Bayesian Linear Regression

- Bayesian treatment: avoids the over-fit and leads to an automatic way of determining the model complexity using only the training data.
- We start by defining a simple likelihood conjugate prior,
- For example, a zero-mean Gaussian prior governed by a precision parameter:

$$p(\boldsymbol{w}|\alpha) = N(w|\mathbf{0}, \alpha^{-1}\boldsymbol{I})$$

This prior, when combines with the "least squares" likelihood via Bayes rule, yields the posterior distribution:

$$p(\mathbf{w} \mid \vec{t}, X) = \mathcal{N}\left(\mathbf{w}; \mathbf{m}_N, \mathbf{S}_N\right)$$

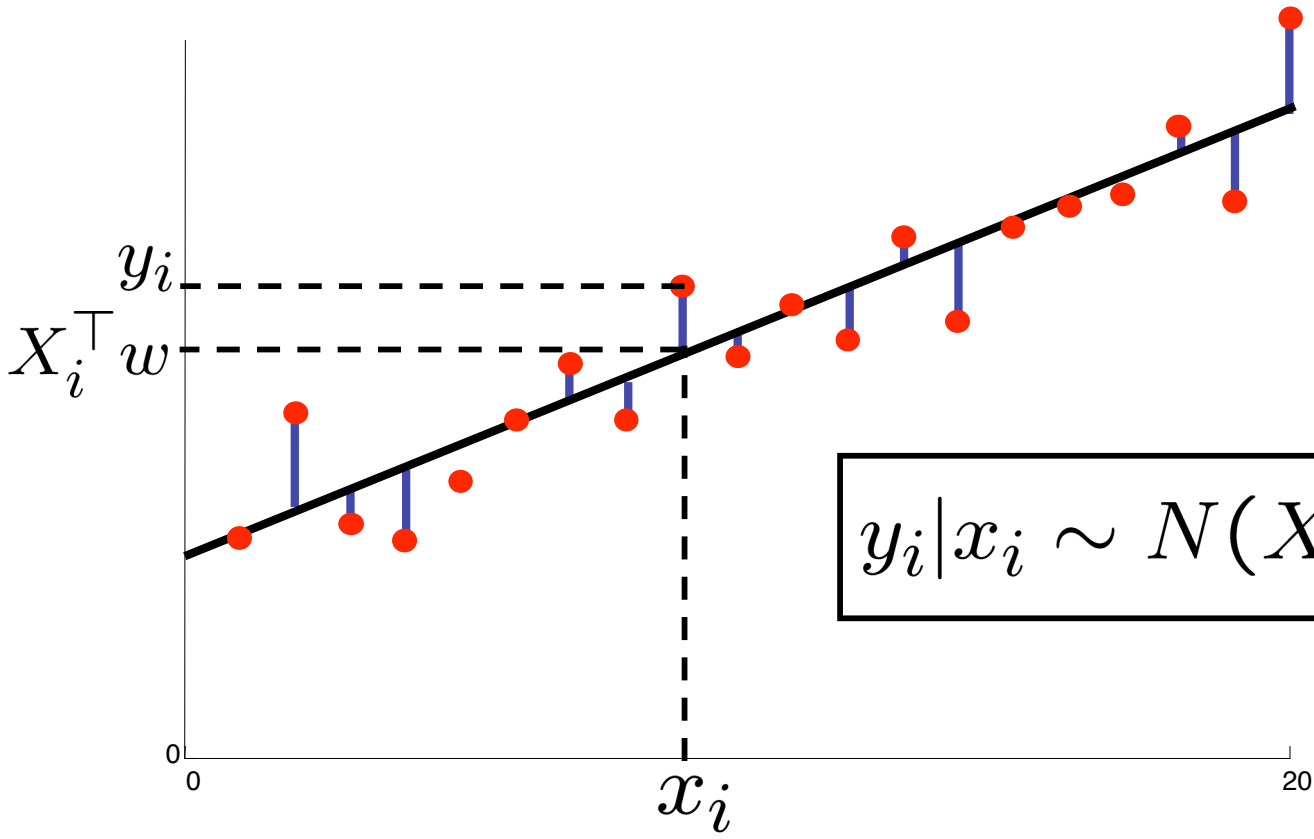
For example, we will show that for the prior above, and for a matrix of transformed predictors $\boldsymbol{\Phi}$

$$p(\boldsymbol{w}|t) = N(\boldsymbol{w}|\boldsymbol{m}_N, \boldsymbol{S}_N) = N(\boldsymbol{w}|\beta \boldsymbol{S}_N^{-1} \Phi^T \mathbf{t}, \alpha \boldsymbol{I} + \beta \Phi^T \Phi)$$

In other words:

$$\mathbf{w} = \beta(\Phi^T\Phi + \alpha\mathbf{I})^{-1}\Phi^T\vec{t}$$
$$\text{cov}[\mathbf{w}] = (\Phi^T\Phi + \alpha\mathbf{I})$$

RECALL-Probabilistic interpretation



Likelihood $L = \prod_i \exp -\frac{1}{2\sigma^2}(X_i^\top w - y_i)^2 = \exp -\frac{1}{2\sigma^2}\sum_i (X_i^\top w - y_i)^2$

$$\operatorname*{argmax}_w L = \operatorname*{argmin}_w E$$

Statistical Models

$$Y = f(X) + \varepsilon$$

with $\mathrm{E}(\varepsilon) = 0$ and X and ε independent.

- $\mathrm{E}(Y|X) = f(X)$
- $\Pr(Y|X)$ depends on X only through $f(X)$.
- Useful approximation to the truth — all unmeasured variables captured by ε
- N realizations $y_i = f(x_i) + \varepsilon_i,\ i = 1, \ldots, N$
- Assume ε_i and ε_j are independent.

More generally can have, for example, $\mathrm{Var}(Y|X) = \sigma^2(X)$.

For qualitative outcomes $\{\Pr(G = \mathcal{G}_k|X)\}_1^K = p(X)$ which we model directly.

Bayesian Linear Regression

- Given target values, modeled as a sum of basis functions plus Gaussian noise

$$t = y(X, w) + \epsilon$$

$$y(x, w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = \mathbf{w}^T \boldsymbol{\phi}(x)$$

- Then the likelihood is Gaussian

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1})$$

- Assur

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$$

Bayesian Regression

- The posterior is straightforward to derive. The computations follow the classification case:

$$p(\mathbf{w} \mid \vec{t}, X, \beta) = N(\mathbf{w} \mid \mu_N, S_N)$$
$$\propto p(\vec{t} \mid \mathbf{w}, X, \beta)\, p(\mathbf{w})$$
$$= \mathcal{N}(\vec{t} \mid \mathbf{w}^T \mathbf{\Phi}, \beta^{-1}\mathbf{I})\mathcal{N}(\mathbf{w} \mid \mu_0, S_0)$$

$$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$$

Where the posterior mean and variance are given by:

$$\mathbf{m}_N = \mathbf{S}_N \left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta \mathbf{\Phi}^{\mathrm{T}}\mathbf{t}\right)$$
$$\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta \mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}.$$

Given that β is the noise in the measurements

Basic Results for Gaussians

- The results stem from the fact that multiplying two Gaussians is Gaussian (although no longer normalized). In particular,

$$\mathcal{N}(\mathbf{a}, \mathbf{A}) \cdot \mathcal{N}(\mathbf{b}, \mathbf{B}) \propto \mathcal{N}(\mathbf{c}, \mathbf{C})$$

where

$$\mathbf{C} = \left(\mathbf{A}^{-1} + \mathbf{B}^{-1}\right)^{-1}$$
$$\mathbf{c} = \mathbf{C}\mathbf{A}^{-1}\mathbf{a} + \mathbf{C}\mathbf{B}^{-1}\mathbf{b}$$

amazingly, the normalization constant z_c is Gaussian in either $\mathbf{a}$ or $\mathbf{b}$:

$$z_c = (2\pi)^{-d/2}|\mathbf{C}|^{+1/2}|\mathbf{A}|^{-1/2}|\mathbf{B}|^{-1/2}\exp\left[-\frac{1}{2}(\mathbf{a}^T\mathbf{A}^{-1}\mathbf{a} + \mathbf{b}^T\mathbf{B}^{-1}\mathbf{b} - \mathbf{c}^T\mathbf{C}^{-1}\mathbf{c})\right]$$

$$z_c(\mathbf{a}) \sim \mathcal{N}\left((\mathbf{A}^{-1}\mathbf{C}\mathbf{A}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{C}\mathbf{B}^{-1})\mathbf{b}, (\mathbf{A}^{-1}\mathbf{C}\mathbf{A}^{-1})^{-1}\right)$$
$$z_c(\mathbf{b}) \sim \mathcal{N}\left((\mathbf{B}^{-1}\mathbf{C}\mathbf{B}^{-1})^{-1}(\mathbf{B}^{-1}\mathbf{C}\mathbf{A}^{-1})\mathbf{a}, (\mathbf{B}^{-1}\mathbf{C}\mathbf{B}^{-1})^{-1}\right)$$

