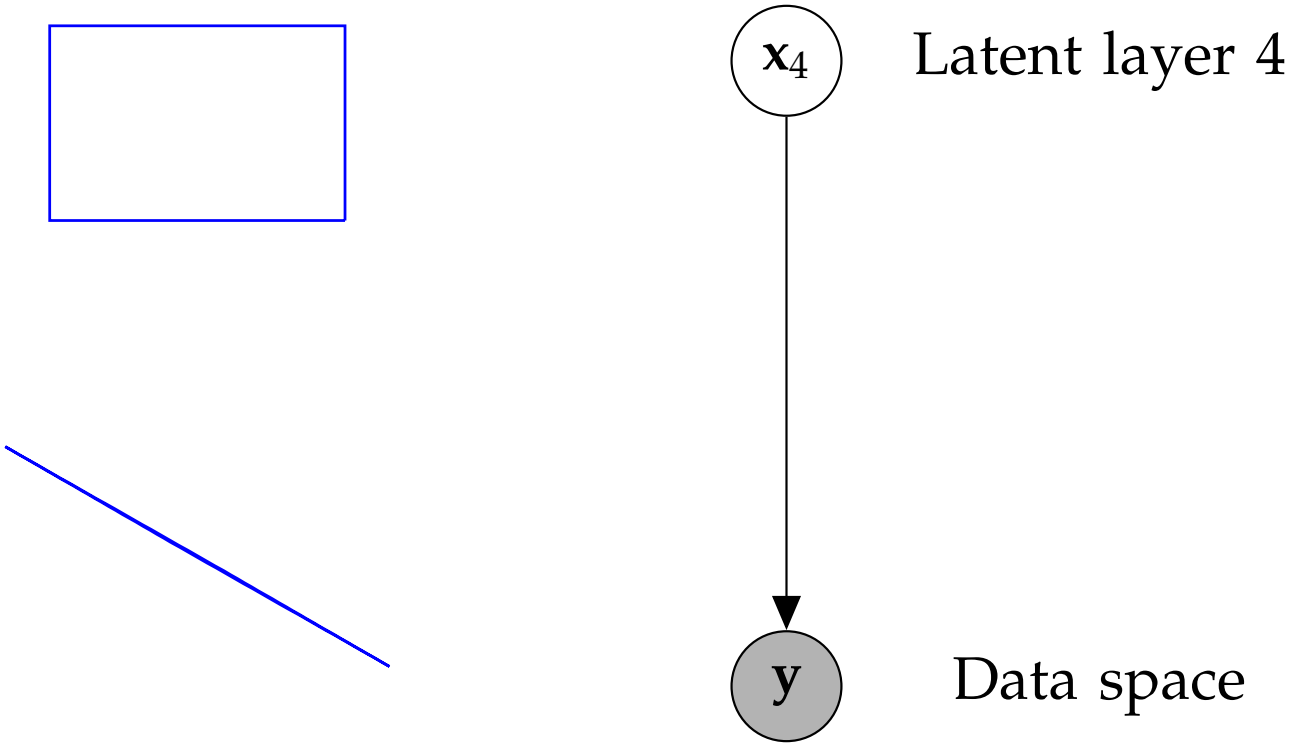
- Deep architectures allow abstraction of features (Bengio, 2009; Hinton and Osindero, 2006; Salakhutdinov and Murray, 2008).
- We use variational approach to stack GP models.

Stacked PCA

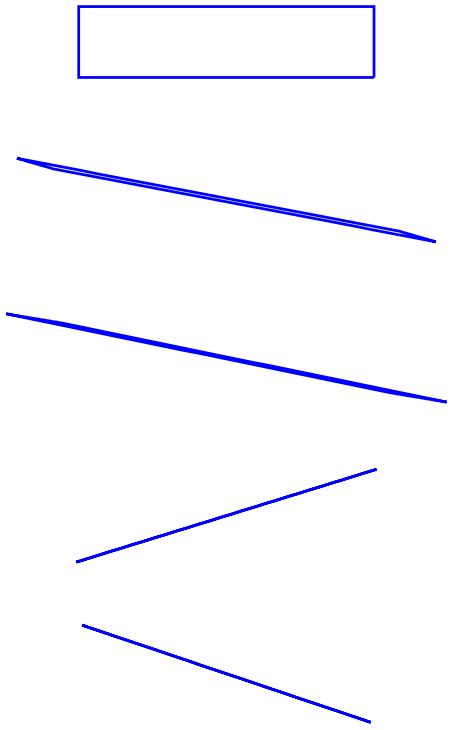


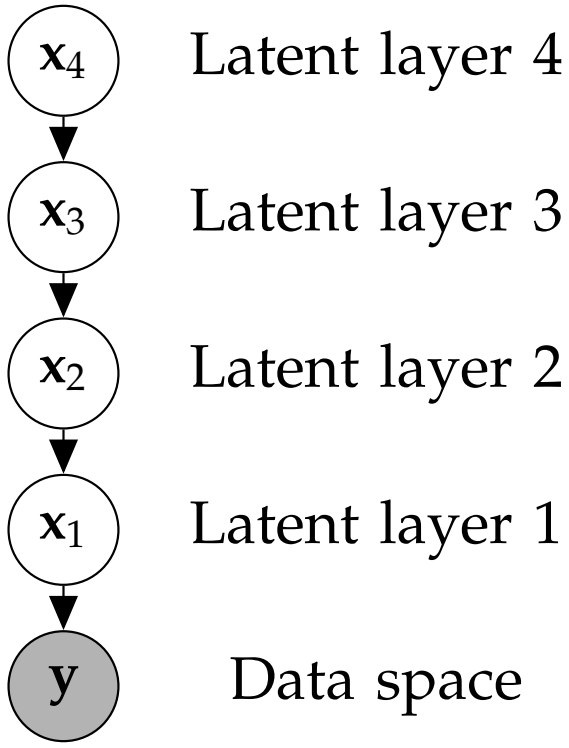

$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked PCA

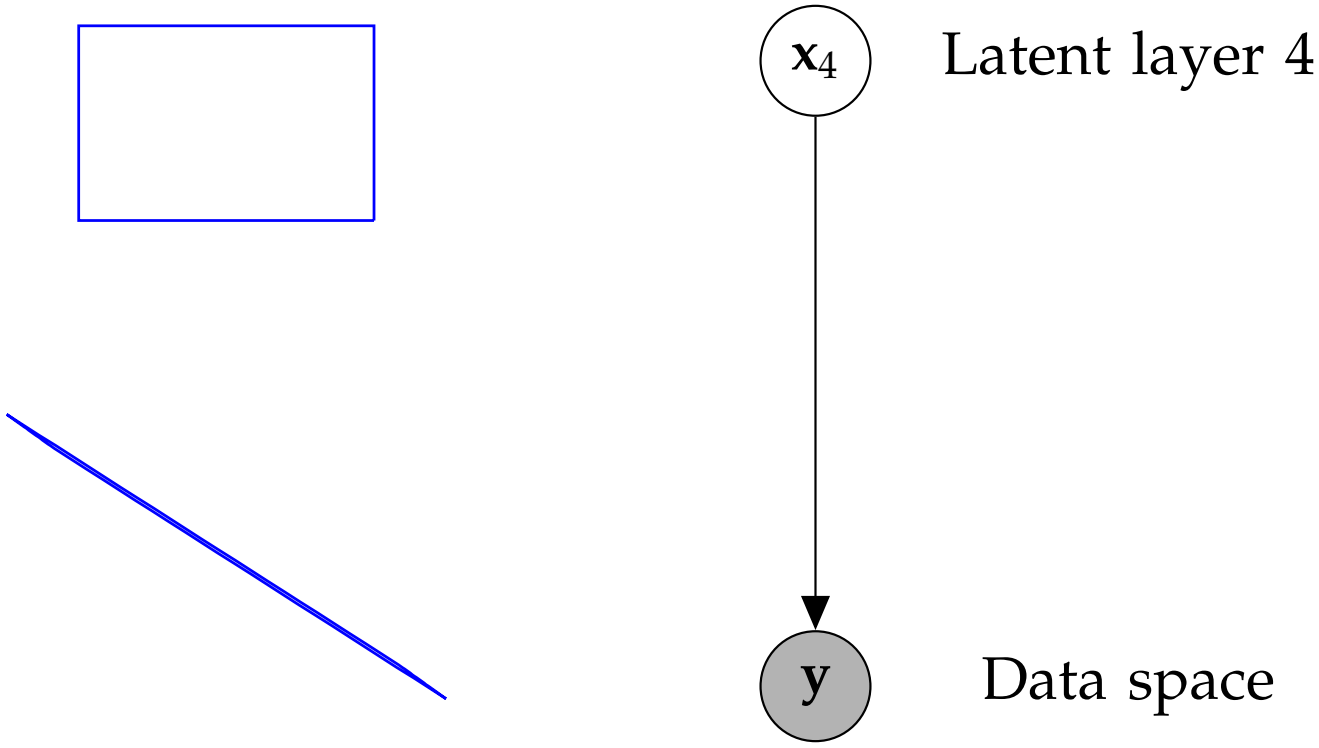


Stacked PCA

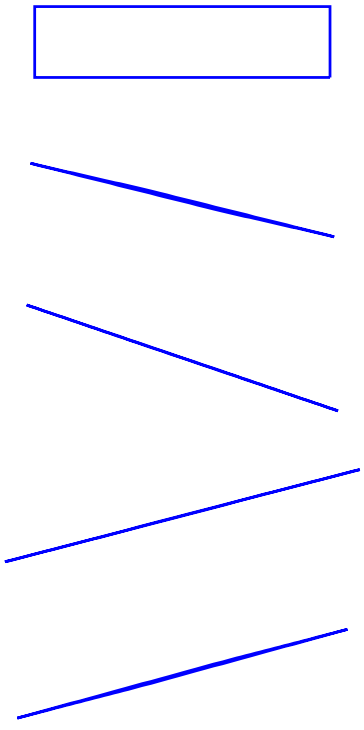


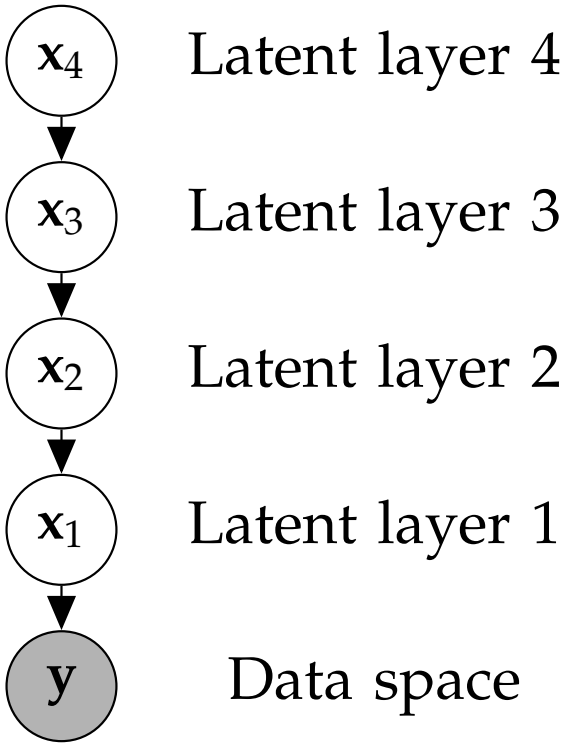

$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked PCA

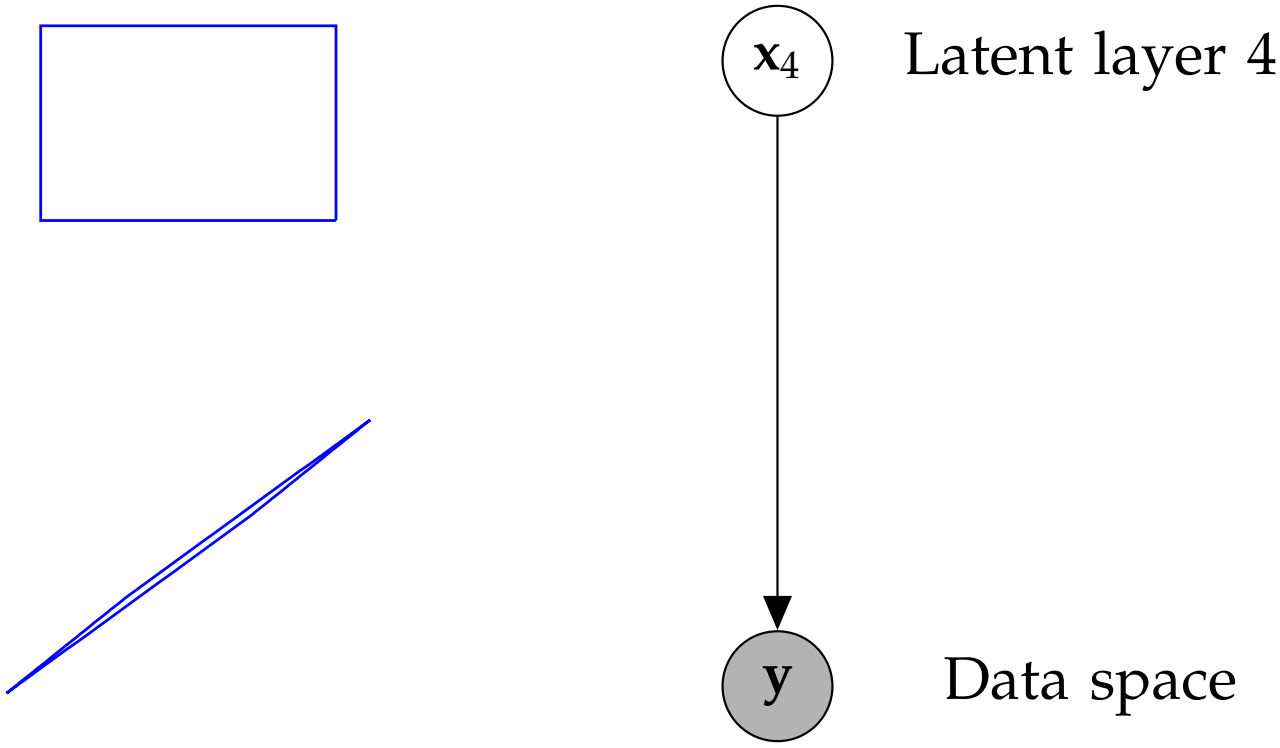


Stacked PCA

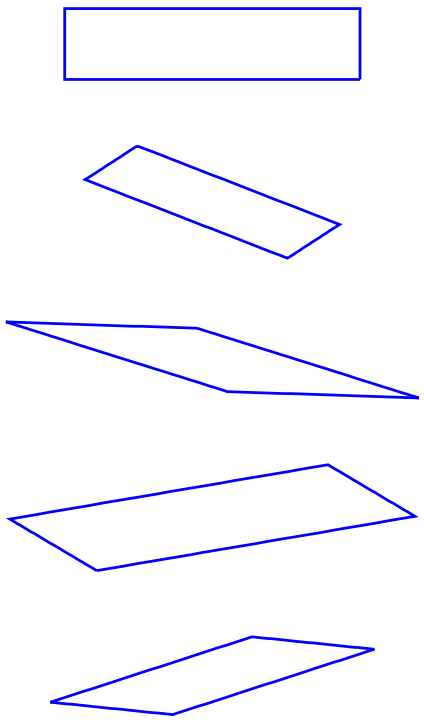


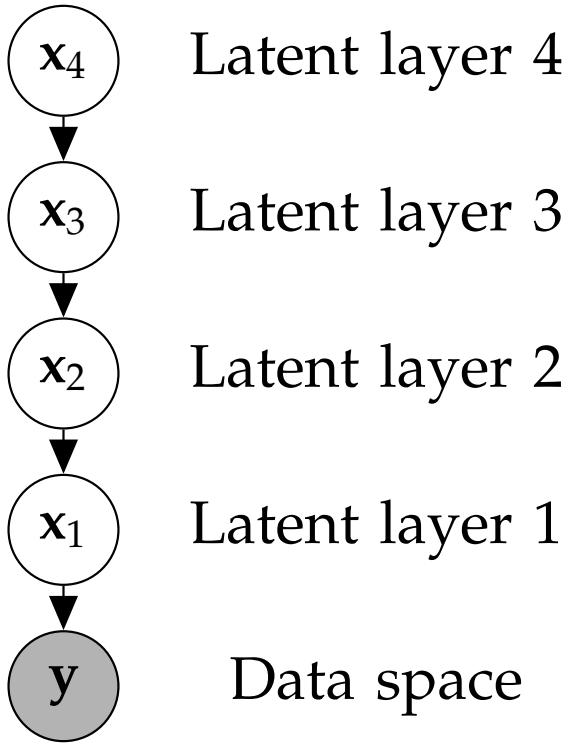

$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked PCA

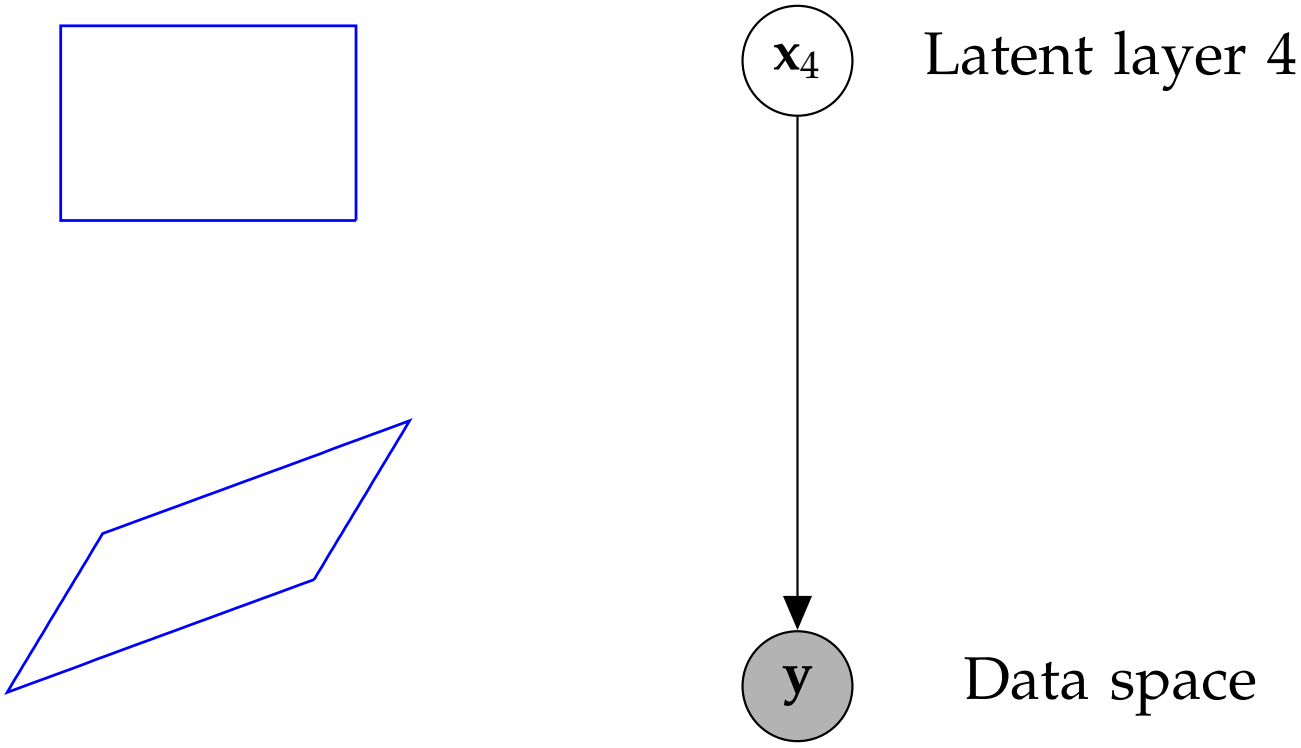


Stacked PCA

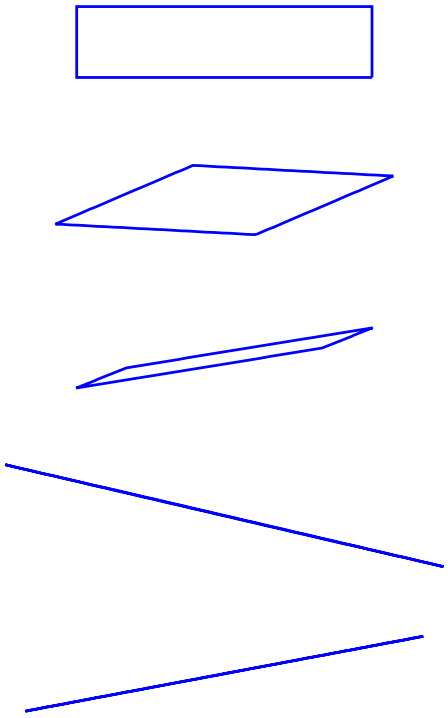


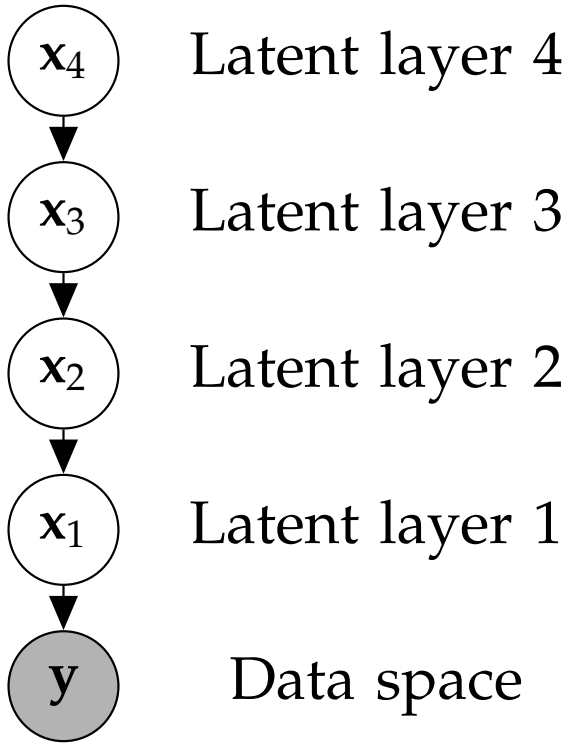

$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked PCA

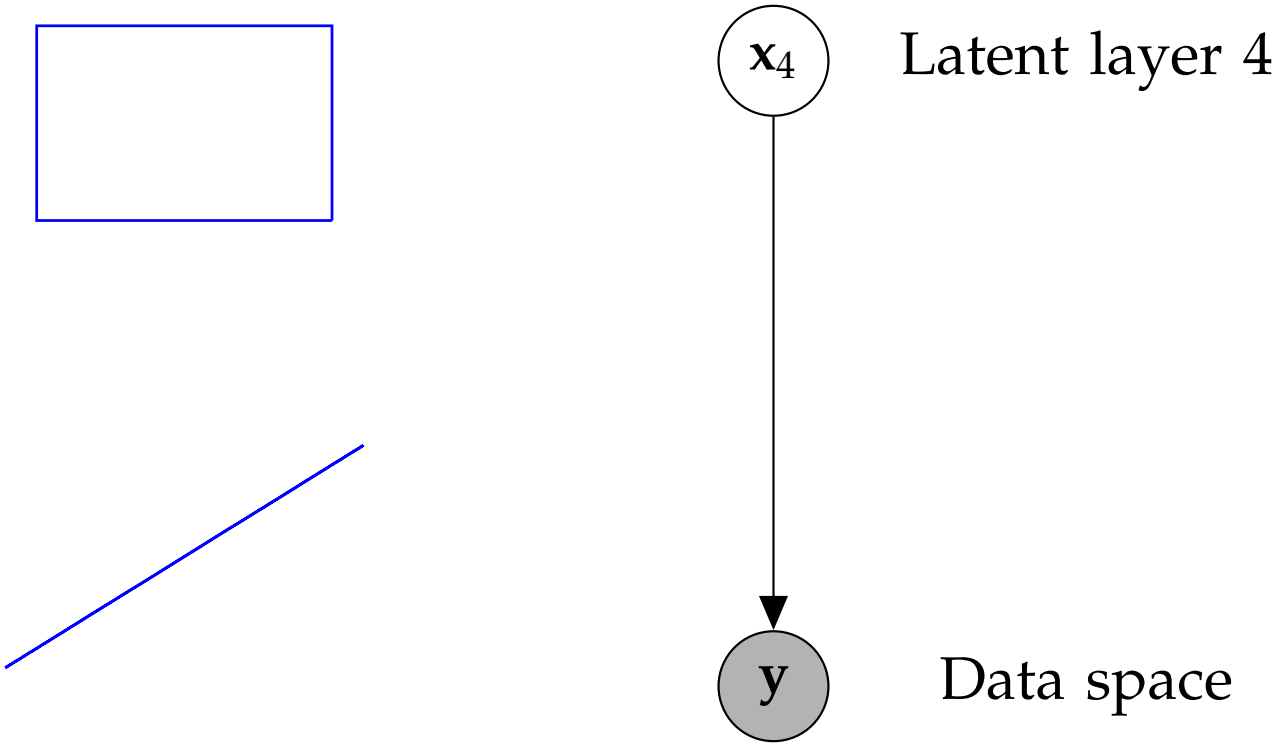


Stacked PCA

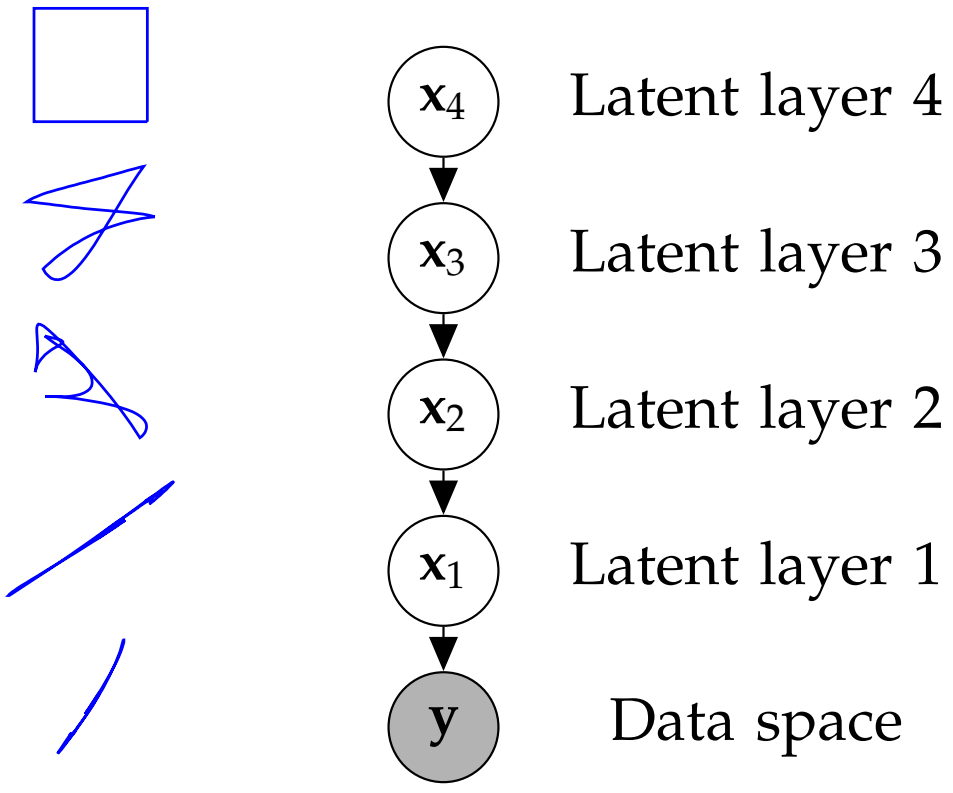



$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

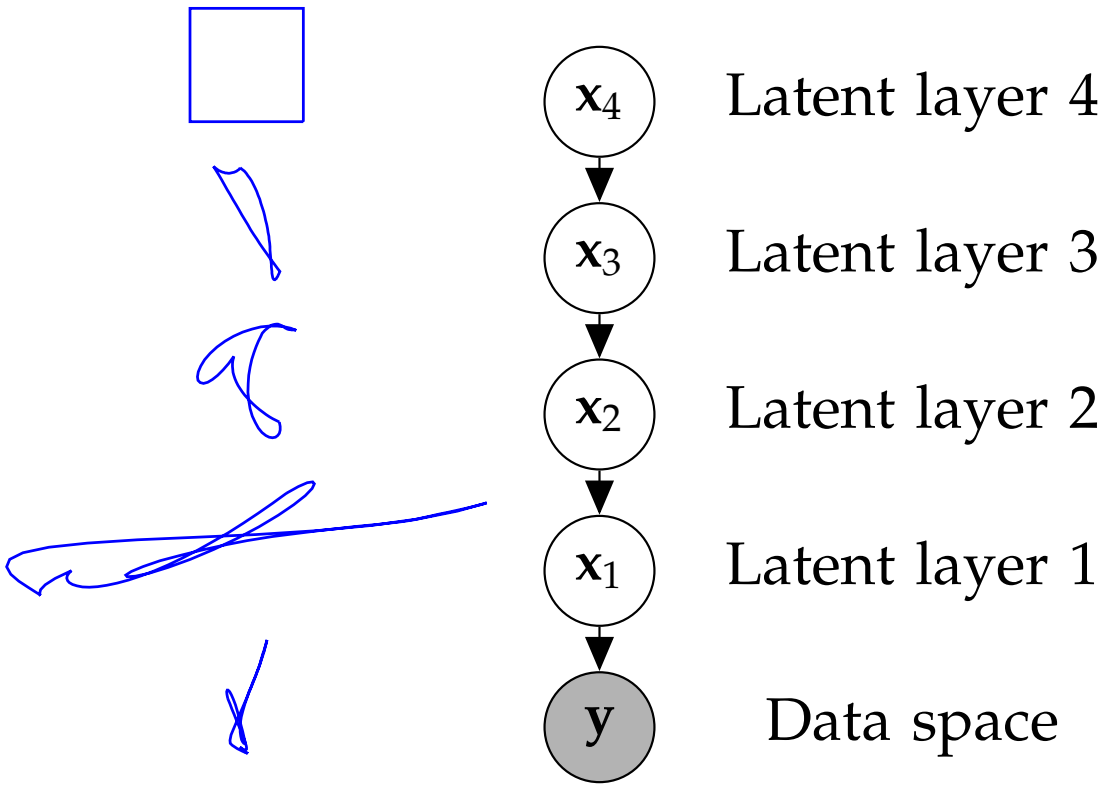
Stacked PCA



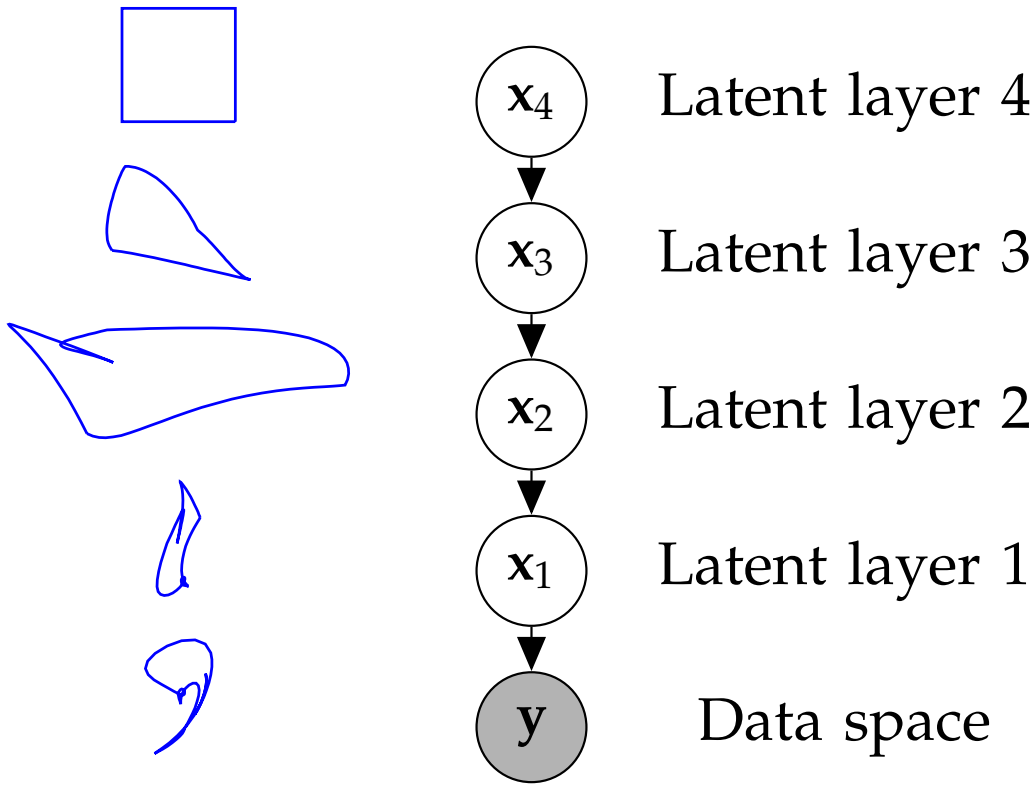
Stacked GPs


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

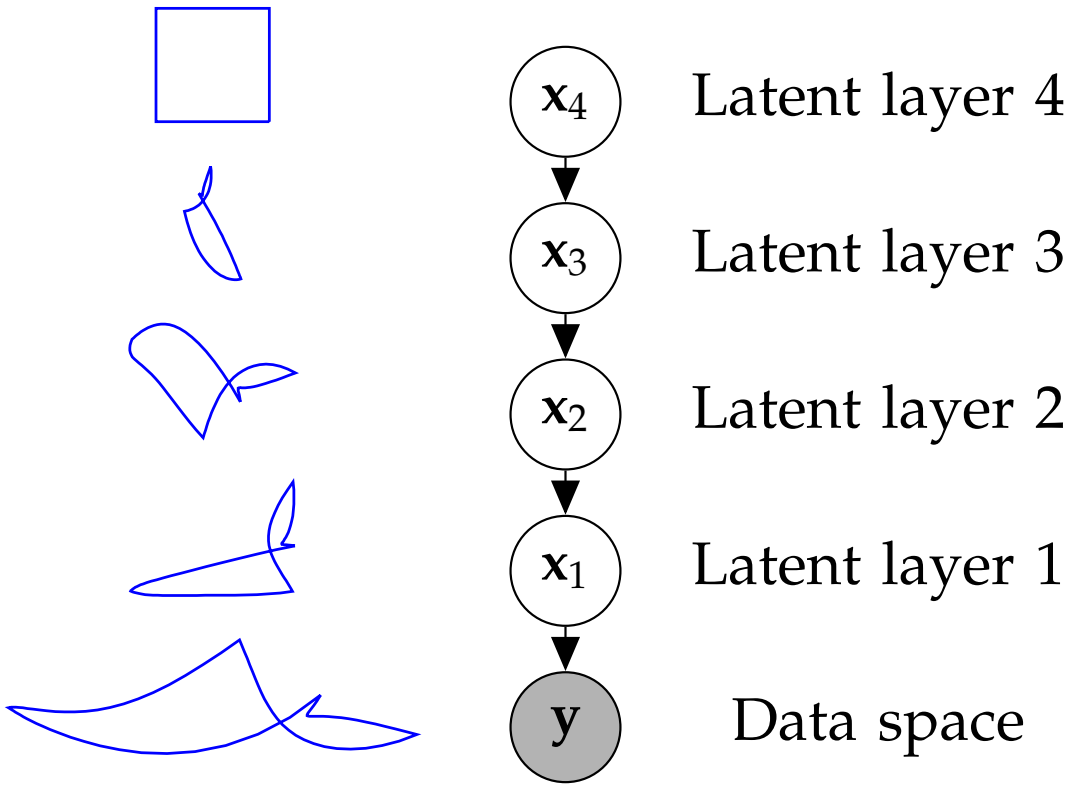
Stacked GPs


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

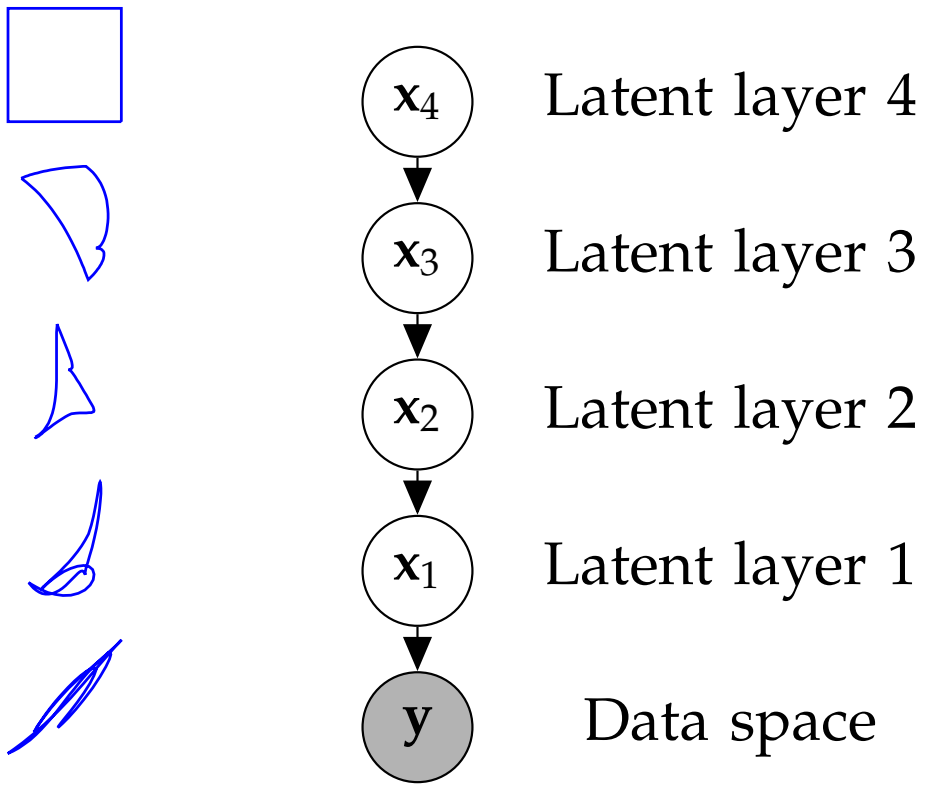
Stacked GPs


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked GPs


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked GPs


$\mathbf{x}_4$
Latent layer 4
$\mathbf{x}_3$
Latent layer 3
$\mathbf{x}_2$
Latent layer 2
$\mathbf{x}_1$
Latent layer 1
$\mathbf{y}$
Data space

Stacked GPs (video by David Duvenaud)

Duvenaud et al. (2014)