Special Cases

- Prior on **w**

$$\mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

- Ridge Regression Posterior on **w**

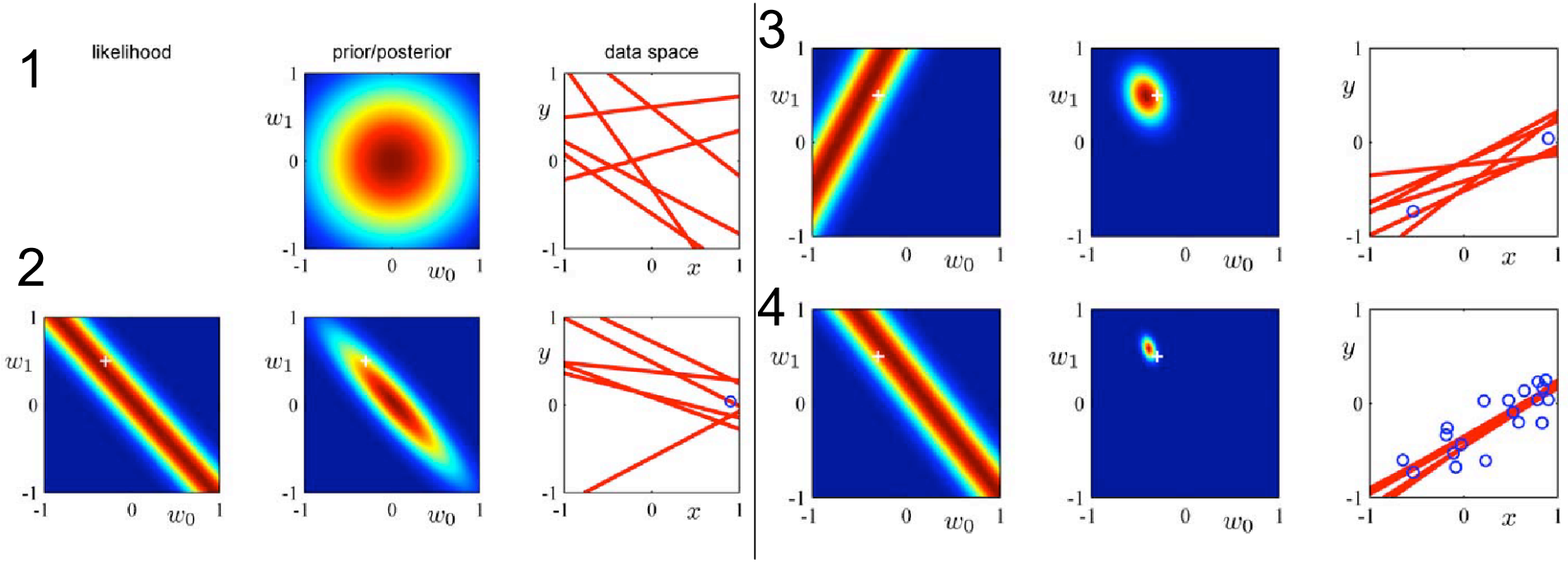
$$\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \mathbf{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi}. \end{aligned}$$

From Ridge to Lasso etc

- A family of priors for regularized regression:

$$p(\mathbf{w}|\alpha) = \left[\frac{q}{2}\left(\frac{\alpha}{2}\right)^{1/q}\frac{1}{\Gamma(1/q)}\right]^M \exp\left(-\frac{\alpha}{2}\sum_{j=1}^{M}|w_j|^q\right)$$

Visualizing Bayesian Regression



Sequential Bayesian Learning: As each data point comes in,the posterior on **w** is updated. Lines show samples from the posterior distribution.

1 No Data
2 One data point
3 Two data points
4 Twenty data points

Predictive Distribution

- To predict new datapoints, we need to marginalize the basic regression model across our uncertainty in the regression coefficients (model averaging

$$
\begin{aligned}
p\left(y_{pred}\middle|\vec{t},\alpha,\beta\right) &= \int p\left(y_{pred}\middle|\mathbf{w},\beta\right)p(\mathbf{w}\,|\,\vec{t},X,\alpha,\beta)d\mathbf{w} \\
&= \int \mathcal{N}\left(y_{pred}\,|\,\Phi\mathbf{w},\beta^{-1}\mathbf{I}\right)\mathcal{N}\left(\mathbf{w}\,|\,S_N^{-1}\left(S_0^{-1}\mu_0+\beta\Phi^T\vec{t}\right),S_N\right)d\mathbf{w} \\
&= \int \mathcal{N}\left(y_{pred}\,|\,\Phi\mathbf{w},\beta^{-1}\mathbf{I}\right)\mathcal{N}\left(\mathbf{w}\,|\,\mathbf{m}_N,S_N\right)d\mathbf{w} \\
&= \mathcal{N}\left(y_{pred}\,|\,\Phi\mathbf{m}_N,\beta^{-1}\mathbf{I}+\Phi S_N\Phi^T\right)
\end{aligned}
$$

where

$$S_N = S_0^{-1} + \beta\Phi^T\Phi$$

More Results

Marginal and Conditional Gaussians

Given a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution for $\mathbf{y}$ given $\mathbf{x}$ in the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) \quad (2.113)$$

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1}) \quad (2.114)$$

the marginal distribution of $\mathbf{y}$ and the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ are given by

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}}) \quad (2.115)$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Sigma}\{\mathbf{A}^{\mathrm{T}}\mathbf{L}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Lambda}\boldsymbol{\mu}\}, \boldsymbol{\Sigma}) \quad (2.116)$$

where

$$\boldsymbol{\Sigma} = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}. \quad (2.117)$$

Predictive Distribution

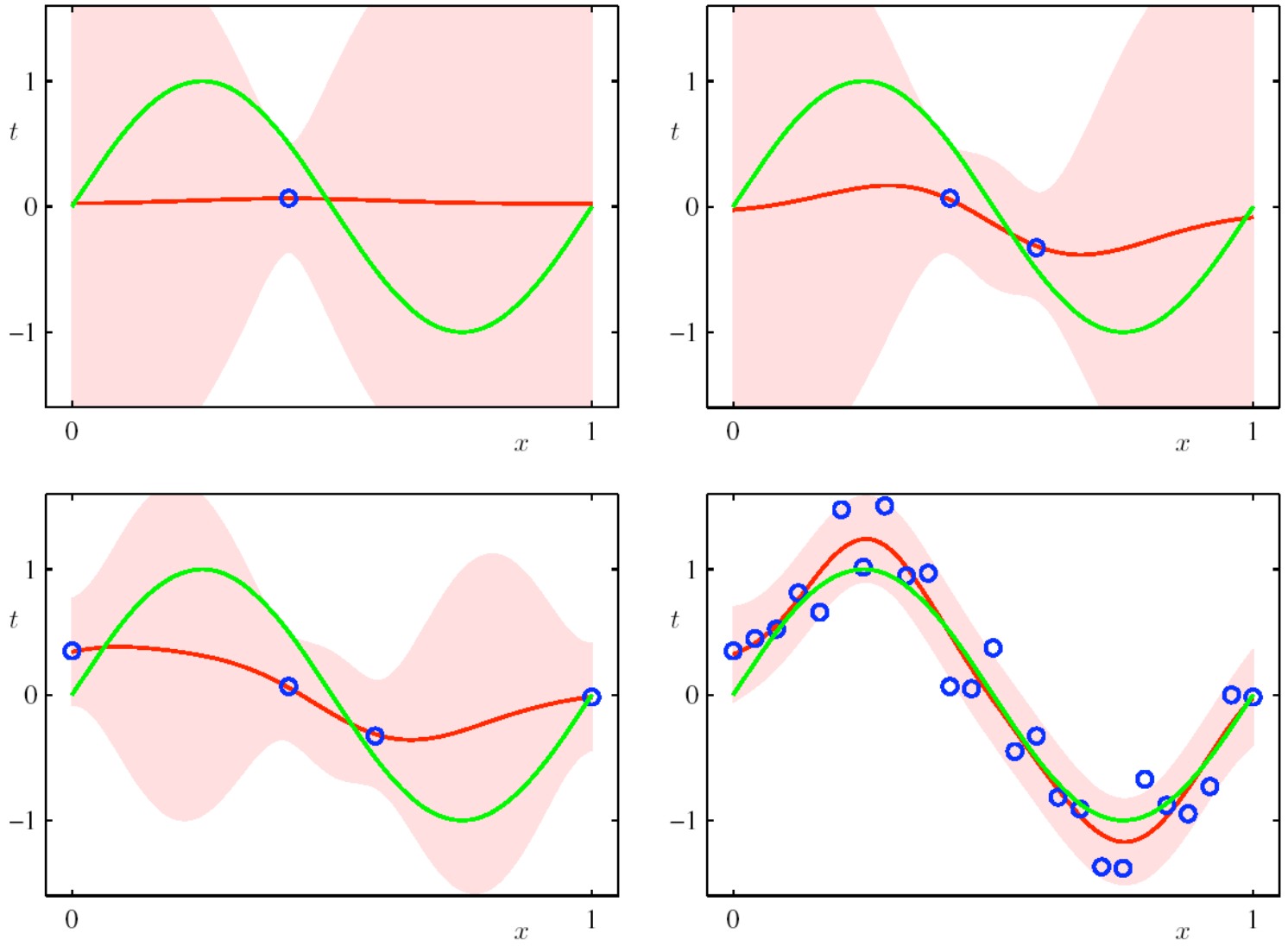


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis functions of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