Avoiding Pathologies in Very Deep Networks

David Duvenaud
University of Cambridge
dkd23@cam.ac.uk

Oren Rippel
M.I.T., Harvard University
rippel@math.mit.edu

Ryan Adams
Harvard University
rpa@seas.harvard.edu

Zoubin Ghahramani
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

Choosing appropriate architectures and initialization strategies is crucial to good performance of deep networks. To shed light on this problem, we analyze the analogous problem of constructing useful priors on compositions of functions. Specifically, we study deep Gaussian processes, a type of infinitely-wide, deep neural network. We show that in these architectures, the representational capacity of the network tends to capture fewer degrees of freedom as the number of layers increases, retaining only a single degree of freedom in the limit. We propose alternate network architectures which do not suffer from these pathologies. We also derive novel covariance functions obtained by composing infinitely many feature transforms.

1 INTRODUCTION

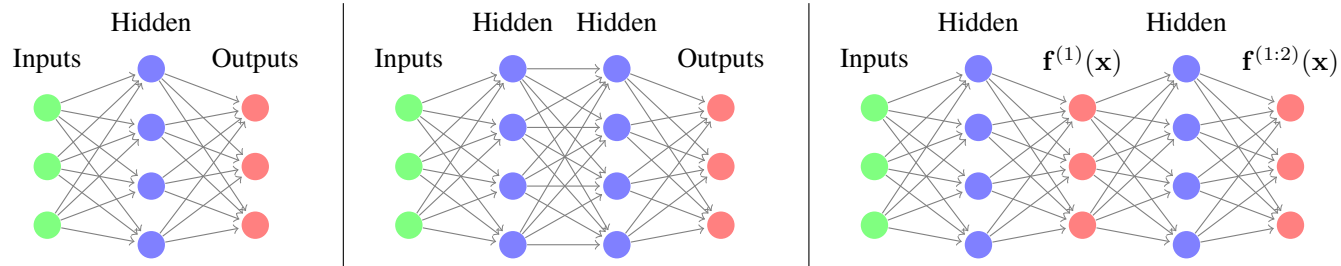
Much recent work on deep networks has focused on weight initialization (Martens, 2010), regularization (Lee *et al.*,

Hence, understanding properties of such function compositions helps us gain insight into deep networks. In this paper, we examine a simple and flexible class of priors on compositions of functions, namely deep Gaussian processes. (Damianou and Lawrence, 2013). Deep GPs are simply priors on compositions of vector-valued functions, where each output of each layer is distributed independently according to a GP prior:

$$\mathbf{f}^{(1:L)}(\mathbf{x}) = \mathbf{f}^{(L)}(\mathbf{f}^{(L-1)}(\dots \mathbf{f}^{(2)}(\mathbf{f}^{(1)}(\mathbf{x}))\dots)) \qquad (1)$$

where each $\mathbf{f}_d^{(\ell)} \overset{\text{ind}}{\sim} \mathcal{GP}\big(0, k_d^{\ell}(\mathbf{x}, \mathbf{x}')\big)$. Although inference in these models is non-trivial, they can be derived as a special case of multi-layer perceptrons (MLPs), and as such make canonical candidates for generative models of functions that closely relate to neural networks.

By characterizing these models, this paper shows that representations based on repeated composition of independently-initialized functions exhibit a pathology where the representation becomes invariant to all but one direction of variation. This corresponds to an eventual debilitating decrease in the information capacity of networks as a function of their number of layers. However, we will demonstrate that a simple change in architecture — namely, connecting the input to each layer — alleviates this prob-



a) One-hidden-layer MLP b) Two-hidden layer MLP c) Two-layer GP: $\mathbf{y} = \mathbf{f}^{(2)}(\mathbf{f}^{(1)}(\mathbf{x}))$

Figure 1: Comparing architectures. In the deep GP models, there are two possible meanings for the hidden units. We can consider every other layer to be a linear combination of an infinite number of parametric hidden units. Alternatively, we can integrate out the hidden layers, and consider the deep GP to be a neural network with a finite number of hidden units, each with a different non-parametric activation function.

where $\mathbf{V}^{(1)}$ is another weight matrix.

There exists a correspondence between one-layer MLPs and GPs (Neal, 1995). GP priors can be viewed as a prior on neural networks with infinitely many hidden units. More precisely, for any model of the form

$$f(\mathbf{x}) = \frac{1}{K}\alpha^{\mathsf{T}}\Phi(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K}\alpha_i\phi_i(\mathbf{x}), \tag{4}$$

with fixed features $[\phi_1(\mathbf{x}), \ldots, \phi_K(\mathbf{x})]^{\mathsf{T}} = \Phi(\mathbf{x})$ and i.i.d. α's with zero mean and finite variance σ^2, the central limit theorem implies that as the number of features $K \to \infty$, any two function values $f(\mathbf{x}), f(\mathbf{x}')$ have a joint distribution approaching $\mathcal{N}\left(0, \frac{\sigma^2}{K}\sum_{i=1}^{K}\phi_i(\mathbf{x})\phi_i(\mathbf{x}')\right)$. A joint Gaussian distribution between any two function values is the definition of a Gaussian process.

The result is surprisingly general: It doesn't put any constraints on what the features are (other than having bounded activation), nor does it require that the feature weights α be Gaussian distributed.

We can also work backwards to derive a one-layer MLP from a GP: Mercer's theorem implies that any positive-

However, in a deep GP, the D outputs $\mathbf{f}^{(n)}(\mathbf{x})$ in between each layer are weighted sums of the hidden units of the layer below, and the next layer's hidden units depend only on these D outputs. Thus deep GPs have an extra set of layers that a MLP doesn't have, shown in figure 1c.

There are two ways to directly relate deep GPs to MLPs. First, we can note that, if the hidden units in a deep GP implied by Mercer's theorem $\phi^{(n)}(\mathbf{x})$ depend only on a linear projection of their inputs, as in the sigmoidal activation function $\phi(\mathbf{x}) = \sigma\left(\mathbf{b}^{(n)} + \mathbf{W}^{(n)}f^{(n-1)}(\mathbf{x})\right)$, then we can simply substitute $\mathbf{f}^{(n-1)}(\mathbf{x}) = \mathbf{V}^{(n-1)}\phi^{(n-1)}(\mathbf{x})$ to recover $\phi(\mathbf{x}) = \sigma\left(\mathbf{b}^{(n)} + \mathbf{W}^{(n)}\mathbf{V}^{(n-1)}\phi^{(n-1)}(\mathbf{x})\right)$. Thus, we can ignore the intermediate outputs $\mathbf{f}^{(n)}(\mathbf{x})$, and exactly recover an MLP with activation functions given by Mercer's theorem, but with rank-D weight matrices between layers!

The second, more general way we can relate the two model classes is to integrate out all $\mathbf{V}^{(n)}$, and view deep GP models as a neural network with a finite number of nonparametric, GP-distributed basis functions, where the D outputs of $\mathbf{f}^{1:\ell}(\mathbf{x})$ represent the output of the hidden nodes at the ℓ^{th} layer. This second view lets us compare deep GP models

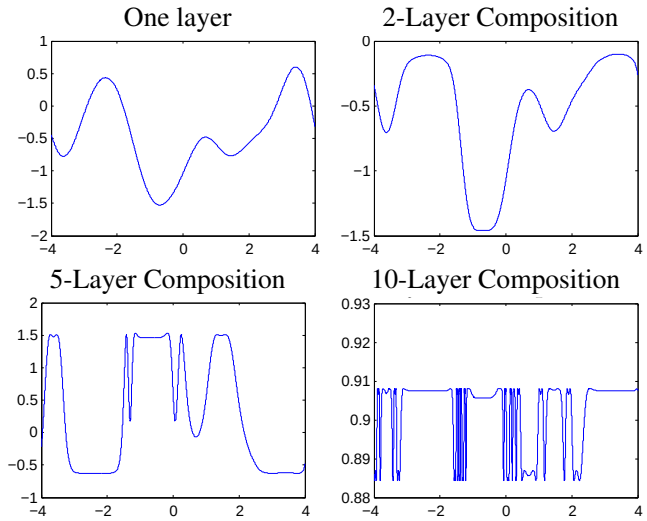


Figure 2: One-dimensional draws from a deep GP prior. After a few layers, the functions begin to be either nearly flat, or highly varying, everywhere. This is a consequence of the distribution on derivatives becoming heavy-tailed.

the smoothness. The derivative of a GP with a squared-exp kernel is distributed as $\mathcal{N}\left(0, \sigma_f^2/\ell^2\right)$. Intuitively, a GP is likely to have large derivatives if it has high variance and small lengthscales.

By the chain rule, the derivative of a one-dimensional deep GP is simply a product of its (independent) derivatives. The distribution of the absolute value of this derivative is a product of half-normals, each with mean $\sqrt{2\sigma_f^2/\pi\ell^2}$.

Thus, if we choose kernel parameters such that $\sigma_f^2/\ell_{d_1}^2 = \pi/2$, then $\mathbb{E}\left[\left|\partial f(x)/\partial x\right|\right] = 1$, and so $\mathbb{E}\left[\left|\partial f^{1:L}(x)/\partial x\right|\right] = 1$, that is to say, the expected magnitude of the derivative remains constants no matter the depth. If σ_f^2/ℓ^2 is less than $\pi/2$, the expected derivative magnitude goes to zero, and if greater, the expected magnitude goes to infinity as a function-normal as L grows:

$$\log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| = \sum_{i=1}^{L} \log\left|\frac{\partial f^{i}(x)}{\partial x}\right|$$

$$\implies \log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| \overset{L\to\infty}{\sim} \mathcal{N}\left(Lm_{\log}, L^2 v_{\log}\right) \quad (9)$$

Even if the expected magnitude of the derivative remains constant, the variance of the log-normal distribution grows without bound as the depth increases. Because the log-normal distribution is heavy-tailed, and its domain is bonded below by zero, the derivative will become very small almost everywhere, with rare but very large jumps.

Figure 2 shows this behavior in a draw from a 1D deep GP prior, at varying depths. This figure also shows that once the derivative in one region of the input space becomes very large or very small, it is likely to remain that way in subsequent layers.

4 THE JACOBIAN OF A DEEP GP IS A PRODUCT OF INDEPENDENT NORMAL MATRICES

We now derive the distribution on Jacobians of multivariate functions drawn from a deep GP prior.

Lemma 4.1. *The partial derivatives of a function mapping $\mathbb{R}^D \to \mathbb{R}$ drawn from a GP prior with a product kernel are independently Gaussian distributed.*

Proof. Because differentiation is a linear operator, the derivatives of a function drawn from a GP prior are also jointly Gaussian distributed. The covariance between partial derivatives w.r.t. input dimensions d_1 and d_2 of vector $\mathbf{x}$ are given by Solak *et al.* (2003):

$$\operatorname{cov}\left(\frac{\partial f(\mathbf{x})}{\partial x_{d_1}}, \frac{\partial f(\mathbf{x})}{\partial x_{d_2}}\right) = \left.\frac{\partial^2 k(\mathbf{x}, \mathbf{x}')}{\partial x_{d_1} \partial x'_{d_2}}\right|_{\mathbf{x}=\mathbf{x}'} \quad (10)$$

Proof. The Jacobian of the vector-valued function $\mathbf{f}(\mathbf{x})$ is a matrix J with elements $J_{ij} = \frac{\partial \mathbf{f}_i(\mathbf{x})}{\partial x_j}$. Because we've assumed that the GPs on each output dimension $f_d(\mathbf{x}) \sim \mathcal{GP}$ are independent, it follows that each row of J is independent. Lemma 4.1 shows that the elements of each row are independent Gaussian. Thus all entries in the Jacobian of a GP-distributed transform are independent Gaussian R.V.s. □

Theorem 4.3. *The Jacobian of a deep GP with a product kernel is a product of independent Gaussian matrices, with each entry in each matrix being drawn independently.*

Proof. When composing L different functions, we'll denote the *immediate* Jacobian of the function mapping from layer $\ell-1$ to layer ℓ as $J^\ell(\mathbf{x})$, and the Jacobian of the entire composition of L functions by $J^{1:L}(\mathbf{x})$.

By the multivariate chain rule, the Jacobian of a composition of functions is simply the product of the Jacobian matrices of each function. Thus the Jacobian of the composed (deep) function $\mathbf{f}^{(L)}(\mathbf{f}^{(L-1)}(\dots \mathbf{f}^{(3)}(\mathbf{f}^{(2)}(\mathbf{f}^{(1)}(\mathbf{x}))) \dots))$ is

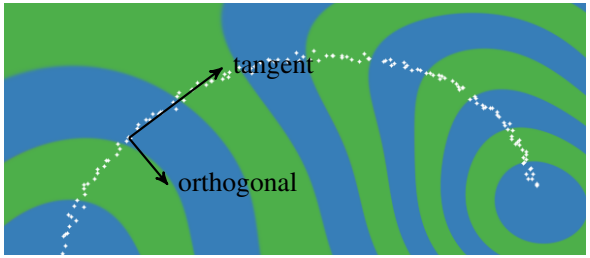
$$J^{1:L}(x) = J^L J^{L-1} \dots J^3 J^2 J^1. \tag{12}$$

By Lemma 4.2, each $J^\ell_{i,j} \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \frac{\sigma_f^2}{\ell^2}\right)$, so the complete Jacobian is a product of independent Gaussian matrices, and each entry of those matrices is drawn independently. □

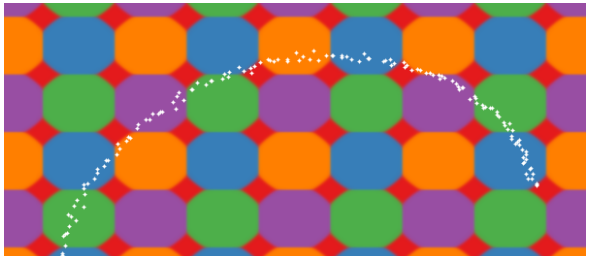
Theorem 4.3 allows us to analyze the representational properties of a deep Gaussian process by simply examining the properties of products of independent Gaussian matrices, a well-studied object.