Bayes does model averaging: with the average across set of **w**

Samples from the Posterior Distribution on **w**

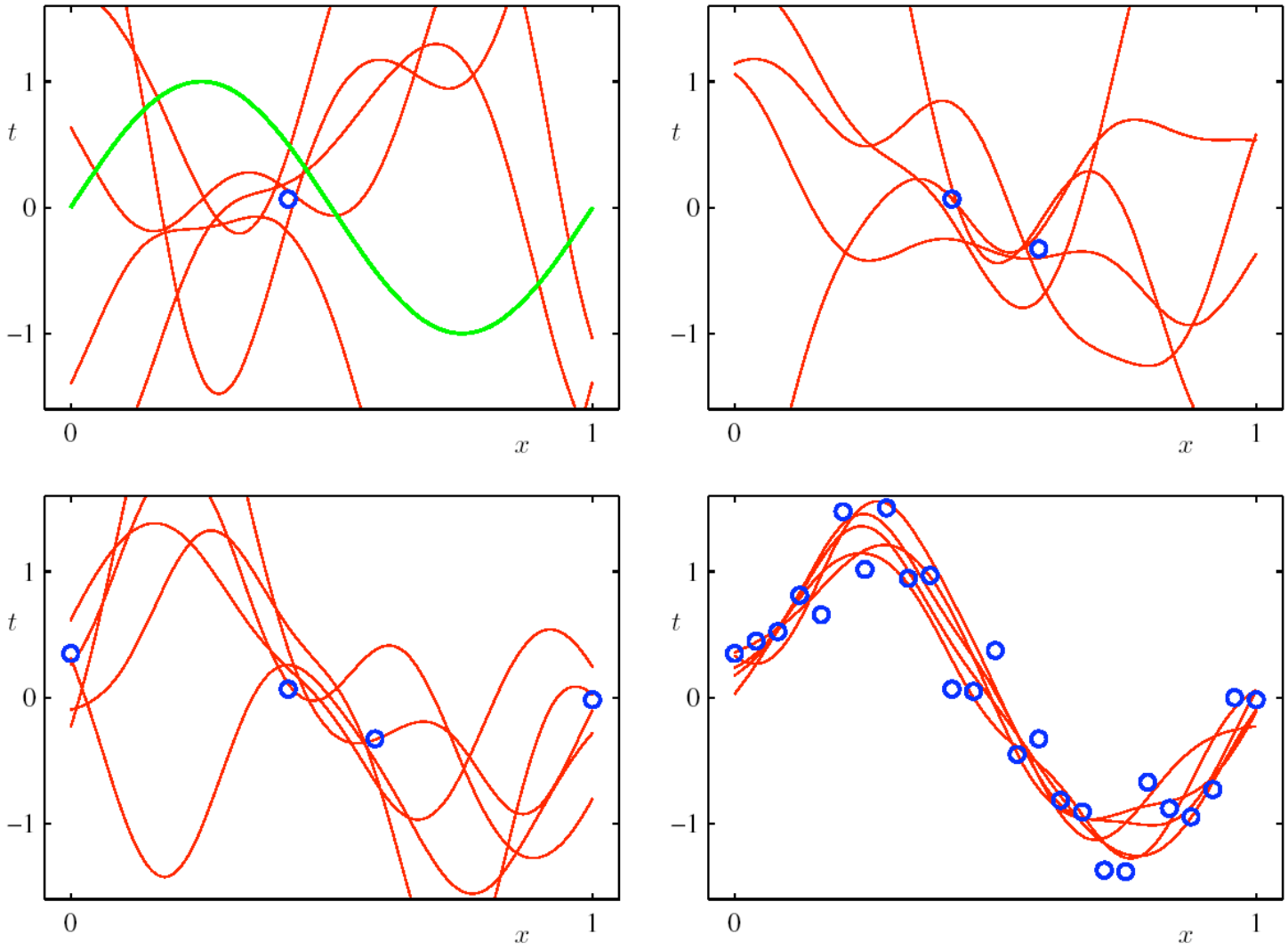


Figure 3.9 Plots of the function $y(x, \mathbf{w})$ using samples from the posterior distributions over $\mathbf{w}$ corresponding to the plots in Figure 3.8.

Posterior mean on **w** and the equivalent kernel

3.3.3 Equivalent kernel

The posterior mean solution (3.53) for the linear basis function model has an interesting interpretation that will set the stage for kernel methods, including Gaussian processes. If we substitute (3.53) into the expression (3.3), we see that the predictive mean can be written in the form

$$y(\mathbf{x}, \mathbf{m}_N) = \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} = \sum_{n=1}^{N} \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}_n) t_n \qquad (3.60)$$

where $\mathbf{S}_N$ is defined by (3.51). Thus the mean of the predictive distribution at a point $\mathbf{x}$ is given by a linear combination of the training set target variables t_n, so that we can write

$$y(\mathbf{x}, \mathbf{m}_N) = \sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) t_n \qquad (3.61)$$

where the function

$$k(\mathbf{x}, \mathbf{x}') = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}') \qquad (3.62)$$

Another Meaning for the Kernel

- In the Bayesian framework, it is easy to show that the equivalent kernel is the covariance matrix of the predictive distribution.

$$\begin{aligned}\mathrm{cov}[y(\mathbf{x}), y(\mathbf{x}')] &= \mathrm{cov}[\boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{w}, \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}')] \\ &= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}') = \beta^{-1}k(\mathbf{x}, \mathbf{x}')\end{aligned}$$

What these results show is that Bayesian regression can be

Viewed as a kernel based algorithm. Rather than choosing the

Weights on a fixed set of kernels (SVR), the kernels are constructed from the data modeling assumptions.

Gaussian Processes in Machine Learning

Outline of the talk

- Gaussian Processes (GP) [ma05, rs03]
  - Bayesian Inference
  - GP for regression
  - Optimizing the hyperparameters
- Applications
  - GP Latent Variable Models [la04]
  - GP Dynamical Models [wa05]

GP: Introduction

- Gaussian Processes:
  - Definition: A GP is a collection of random variables, any finite number of which have joint Gaussian Distribution
    - Distribution over functions:
      - $f \sim \mathcal{GP}(m,\ k)$ $m(x)$ $k(x, x')$
    - Gaussian Distribution: over vectors
      - $\mathbf{f} \sim \mathcal{N}(\mu, \Sigma)$ $\mu_i = m(x_i)$ $\Sigma_{ij} = k(x_i, x_j)$
- Nonlinear Regression:
  - X_N … Data Points
  - t_N … Target Vector
- Infer Nonlinear parameterized function, *y(x;w)*, predict values t_{N+1} for new data points x_{N+1}
- E.g. Fixed Basis Functions
  - $$y(\mathbf{x}; \mathbf{w}) = \sum_{h=1}^{H} w_h \phi_h(\mathbf{x}) \qquad \phi_h(\mathbf{x}) = \exp\left[-\frac{(\mathbf{x} - \mathbf{c}_h)^2}{2r^2}\right]$$

Bayesian Inference of the parameters

- Posterior propability of the parameters:

$$P(\mathbf{w}|\mathbf{t}_N, \mathbf{X}_N) = \frac{P(\mathbf{t}_N|\mathbf{w}, \mathbf{X}_N)P(\mathbf{w})}{P(\mathbf{t}_N|\mathbf{X}_N)}$$

$P(\mathbf{t}_N|\mathbf{w}, \mathbf{X}_N)$ bability that the observed data points have been generated by y(x;w)

  - Often separable Gaussian distribution is used
    - Each data point t_i differing from $y(x_i;w)$ by additive noise