5 FORMALIZING A PATHOLOGY

Rifai *et al.* (2011b) argue that a good latent representation is invariant in directions orthogonal to the manifold on which the data lie. Conversely, a good latent represen-



a) A noise-tolerant representation (blue & green) of a one-dimensional manifold (white)



b) A naïve representation (colors) of a one-dimensional manifold (white)

Figure 3: Comparing representations of data on a 1-D manifold. A representation is a function mapping the input space to some set of outputs. Here, colors show the output of the computed representation. Representation a) is invariant in directions orthogonal to the data manifold, making it robust to noise in those directions, and reducing the number of parameters needed to represent a datapoint. It also changes in directions tangent to the manifold, preserving information for later layers. Representation b) changes in all directions, preserving potentially useless information.

implying that such models are insuitable to model manifolds of greater than one dimension.

To visualize this pathology in another way, figure 6 illustrates the value that at each point in the input space is mapped to after successive warpings. After 40 warpings, we can see that locally, there is usually only one direction that one can move in x-space in order to change the value

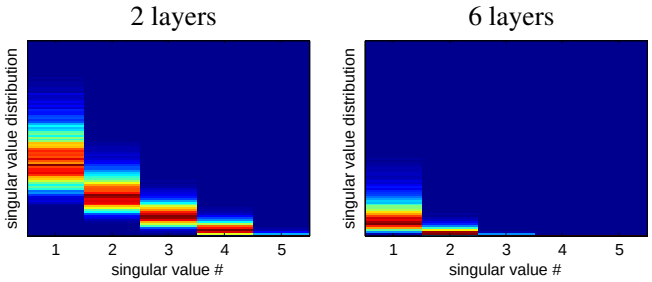
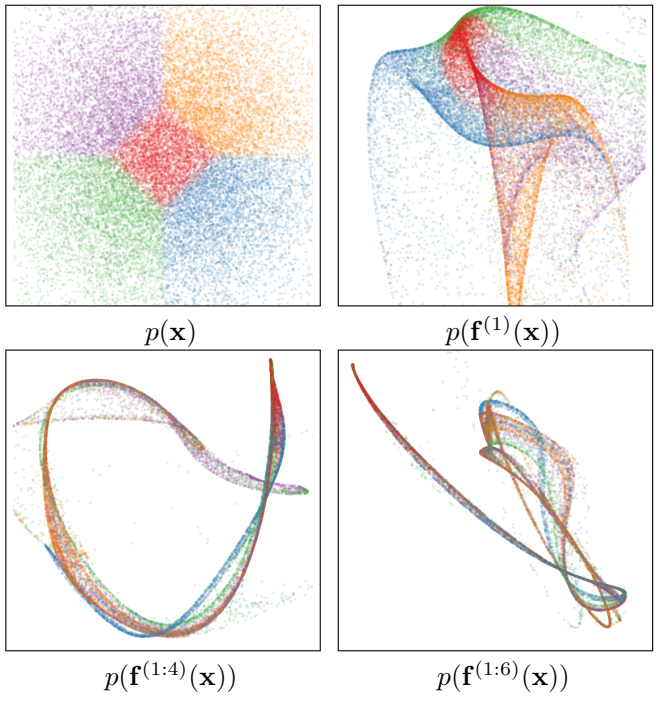


Figure 4: Normalized singular value spectrum of the Jacobian of a deep GP. As the net gets deeper, the largest singular value dominates. This implies that with high probability, there is only one effective degree of freedom in the representation being computed. As depth increases, the distribution on singular values also becomes heavy-tailed.



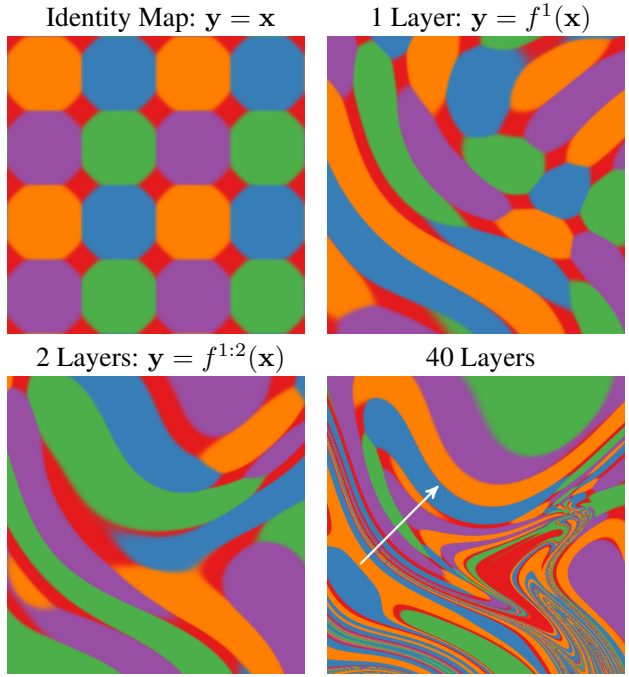
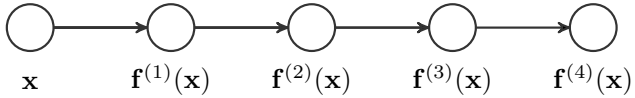


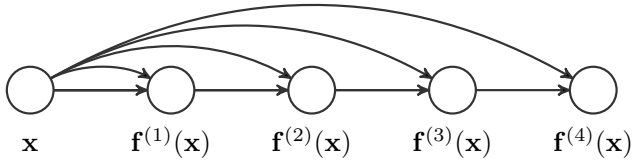
Figure 6: Feature Mapping of a deep GP. Colors correspond to the location $\mathbf{y} = \mathbf{f}(\mathbf{x})$ that each point is mapped to after being warped by a deep GP. Just as the densities in figure 5 became locally one-dimensional, there is usually only one direction that one can move $\mathbf{x}$ in locally to change $\mathbf{y}$. The number of directions in which the color changes rapidly corresponds to the number of large singular values in the Jacobian.

6.1 Jacobians of Input-connected Deep Networks

We can similarly examine the Jacobians of the input-connected architecture. The Jacobian of the composed, input-connected deep function is defined by the recurrence: $J^{1:L}(\mathbf{x}) = J^L \begin{bmatrix} J^{1:L-1} \\ I_D \end{bmatrix}$. Figure 10 shows that with this



a) The standard MLP connectivity architecture.



b) Input-connected architecture.

Figure 7: Two different architectures for deep neural networks. The standard architecture connects each layer's outputs to the next layer's inputs. The input-connected architecture connects also connects the original input $\mathbf{x}$ to each layer.

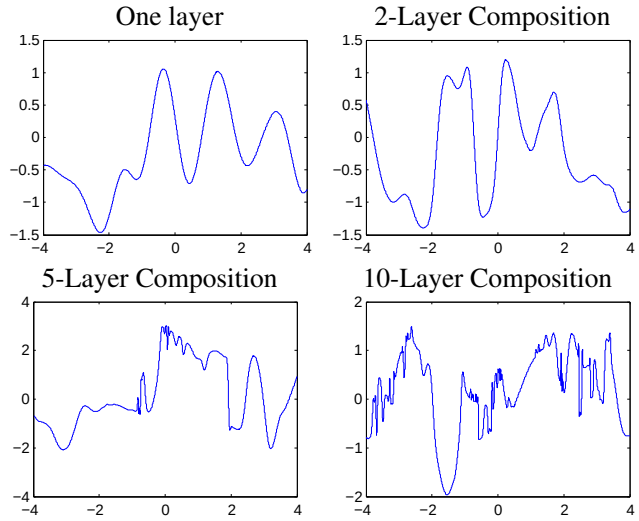
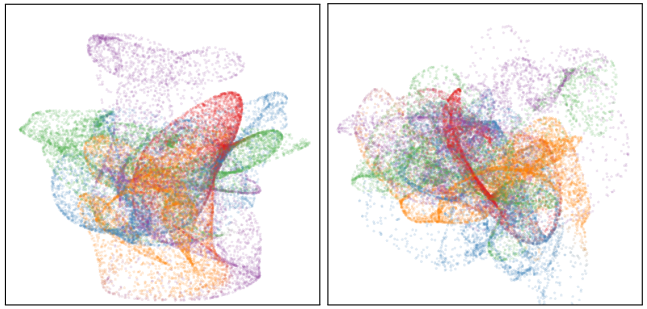


Figure 8: Draws from a 1D deep GP prior with each layer connected to the input. Even after many layers, the functions remain smooth in some regions, while varying rapidly



4 Layers 5 Layers

Figure 9: Left: Densities defined by a draw from a deep GP, with each layer connected to the input $\mathbf{x}$. As depth increases, the density becomes more complex without concentrating along filaments.

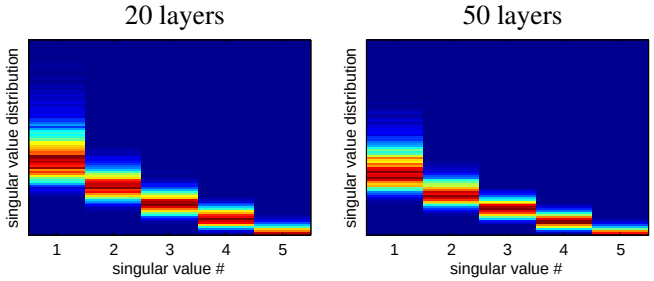


Figure 10: The distribution of singular values drawn from a 5-dimensional input-connected deep GP prior 25 and 50 layers deep. The singular values remain roughly the same scale as one another.

7.1 Infinitely Deep Kernels

What happens when repeat this feature mapping many times, starting with the squared-exp kernel? In the infinite limit, this recursion converges to $k(\mathbf{x}, \mathbf{x}') = 1$ for all pairs of inputs. One interpretation of why repeated feature transforms lead to this degenerate prior is that each layer can only lose information about the previous set of features. In

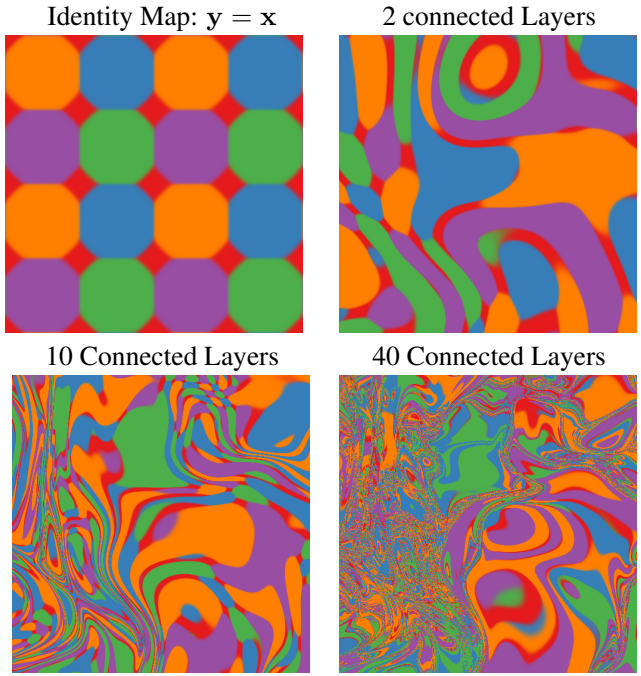
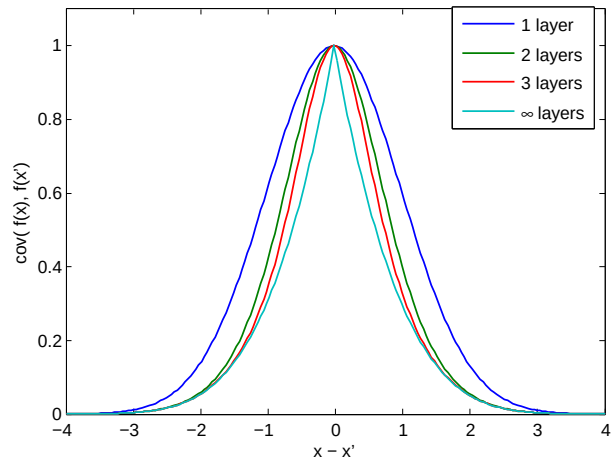


Figure 11: Feature Mapping of a deep GP with each layer connected to the input $\mathbf{x}$. Just as the densities in figure 9 remained locally two-dimensional even after many transformations, in this mapping there are usually two directions that one can move locally in $\mathbf{x}$ to change $\mathbf{y}$.

This kernel satisfies the recurrence $k - \log(k) = 1 + \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2$, a non-degenerate limit. The solution to this recurrence has no closed form, but it is continuous and differentiable everywhere except at $\mathbf{x} = \mathbf{x}'$. Samples from a GP with this prior are not differentiable, having a similar shape to the Ornstein-Uhlenbeck covariance: $\exp(-|x - x'|)$, but with lighter tails.

7.2 Can Deep Kernels Be Useful Models?

Bengio *et al.* (2006) showed that kernel machines, such as



Connected transform kernel

Figure 12: A non-degenerate version of the infinitely deep feature transform kernel. By connecting the inputs $\mathbf{x}$ to each layer, the function can still depend on its input even after arbitrarily many layers of computation.

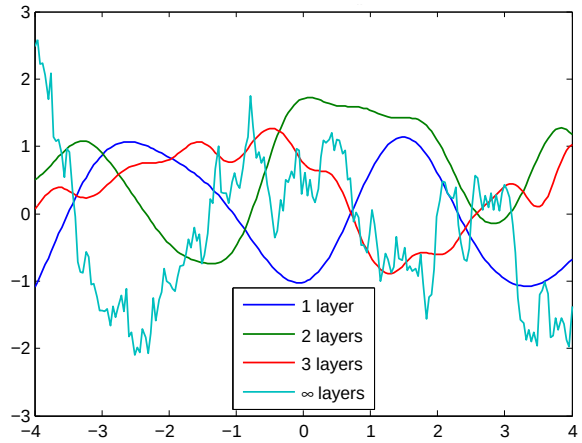
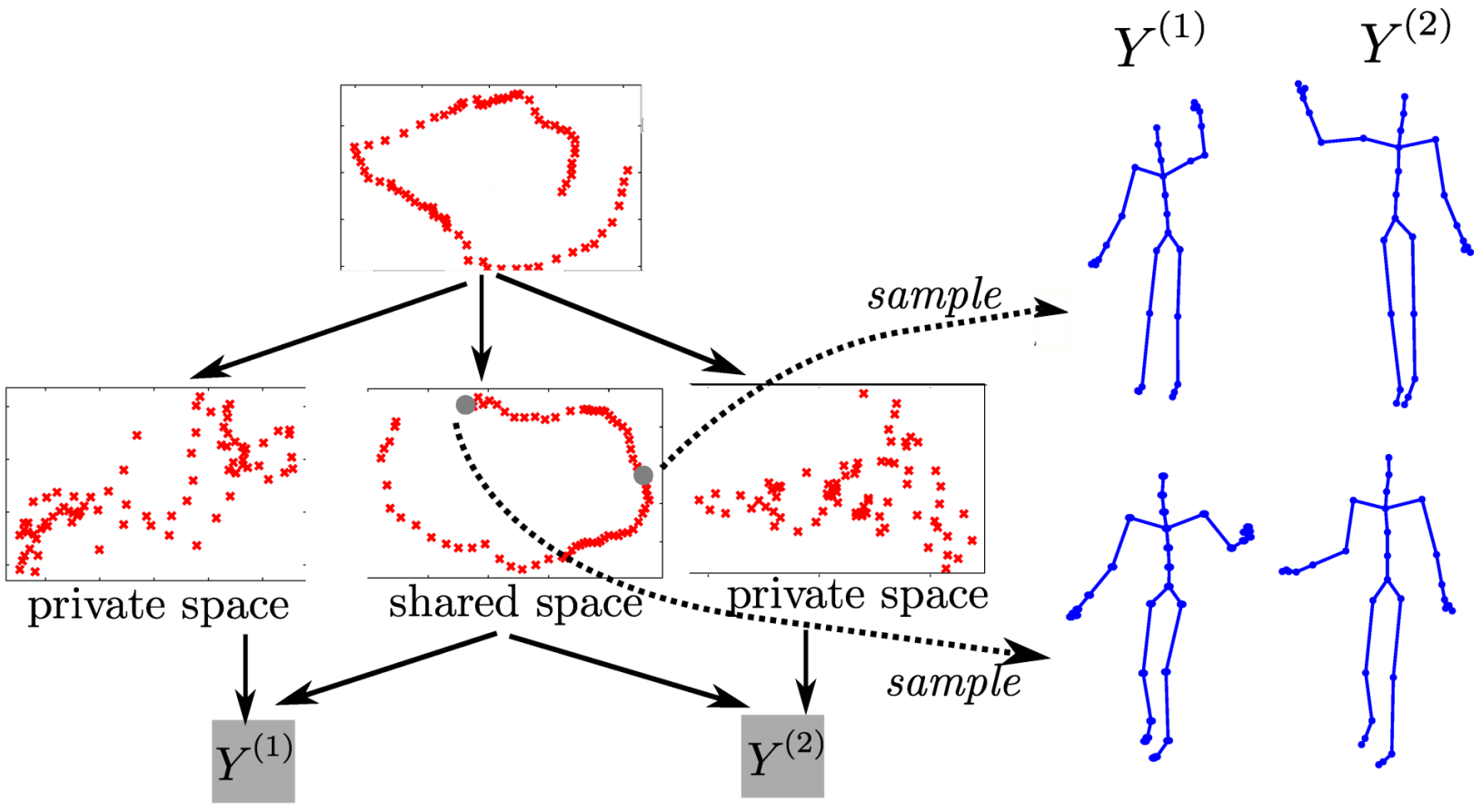


Figure 13: Draws from the deep input-connected kernel.

Motion Capture

- Revisit 'high five' data.
- This time allow model to learn structure, rather than imposing it.

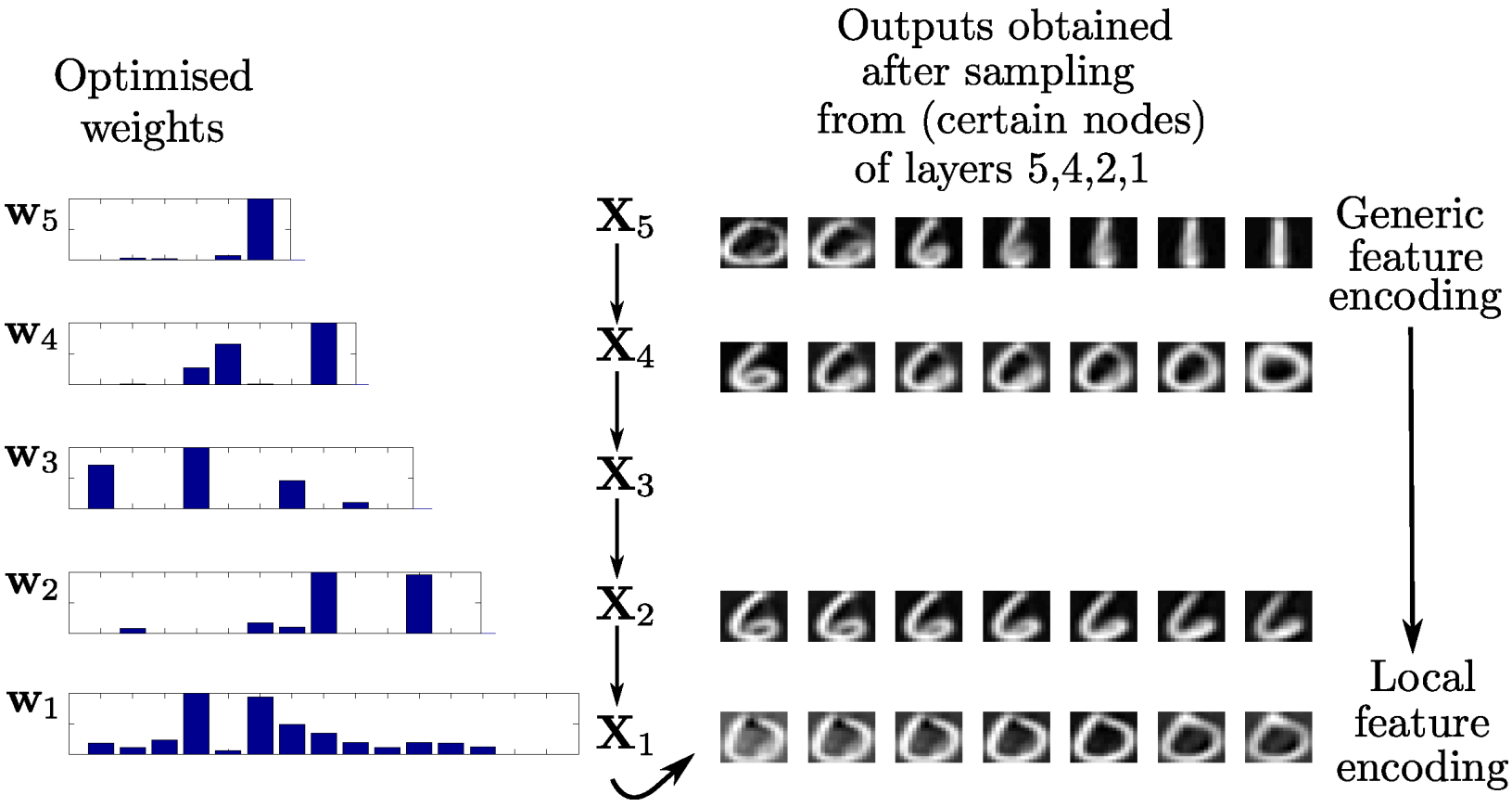
Deep hierarchies – motion capture



Digits Data Set

- Are deep hierarchies justified for small data sets?
- We can lower bound the evidence for different depths.
- For 150 6s, 0s and 1s from MNIST we found at least 5 layers are required.