$P(\mathbf{w})$ priors on the weights

- Prediction is made by marginalizing over the parameters

$$P(t_{N+1}|\mathbf{t}_N, \mathbf{X}_{N+1}) = \int d^H\mathbf{w}\, P(t_{N+1}|\mathbf{w}, \mathbf{x}^{(N+1)})P(\mathbf{w}|\mathbf{t}_N, \mathbf{X}_N)$$

  - Sample parameters w from the distribution $P(\mathbf{w}|\mathbf{t}_N, \mathbf{X}_N)$ with Markov chain Monte Carlo techniques
  - Or Approximate $P(\mathbf{w}|\mathbf{t}_N, \mathbf{X}_N)$ with a Gaussian Distribution

Bayesian Inference: Simple Example

- GP: $P(t_{N+1}|\mathbf{t}_N, \mathbf{X}_{N+1})$ is a Gaussian distribution
- Example: H Fixed Basis functions, N input points
  - $$R_{nh} \equiv \phi_h(\mathbf{x}^{(n)}) \qquad y_n \equiv \sum_h R_{nh} w_h$$
  - Prior on w: $P(\mathbf{w}) = \text{Normal}(\mathbf{0}, \sigma_w^2 \mathbf{I})$
  - Calculate prior for y(**x**) :

$$\mathbf{Q} = \langle \mathbf{y}\mathbf{y}^{\mathrm{T}} \rangle = \langle \mathbf{R}\mathbf{w}\mathbf{w}^{\mathrm{T}}\mathbf{R}^{\mathrm{T}} \rangle = \mathbf{R} \langle \mathbf{w}\mathbf{w}^{\mathrm{T}} \rangle \mathbf{R}^{\mathrm{T}}$$

$$P(\mathbf{y}) = \text{Normal}(\mathbf{0}, \mathbf{Q}) = \text{Normal}(\mathbf{0}, \sigma_w^2 \mathbf{R}\mathbf{R}^{\mathrm{T}})$$

    - Prior for the target values
      - generated from y(x;w) + noise:

$$P(\mathbf{t}) = \text{Normal}(\mathbf{0}, \mathbf{Q} + \sigma_\nu^2 \mathbf{I})$$

    - Covariance Matrix: $\mathbf{C} = \mathbf{Q} + \sigma_\nu^2 \mathbf{I}$
    - Covariance Function $k(x_i, x_j) = \sigma_w^2 \phi_h(x_i) \phi_h(x_j) + \delta_{ij} \sigma_\nu^2$

Predicting Data

- Infer t_{N+1} given $\mathbf{t}_N$:
  - Simple, because conditional distribution is also a Gaussian
  - $$P(t_{N+1}|\mathbf{t}_N) = \frac{P(t_{N+1}, \mathbf{t}_N)}{P(\mathbf{t}_N)}$$
  - Use incremental form of

$$P(t_{N+1}|\mathbf{t}_N) \propto \exp\left[-\frac{1}{2}\begin{bmatrix} \mathbf{t}_N & t_{N+1} \end{bmatrix} \mathbf{C}_{N+1}^{-1} \begin{bmatrix} \mathbf{t}_N \\ t_{N+1} \end{bmatrix}\right]$$

$$\mathbf{C}_{N+1}$$

$$\mathbf{C}_{N+1} \equiv \begin{bmatrix} \begin{bmatrix} & \mathbf{C}_N & \end{bmatrix} & \begin{bmatrix} \mathbf{k} \\ \\ \end{bmatrix} \\ \begin{bmatrix} & \mathbf{k}^{\mathrm{T}} & \end{bmatrix} & \begin{bmatrix} \kappa \end{bmatrix} \end{bmatrix}$$

$$k^T = [k(x_{N+1}, x_1), \ldots, k(x_{N+1}, x_N)]$$
$$\kappa = k(x_{N+1}, x_{N+1})$$

Predicting Data

- We can rewrite this equation
  - Use partitioned inverse equations to get $\mathbf{C}_{N+1}^{-1}$ from $\mathbf{C}_N^{-1}$
  - $$P(t_{N+1}|\mathbf{t}_N) = \frac{1}{Z}\exp\left[-\frac{(t_{N+1}-\hat{t}_{N+1})^2}{2\sigma^2_{\hat{t}_{N+1}}}\right]$$

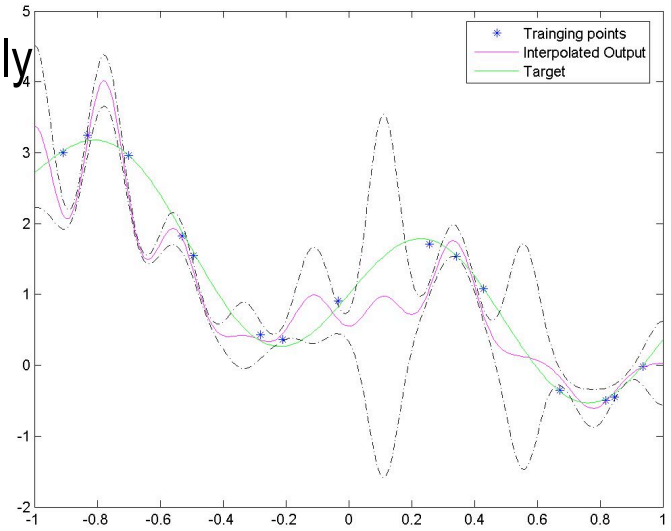
$$P(t_{n+1} \mid t_N) = Normal(\hat{t}_{N+1}, \sigma^2_{\hat{t}_{N+1}})$$

  - Predictive mean:
    - Usually used for the interpolation $\hat{t}_{N+1} = \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{t}_N$
  - Uncertainty in the result :

$$\sigma^2_{\hat{t}_{N+1}} = \kappa - \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{k}$$

Bayesian Inference: Simple Example

- How does the covariance matrix look like $\mathbf{C} = \mathbf{Q} + \sigma_\nu^2 \mathbf{I}$
  -
  - $Q_{nn'} = [\sigma_w^2 \mathbf{R}\mathbf{R}^{\mathrm{T}}]_{nn'} = \sigma_w^2 \sum \phi_h(\mathbf{x}^{(n)}) \phi_h(\mathbf{x}^{(n')})$
  - $C_{nn'} = \sigma_w^2 \sum_h \phi_h(\mathbf{x}^{(n)}) \phi_h(\mathbf{x}^{(n')}) + \delta_{nn'} \sigma_\nu^2$
  - as (due to the addition of I)
  - Simple Example: 10 RBF functions, uniformly distributed over the input space



Bayesian Inference: Simple Example

- Assume uniformly spaced basis functions,

$$
\begin{aligned}
Q_{nn'} &= S \int_{h_{\min}}^{h_{\max}} dh\, \phi_h(x^{(n)}) \phi_h(x^{(n')}) \\
&= S \int_{h_{\min}}^{h_{\max}} dh\, \exp\left[-\frac{(x^{(n)}-h)^2}{2r^2}\right] \exp\left[-\frac{(x^{(n')}-h)^2}{2r^2}\right]
\end{aligned}
$$

- Solution of the integral
  - Limits of integration to $\pm\infty$ $H \to \infty$
  - More general form

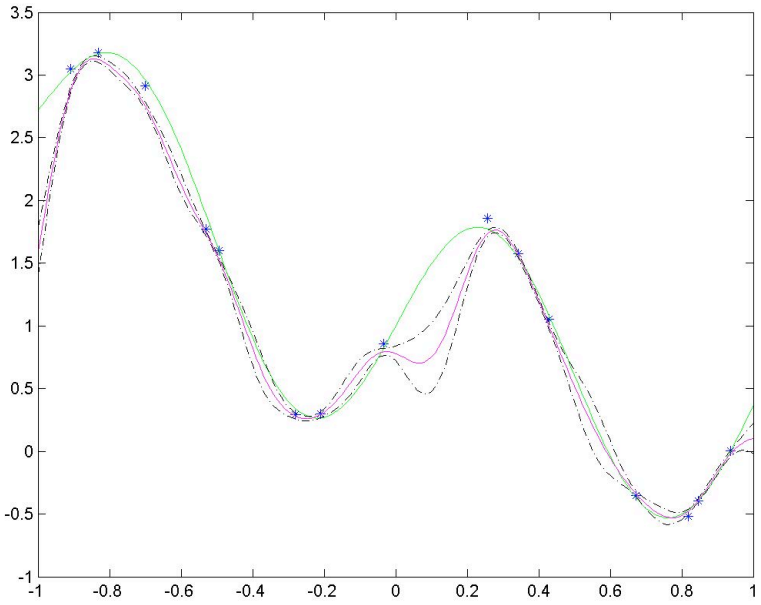
$$
Q_{nn'} = \sqrt{\pi r^2}\, S \exp\left[-\frac{(x^{(n')}-x^{(n)})^2}{4r^2}\right]
$$

$$
\widetilde{k}(x_n, x_m) = \alpha \exp\left(-\frac{\gamma}{2}(\mathbf{x}_n - \mathbf{x}_m)^{\mathrm{T}}(\mathbf{x}_n - \mathbf{x}_m)\right)
$$

$$
k(x_n, x_m) = \alpha \exp\left(-\frac{\gamma}{2}(\mathbf{x}_n - \mathbf{x}_m)^{\mathrm{T}}(\mathbf{x}_n - \mathbf{x}_m)\right) + \delta_{nm}\beta^{-1}
$$

Gaussian Processes

$$\hat{t}_{N+1} = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}_N \qquad \sigma^2_{\hat{t}_{N+1}} = \kappa - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k}$$



- Only C_N needs to be inverted ($O(N^3)$)
- Prediction depend entirely on C and the known targets t_N

Gaussian Processes: Covariance functions

- Must generate a non-negative definite covariance matrix for any set of points
  - $\{\Theta, \beta\}$
  - Hyperparameters of C
- Some Examples:
  - RBF:
  - Linear:

$$k(x_n, x_m; \Theta) = \tilde{k}(x_n, x_m; \Theta) + \delta_{nm}\beta^{-1}$$

$$\tilde{k}(x_n, x_m) = \alpha \exp\left(-\frac{\gamma}{2}(\mathbf{x}_n - \mathbf{x}_m)^{\mathrm{T}}(\mathbf{x}_n - \mathbf{x}_m)\right)$$

$$\tilde{k}(x_n, x_m) = \alpha \cdot x_n^T x_m + \gamma$$

- Some Rules:
  - Sum:
  - Product:
  - Product Spaces: ,

$$\tilde{k}(x_n, x_m) = \tilde{k}_2(x_n, x_m) + \tilde{k}_1(x_n, x_m)$$

$$\tilde{k}(x_n, x_m) = \tilde{k}_2(x_n, x_m) \cdot \tilde{k}_1(x_n, x_m)$$

$$\mathbf{z} = (x, y)$$

$$\tilde{k}(z_n, z_m) = \tilde{k}_2(x_n, x_m) + \tilde{k}_1(y_n, y_m)$$

Adaption of the GP models

- Hyperparameters: $\Theta = \{\alpha, \beta, \gamma\}$
  - Typically : $k_{n,m} = \alpha \exp\left(-\frac{\gamma}{2}(\mathbf{x}_n - \mathbf{x}_m)^{\mathrm{T}}(\mathbf{x}_n - \mathbf{x}_m)\right) + \delta_{nm}\beta^{-1}$
    - =>
- $P(\Theta \mid \mathbf{t}_N, \mathbf{X}_N) \propto P(\mathbf{t}_N \mid \mathbf{X}_N, \Theta)P(\Theta)$
- Log marginal Likelihood (first term)
  - $\mathcal{L} = -\frac{1}{2}\log\det \mathbf{C}_N - \frac{1}{2}\mathbf{t}_N^T \mathbf{C}_N^{-1}\mathbf{t}_N - \frac{N}{2}\log 2\pi$
  - Optimize via gradient descent (LM algorithm)
  - First term: complexity penalty term
    - => Occams Razor ! Simple models are prefered
  - Second term: Data-fit measure
- Priors on hyperparameters (second term)
  - Typically used: $P(\Theta) = \prod P(\theta_i) \propto \prod \theta_i^{-1}$
  - Prefer small parameters:
    - Small output scale ( ) α
    - Large width for the RBF ( ) γ
    - Large noise variance ( ) β
  - Additional mechanism to avoid overfitting

GP: Conclusion/Summary

- Memory-Based linear-interpolation method
- y(x) is uniquely defined by the definition of the C-function
- Also Hyperparameters are optimized
- Defined just for one output variable
  - Individual GP for each output variable
  - Use the same Hyperparameters
- Avoids overfitting
  - Tries to use simple models
  - We can also define priors
- No Methods for input data selection
- Difficult for a large input data set (Matrix inversion $O(N^3)$)
  - C_N can also be approximated, up to a few thousand input points possible
- Interpolation : No global generalization possible

Applications of GP

- Gaussian Process Latent Variable Models (GPLVM) [la04]
  - Style Based Inverse Kinematics [gr04]
  - Gaussian Process Dynamic Model (GPDM) [wa05]
- Other applications:
  - GP in Reinforcement Learning [ra04]
  - GP Model Based Predictive Control [ko04]

Propabilistic PCA: Short Overview

- Latent Variable Model:
  - Project high dimensional data (Y, d-dimensional) onto a low dimensional latent space (X, q-dimensional, q << d)
- Propabilistic PCA
  - Likelihood of a datapoint: $$p(\mathbf{y}_n|\mathbf{W},\beta) = \int p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{W},\beta)\, p(\mathbf{x}_n)\, d\mathbf{x}_n$$
    $$p(\mathbf{y}_n|\mathbf{x}_n,\mathbf{W},\beta) = N\left(\mathbf{y}_n|\mathbf{W}\mathbf{x}_n,\beta^{-1}\mathbf{I}\right)$$
  - Likelihood of the dataset: $$p(\mathbf{Y}|\mathbf{W},\beta) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mathbf{W},\beta)$$
  - Marginalize W:
    - Prior on W: $$p(\mathbf{W}) = \prod_{i=1}^{D} N\left(\mathbf{w}_i|0,\alpha^{-1}\mathbf{I}\right)$$
    - Marginalized likelihood of Y: $$p(\mathbf{Y}|\mathbf{X},\beta) = \frac{1}{(2\pi)^{\frac{DN}{2}}|\mathbf{K}|^{\frac{D}{2}}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right)\right)$$
      - » Where $\mathbf{K} = \alpha\mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}$ and $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_N]^{\mathrm{T}}$

PPCA: Short Overview

- Optimize X:
  - Log-likelihood: $$L = -\frac{DN}{2}\ln(2\pi) - \frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right)$$
  - Optimize X: $$\frac{\partial L}{\partial \mathbf{X}} = \alpha\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{X} - \alpha D\mathbf{K}^{-1}\mathbf{X}$$
  - Solution: $$\mathbf{X} = \mathbf{U}_q\mathbf{L}\mathbf{V}^{\mathrm{T}}$$

    U….N x q matrix of q eigenvectors of $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$

    L … diagonal matrix containing the eigen values of $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$

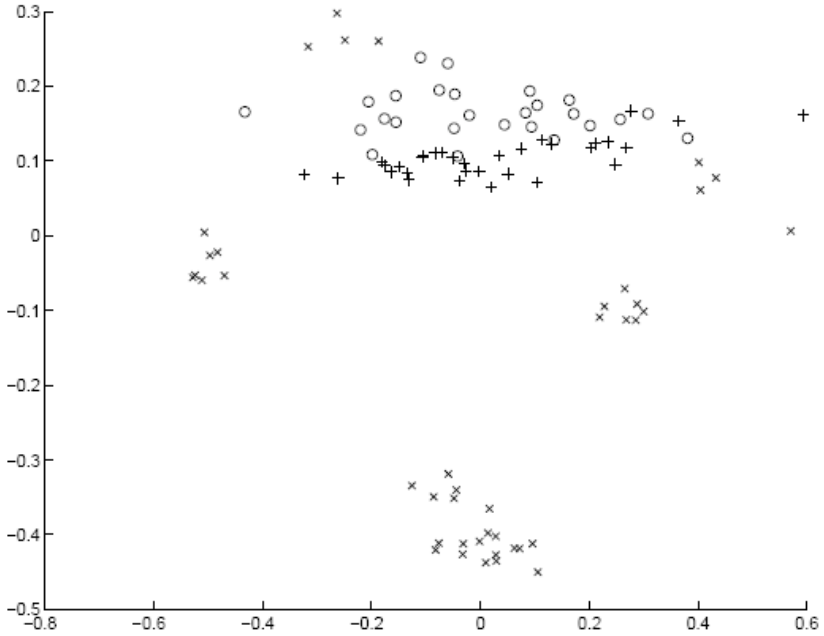
    V… arbitrary $q \times q$ orthogonal matrix
    - It can be shown that this solution is equivalent to that solved in PCA
- Kernel PCA: Replace $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ with a kernel