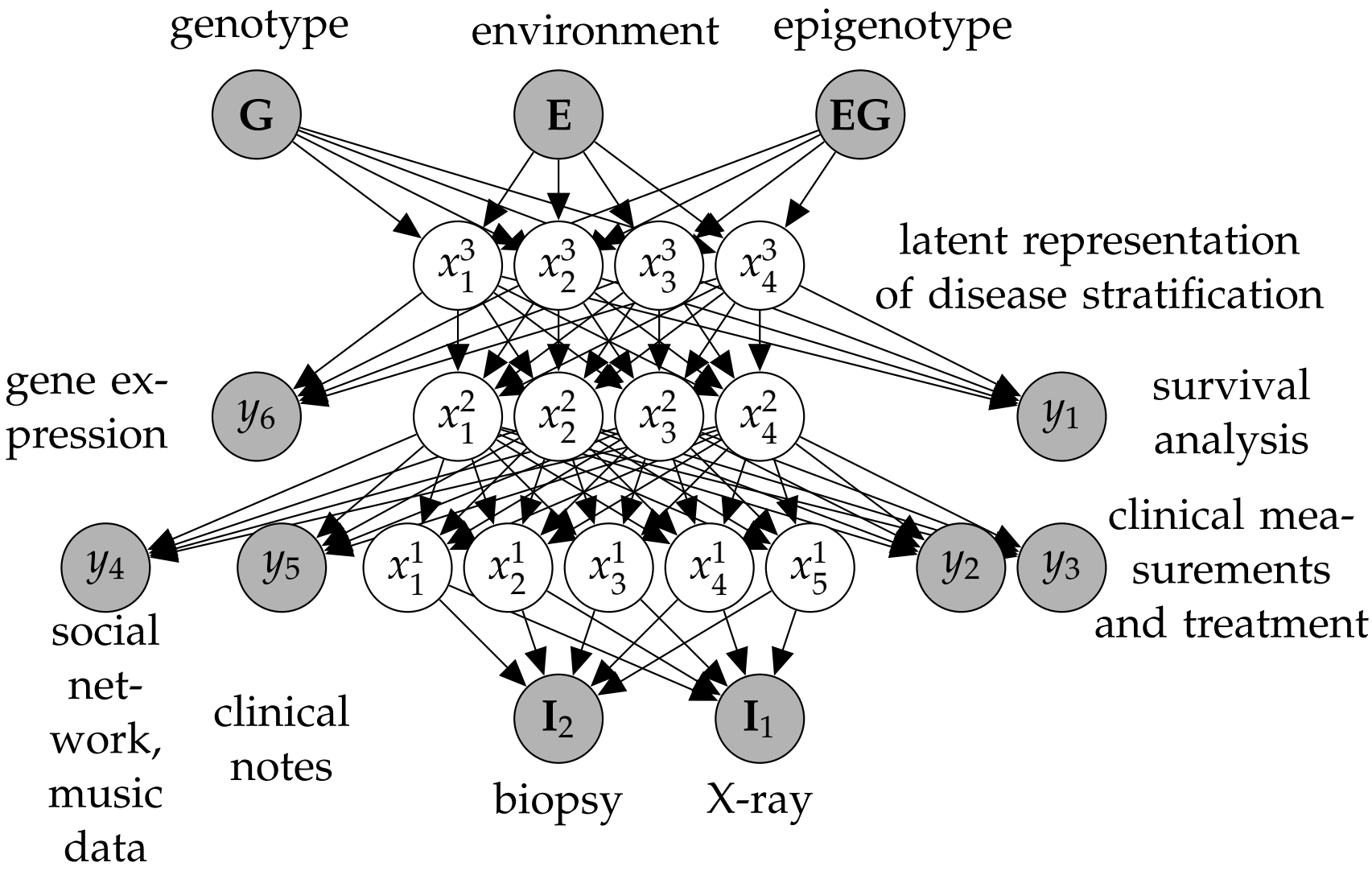
Deep hierarchies – MNIST



What Can We Do that Google Can't?

- Google's resources give them access to volumes of data (or Facebook, or Microsoft, or Amazon).
- Is there anything for Universities to contribute?
- Assimilation of multiple views of the patient: each perhaps from a different patient.
- This may be done by small companies (with support of Universities).
- A Facebook app for your personalised health.
- These methodologies are part of that picture.

Deep Health


genotype
environment
epigenotype
G
E
EG
x_1^3 x_2^3 x_3^3 x_4^3
latent representation of disease stratification
gene expression
y_6
x_1^2 x_2^2 x_3^2 x_4^2
y_1
survival analysis
y_4
y_5
x_1^1 x_2^1 x_3^1 x_4^1 x_5^1
y_2
y_3
clinical measurements and treatment
social network, music data
clinical notes
I_2
I_1
biopsy
X-ray

Deep Health: Power Ranger Model of Research

Thanks to Alan Saul for creating the image.

Summary

- Variational GP-LVM gives dimensionality estimation in non linear PCA.
- Shared models use structure learning to do manifold relevance determination.
- Temporal models place a GP prior on the latent space to ensure time dependence of variables.
- Deep GPs place GP-LVM priors on each layer recursively.

References I

Y. Bengio. Learning Deep Architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, Jan. 2009. ISSN 1935-8237. [DOI].

L. Csató. *Gaussian Processes — Iterative Sparse Approximations*. PhD thesis, Aston University, 2002.

L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

A. Damianou, C. H. Ek, M. K. Titsias, and N. D. Lawrence. Manifold relevance determination. In J. Langford and J. Pineau, editors, *Proceedings of the International Conference in Machine Learning*, volume 29, San Francisco, CA, 2012. Morgan Kauffman. [PDF].

A. Damianou and N. D. Lawrence. Deep Gaussian processes. In C. Carvalho and P. Ravikumar, editors, *Proceedings of the Sixteenth International Workshop on Artificial Intelligence and Statistics*, volume 31, AZ, USA, 2013. JMLR W&CP 31. [PDF].

A. Damianou, M. K. Titsias, and N. D. Lawrence. Variational Gaussian process dynamical systems. In P. Bartlett, F. Peirrera, C. Williams, and J. Lafferty, editors, *Advances in Neural Information Processing Systems*, volume 24, Cambridge, MA, 2011. MIT Press. [PDF].

D. Duvenaud, O. Rippel, R. Adams, and Z. Ghahramani. Avoiding pathologies in very deep networks. In S. Kaski and J. Corander, editors, *Proceedings of the Seventeenth International Workshop on Artificial Intelligence and Statistics*, volume 33, Iceland, 2014. JMLR W&CP 33.

C. H. Ek, J. Rihan, P. Torr, G. Rogez, and N. D. Lawrence. Ambiguity modeling in latent spaces. In A. Popescu-Belis and R. Stiefelhagen, editors, *Machine Learning for Multimodal Interaction (MLMI 2008)*, LNCS, pages 62–73. Springer-Verlag, 28–30 June 2008a. [PDF].

C. H. Ek, P. H. Torr, and N. D. Lawrence. Gaussian process latent variable models for human pose estimation. In A. Popescu-Belis, S. Renals, and H. Bourlard, editors, *Machine Learning for Multimodal Interaction (MLMI 2007)*, volume 4892 of *LNCS*, pages 132–143, Brno, Czech Republic, 2008b. Springer-Verlag. [PDF].

Z. Ghahramani, editor. *Proceedings of the International Conference in Machine Learning*, volume 24, 2007. Omnipress. [Google Books] .

J. Hensman, M. Rattray, and N. D. Lawrence. Fast variational inference in the conjugate exponential family. In P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, Cambridge, MA, 2012. [PDF].

References II

G. E. Hinton and S. Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:2006, 2006.

N. J. King and N. D. Lawrence. Fast variational inference for Gaussian Process models through KL-correction. In *ECML, Berlin, 2006*, Lecture Notes in Computer Science, pages 270–281, Berlin, 2006. Springer-Verlag. [PDF].

A. Klami and S. Kaski. Local dependent components analysis. In Ghahramani (2007). [Google Books] .

A. Klami and S. Kaski. Probabilistic approach to detecting dependencies between data sets. *Neurocomputing*, 72: 39–46, 2008.

N. D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In M. Meila and X. Shen, editors, *Proceedings of the Eleventh International Workshop on Artificial Intelligence and Statistics*, pages 243–250, San Juan, Puerto Rico, 21-24 March 2007. Omnipress. [PDF].

N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In Ghahramani (2007), pages 481–488. [Google Books] . [PDF].

G. Leen and C. Fyfe. A Gaussian process latent variable model formulation of canonical correlation analysis. Bruges (Belgium), 26-28 April 2006 2006.

T. K. Leen, T. G. Dietterich, and V. Tresp, editors. *Advances in Neural Information Processing Systems*, volume 13, Cambridge, MA, 2001. MIT Press.

D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, U.K., 2003. [Google Books] .

R. Navaratnam, A. Fitzgibbon, and R. Cipolla. The joint manifold model for semi-supervised multi-valued regression. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society Press, 2007.

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

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] .

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In S. Roweis and A. McCallum, editors, *Proceedings of the International Conference in Machine Learning*, volume 25, pages 872–879. Omnipress, 2008.

References III

M. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In C. M. Bishop and B. J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 Jan 2003.

A. P. Shon, K. Grochow, A. Hertzmann, and R. P. N. Rao. Learning shared latent structure for image synthesis and robotic imitation. In Weiss et al. (2006).

A. J. Smola and P. L. Bartlett. Sparse greedy Gaussian process regression. In Leen et al. (2001), pages 619–625.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Weiss et al. (2006).

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelfth International Workshop on Artificial Intelligence and Statistics*, volume 5, pages 567–574, Clearwater Beach, FL, 16-18 April 2009. JMLR W&CP 5.

M. K. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. In Y. W. Teh and D. M. Titterington, editors, *Proceedings of the Thirteenth International Workshop on Artificial Intelligence and Statistics*, volume 9, pages 844–851, Chia Laguna Resort, Sardinia, Italy, 13-16 May 2010. JMLR W&CP 9. [PDF].

L. R. Tucker. An inter-battery method of factor analysis. *Psychometrika*, 23(2):111–136, 1958.

S. Virtanen, A. Klami, and S. Kaski. Bayesian CCA via group sparsity. In L. Getoor and T. Scheffer, editors, *Proceedings of the International Conference in Machine Learning*, volume 28, 2011.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008. ISSN 0162-8828. [DOI].

Y. Weiss, B. Schölkopf, and J. C. Platt, editors. *Advances in Neural Information Processing Systems*, volume 18, Cambridge, MA, 2006. MIT Press.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In Leen et al. (2001), pages 682–688.