GPLVM

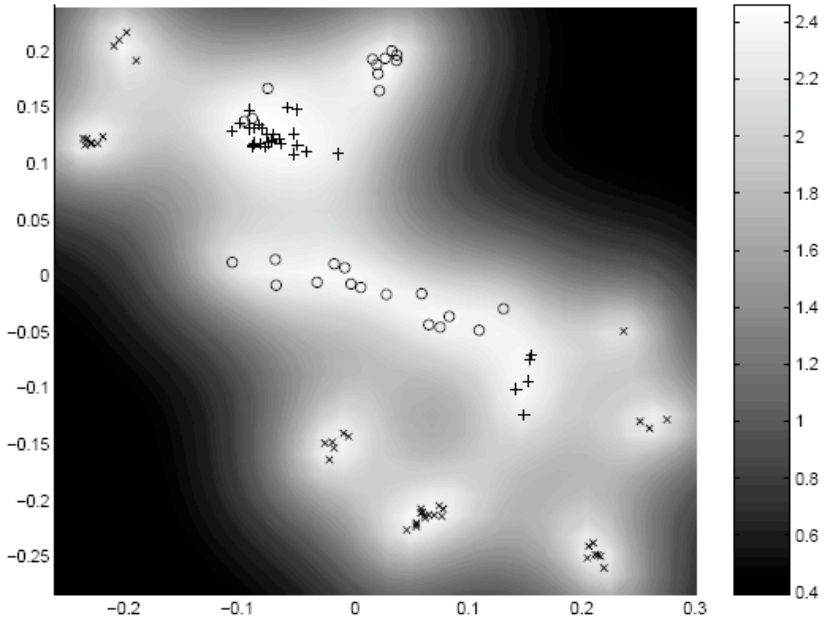
- PCA can be interpreted as GP mapping from X to Y with linearisation of the covariance matrix
- Non-linearisation of the mapping from the latent space to the data space
  - Non-linear covariance function
    - Use Standard RBF Kernel instead of
  - Calculate gradient of the log-likelihood with cha $\mathbf{K} = \alpha \mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}$
    -
    - $\frac{\partial L}{\partial \mathbf{K}} = \mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}^{-1} - D\mathbf{K}^{-1}$
    - $\frac{\partial \mathbf{K}}{\partial x_{n,j}}$
  - Optimise jointly X and hyperparamters of the kernel (e.g. with scaled conjugate gradients)
  - Initialize X with PCA
  - Each Gradient calculation requires inverse of the kernel matrix => $O(N^3)$

GPLVM: illustrative Result

- Oil data set
  - 3 classes coresponding to the phase flow in a pipeline: stratified, annular, homogenous
  - 12 input dimensions



PCA



GPLVM

Style based Inverse Kinematics

- Use GPLVM to represent Human motion data
  - Pose: 42-D vector q (joints, position, orientation)
  - Always use one specific motion style (e.g. walking)
  - Feature Vectors: y
    - Joint Angles
    - Vertical Orientation
    - Velocity and Acceleration for each feature
    - > 100 dimensions
  - Latent Space: usually 2-D or 3-D
- Scaled Version of GPLVM

$$p(\mathbf{Y} \mid \mathbf{X}, \bar{\beta}) = \frac{|\mathbf{W}|^N}{\sqrt{(2\pi)^{ND}|\mathbf{K}_Y|^D}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^T\right)\right)$$

  - Minimize negative log-posterior likelihood

$$\begin{aligned} L_{GP} &= -\ln p(\{\mathbf{x}_i\}, \alpha, \beta, \gamma, \{w_k\} | \{\mathbf{y}_i\}) \\ &= -\ln p(\{\mathbf{y}_i\} | \{\mathbf{x}_i\}, \alpha, \beta, \gamma, \{w_k\})(\prod_i p(\mathbf{x}_i))p(\alpha, \beta, \gamma) \end{aligned}$$

$$L_{GP} = \frac{D}{2}\ln|\mathbf{K}| + \frac{1}{2}\sum_k w_k^2 \mathbf{Y}_k^T \mathbf{K}^{-1}\mathbf{Y}_k + \frac{1}{2}\sum_i ||\mathbf{x}_i||^2 + \ln\frac{\alpha\beta\gamma}{\prod_k w_k^N}$$

Style-based Inverse Kinematics

- Generating new Poses (Predicting) :
  - We do not know the location in latent space
  - Negative Log Likelihood for a new pose (x,y)

    - Standard GP equations:

      - $L_{IK}(\mathbf{x},\mathbf{y}) = \frac{||\mathbf{W}(\mathbf{y}-\mathbf{f}(\mathbf{x}))||^2}{2\sigma^2(\mathbf{x})} + \frac{D}{2}\ln\sigma^2(\mathbf{x}) + \frac{1}{2}||\mathbf{x}||^2$ ction
        - » Certainty is greatest near the trainin
      - => keep y close to prediction f(x) while ke

$$\mathbf{f}(\mathbf{x}) = \mu + \overline{\mathbf{Y}}^T\overline{\mathbf{K}}^{-1}\mathbf{k}(\mathbf{x})$$
$$\sigma^2(\mathbf{x}) = k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x})^T\overline{\mathbf{K}}^{-1}\mathbf{k}(\mathbf{x})$$

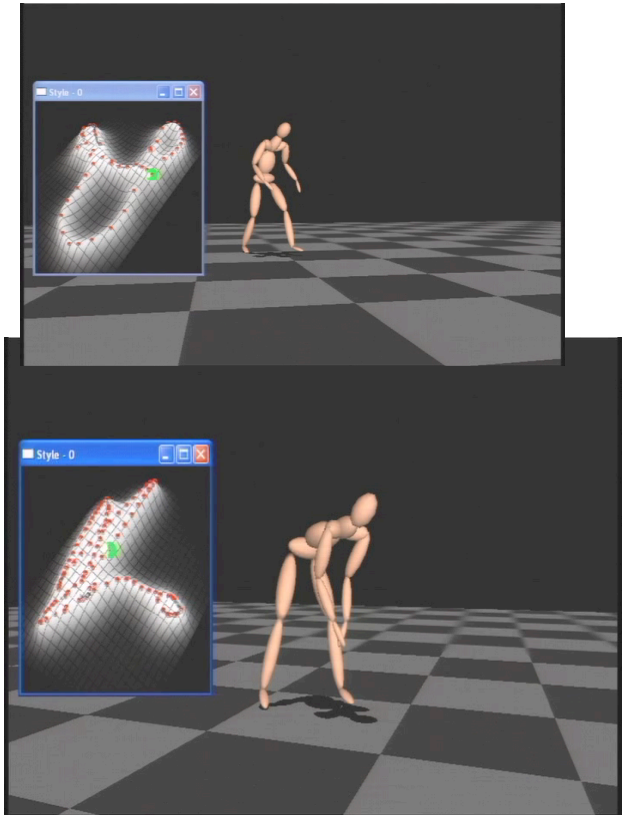
- Synthesis: Optimize q given some constraints C
  - Specified by the user, e.g. positions of the hands, feets

$\sigma^2(\mathbf{x})$

$$\arg\min_{\mathbf{x},\mathbf{q}} L_{IK}(\mathbf{x},\mathbf{y}(\mathbf{q})) \qquad \text{s.t. } C(\mathbf{q}) = 0$$

SBIK: Results

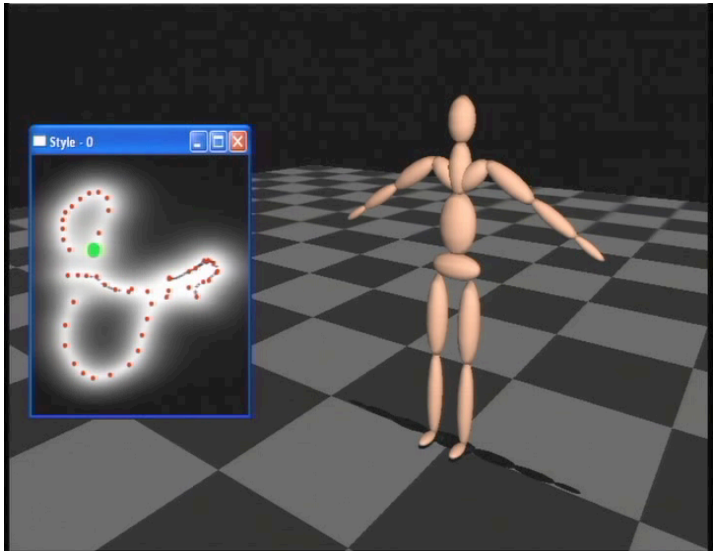
- Different Styles:
  - Base-Ball Pitch
  - Start running

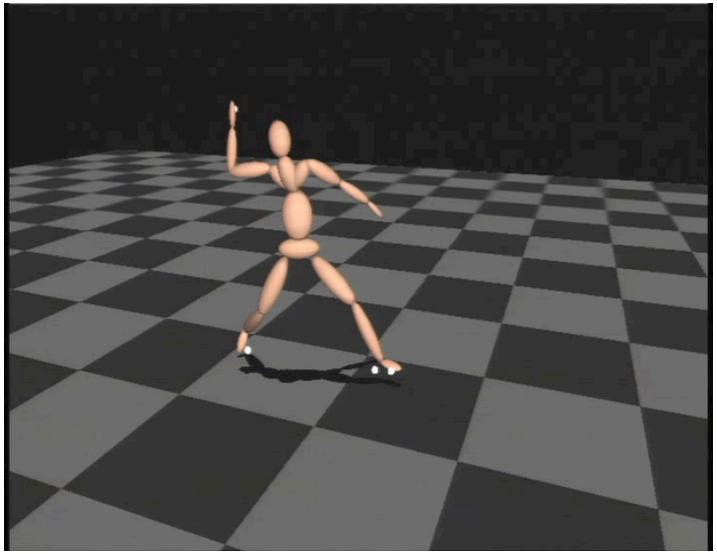


SBIK: Results

- Posing characters
  - Specify position in 2-D latent space

- Specify/Change trajectories





GP Dynamic Model [wa05]

- SBIK does not consider the dynamics of the poses (sequential order of the poses)
  - Model the dynamics in latent Space X
- 2 Mappings:
  - Dynamics in Low dimensional latent space X (q dimensional), markov property
  - Mapping from latent space to data space Y (d dimensional) (GPLVM)

  - Model both mappings with GP

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}; \mathbf{A}) + \mathbf{n}_{x,t} \\ \mathbf{y}_t &= g(\mathbf{x}_t; \mathbf{B}) + \mathbf{n}_{y,t} \end{aligned}$$

GPDM: Learn Mappings

- Fit parameters: Weights, number of basis functions + shape
  - difficult
- From GP view: parameters should be marginalized out

GPDM: Learn Mapping g

- Learning Mapping g:
  - Mapping from latent space X to high dimensional output space Y
- Prior on Y

  - Independent Gaussian for every output dimension
  - W = diag(w_1,..,w_D) … scaling matrix, to account for different variances in different data dimensions
- RBF Covaria

$$p(\mathbf{Y} \mid \mathbf{X}, \bar{\beta}) = \frac{|\mathbf{W}|^N}{\sqrt{(2\pi)^{ND}|\mathbf{K}_Y|^D}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^T\right)\right)$$

- Hyperparameters:

$$k_Y(\mathbf{x}, \mathbf{x}') = \beta_1 \exp\left(-\frac{\beta_2}{2}||\mathbf{x} - \mathbf{x}'||^2\right) + \beta_3^{-1}\delta_{\mathbf{x},\mathbf{x}'}$$

$$\bar{\beta} = \{\beta_1, \beta_2, ..., \mathbf{W}\}$$

GPDM: Learn dynamic Mapping f

- Mapping g:
  - Mapping from latent space X to high dimensional output space Y
  - Same as in Style based kinematics
- GP: marginalizing over weights A
  -
- Markov property
  - $$p(\mathbf{X} \,|\, \bar{\alpha}) \;=\; \int p(\mathbf{X}, \mathbf{A} \,|\, \bar{\alpha})\, d\mathbf{A} \;=\; \int p(\mathbf{X} \,|\, \mathbf{A}, \bar{\alpha})\, p(\mathbf{A} \,|\, \bar{\alpha})\, d\mathbf{A}$$
- Again multivariate GP: Posterior distribution on X

$$p(\mathbf{X} \,|\, \bar{\alpha}) \;=\; p(\mathbf{x}_1) \int \prod_{t=2}^{N} p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{A}, \bar{\alpha})\; p(\mathbf{A} \,|\, \bar{\alpha})\, d\mathbf{A}$$

$$p(\mathbf{X} \,|\, \bar{\alpha}) \quad = \quad p(\mathbf{x}_1) \frac{1}{\sqrt{(2\pi)^{(N-1)d} |\mathbf{K}_X|^d}} \exp\left(-\frac{1}{2} \mathrm{tr}\left(\mathbf{K}_X^{-1} \mathbf{X}_{out} \mathbf{X}_{out}^T \right) \right)$$

GPDM: Learn dynamic Mapping f

- Priors for X: $X_{out} = [x_2, \ldots, x_N]^T$

$$p(X|\alpha) = p(\mathbf{x}_1)\frac{1}{\sqrt{(2\pi)^{(N-1)d}|\mathbf{K}_X|^d}}\exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T\right)\right)$$

  -
    - Future x_{n+1} is target of the approximation
  - x1 is assumed to have Gaussian prior
  - K_X ...(N-1)x(N-1) kernel matrix

$$k_X(\mathbf{x}, \mathbf{x}') = \alpha_1 \exp\left(-\frac{\alpha_2}{2}||\mathbf{x} - \mathbf{x}'||^2\right) + \alpha_3\mathbf{x}^T\mathbf{x}' + \alpha_4^{-1}\delta_{\mathbf{x},\mathbf{x}'}$$

  - Joint distribution of the latent Variables is not Gaussian
    - x_t does occur outside the covariance matrix

GPDM: Algorithm

- Minimize negative log-posterior
  -

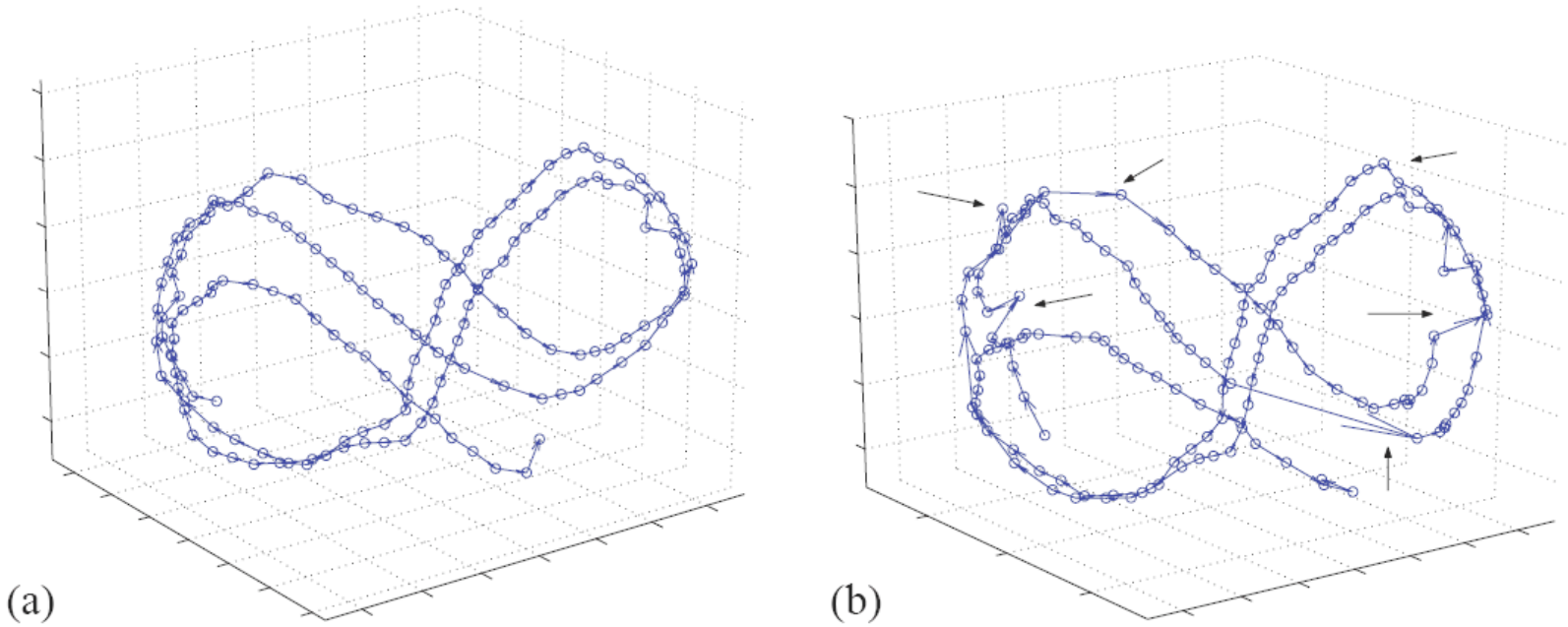
$$\begin{aligned}\mathcal{L} &= -\ln p(\mathbf{X}, \bar{\alpha}, \bar{\beta} \,|\, \mathbf{Y}) \\ &= -\ln p(\mathbf{Y}|\mathbf{X}, \bar{\beta})\, p(\mathbf{X}|\bar{\alpha})\, p(\bar{\alpha})\, p(\bar{\beta}) \\ &= \frac{d}{2}\ln|\mathbf{K}_X| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T\right) + \sum \ln \alpha_j \\ &\quad - N\ln|\mathbf{W}| + \frac{D}{2}\ln|\mathbf{K}_Y| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^T\right) + \sum_j \ln \beta_j\end{aligned}$$

  - Minimize wit
  - Data:
    - Mean-subtracted
    - X was initialized with PCA coordinates

$\mathbf{X}, \bar{\alpha}, \quad \bar{\beta}$

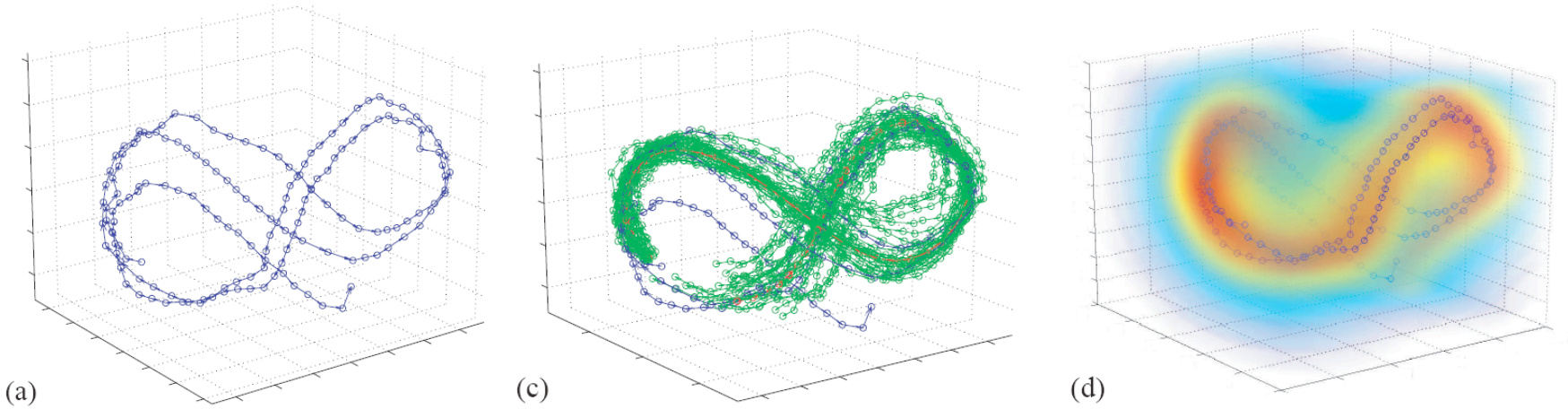
  - Numerical Minimization through Scaled Conjugate Gradient

GPDM: Results



- (b) Style-based Inverse Kinematics
- (a) GPDM
- Smoother trajectory in latent space!

GPDM: Visualization



- (c) 25 samples from the distribution
  - Sampled with the hybrid Monte Carlo Method $\tilde{\mathbf{X}}_{1:60}^{(j)} \sim p(\tilde{\mathbf{X}}_{1:60} \,|\, \mathbf{x}_0, \mathbf{X}, \mathbf{Y}, \bar{\alpha})$
- (d) Confidence with which the model reconstructs the pose from the latent position
  - High probability tube around the data

GPDM: Online generation of new motion

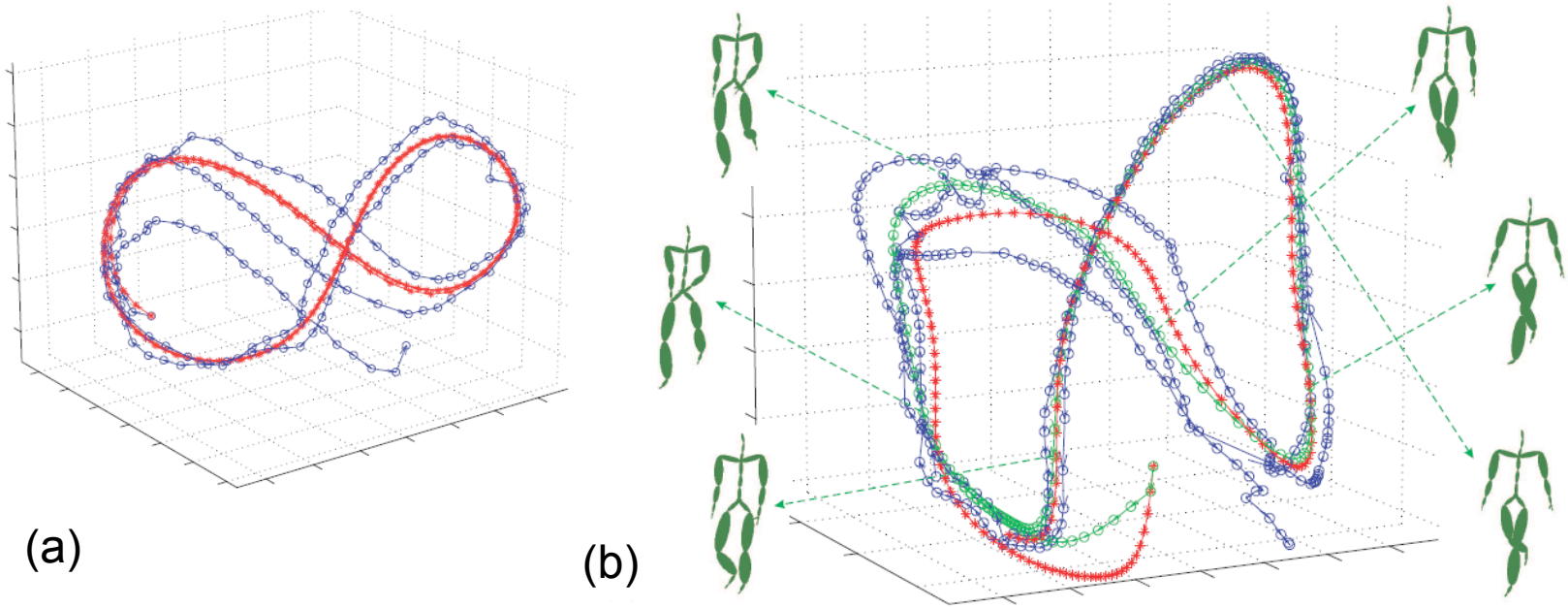
- Mean Prediction Sequences
  - Standard GP equations

$$\tilde{\mathbf{x}}_t \sim \mathcal{N}(\mu_X(\tilde{\mathbf{x}}_{t-1}); \sigma_X^2(\tilde{\mathbf{x}}_{t-1})\mathbf{I})$$

$$\mu_X(\mathbf{x}) = \mathbf{X}_{out}^T \mathbf{K}_X^{-1} \mathbf{k}_X(\mathbf{x}) , \qquad \sigma_X^2(\mathbf{x}) = k_X(\mathbf{x}, \mathbf{x}) - \mathbf{k}_X(\mathbf{x})^T \mathbf{K}_X^{-1} \mathbf{k}_X(\mathbf{x})$$

  - Always take the predicted mean point
  - The same for new poses ($\mathbf{y}_t = \mu_Y(\mathbf{x}_t)$
  - Long sequences generated by mean prediction can diverge from the data
- Optimization
  - Prevent the Mean Prediction from drifting away from training data
  - Optimize the likelihood $p(\tilde{\mathbf{X}} \mid \mathbf{X}, \bar{\alpha})$ ıe new sequence
  - is lower near the trainıng data, consequently the likelihood of x_{t+1} c$\tilde{\mathbf{X}}$ be increased by moving x_t closer to the training data
  - C$\sigma_X^2(\mathbf{x}_t)$n process is initialized with mean prediction sequence

GPDM: Mean Prediction and Optimization



- (a) Mean prediction
- (b) Optimization

Summary/Conclusion

- GPLVM
  - GPs are used to model high dimensional data in a low dimensional latent space
  - Extension of the linear PCA formulation
- Human Motion
  - Generalizes well from small datasets
  - Can be used to generate new motion sequences
  - Very flexible and naturally looking solutions
- GPDM
  - additionally learn the dynamics in latent space

Literature

- [ma05] D. MacKay, *Introduction to Gaussian Processes*, 2005
- [ra03] C. Rasmussen, *Gaussian Processes in Machine Learning*, 2003
- [wa05] J. Wang and A. Hertzmann, *Gaussian Process Dynamical Models*, 2005
- [la04] N. Lawrence, *Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data*, 2004
- [gr04] K. Grochow, Z. Popovic, *Style-Based Inverse Kinematics*, 2004
- [ra04] C. Rasmussen and M. Kuss, *Gaussian Processes in Reinforcement Learning*, 2004
- [ko04] J. Kocijan, C. Rasmussen and A. Girard, *Gaussian Process Model Based Predictive Control*, 2004
- [sh04] J. Shi, D. Titterington, *Hierarchical Gaussian process mixtures for regression*