Variational Gaussian Process Dynamical Systems

Anonymous Author(s)
Affiliation
Address
email

Abstract

High dimensional time series are endemic in applications of machine learning such as robotics (sensor data), computational biology (gene expression data), vision (video sequences) and graphics (motion capture data). Practical nonlinear probabilistic approaches to this data are required. In this paper we introduce the variational Gaussian process dynamical system. Our work builds on recent variational approximations for Gaussian process latent variable models to allow for nonlinear dimensionality reduction simultaneously with learning a dynamical prior in the latent space. The approach also allows for the appropriate dimensionality of the latent space to be automatically determined. We demonstrate the model on a human motion capture data set and a series of high resolution video sequences.

1 Introduction

Nonlinear probabilistic modeling of high dimensional time series data is a key challenge for the machine learning community. A standard approach is to simultaneously apply a nonlinear dimensionality reduction to the data whilst governing the latent space with a nonlinear temporal prior. The key difficulty for such approaches is that analytic marginalization of the latent space is typically intractable. Markov chain Monte Carlo approaches can also be problematic as latent trajectories are strongly correlated making efficient sampling a challenge. One promising approach to these time series has been to extend the Gaussian process latent variable model [1, 2] with a dynamical prior for the latent space and seek a maximum a posteriori (MAP) solution for the latent points [3, 4, 5]. Ko and Fox [6] further extend these models for fully Bayesian filtering in a robotics setting. We refer to this class of dynamical models based on the GP-LVM as Gaussian process dynamical systems (GPDS). However, the use of a MAP approximation for training these models presents key problems. Firstly, since the latent variables are not marginalised, the parameters of the dynamical prior cannot be optimized without the risk of overfitting. Further, the dimensionality of the latent space cannot be determined by the model: adding further dimensions always increases the likelihood of the data. In this paper we build on recent developments in variational approximations for Gaussian processes [7, 8] to introduce a variational Gaussian process dynamical system (VGPDS) where latent variables are approximately marginalized through optimization of a rigorous lower bound on the marginal likelihood. As well as providing a principled approach to handling uncertainty in the latent space, this allows both the parameters of the latent dynamical process and the dimensionality of the latent space to be determined. The approximation enables the application of our model to time series containing millions of dimensions and thousands of time points. We illustrate this by modeling human motion capture data and high dimensional video sequences.

2 The Model

Given a multivariate times series dataset $\{\mathbf{y}_n, t_n\}_{n=1}^N$, where $\mathbf{y}_n \in \mathbb{R}^D$ is a data vector observed at time $t_n \in \mathbb{R}_+$. We are interested in cases where each $\mathbf{y}_n$ is a high dimensional vector. We assume

that there exists a low dimensional manifold that governs the generation of the data. Specifically, a *temporal* latent function $\mathbf{x}(t) \in \mathbb{R}^Q$ (with $Q \ll D$), governs an intermediate *hidden* layer when generating the data, and the dth feature from the data vector $\mathbf{y}_n$ is then produced from $\mathbf{x}_n = \mathbf{x}(t_n)$ according to

$$y_{nd} = f_d(\mathbf{x}_n) + \epsilon_{nd} \ , \quad \epsilon_{nd} \sim \mathcal{N}(0, \beta^{-1}), \tag{1}$$

where $f_d(\mathbf{x})$ is a latent mapping from the low dimensional space to dth dimension of the observation space and β is the inverse variance of the white Gaussian noise. We do not want to make strong assumptions about the functional form of the latent functions $(\mathbf{x}, \mathbf{f})$.[1] Instead we would like to infer them in a fully Bayesian non-parametric fashion using Gaussian processes [9]. Therefore, we assume that $\mathbf{x}$ is a multivariate Gaussian process indexed by time t and $\mathbf{f}$ is a different multivariate Gaussian process indexed by $\mathbf{x}$, and we write

$$x_q(t) \sim \mathcal{GP}(0, k_x(t_i, t_j)), \ \ q = 1, \ldots, Q, \tag{2}$$

$$f_d(\mathbf{x}) \sim \mathcal{GP}(0, k_f(\mathbf{x}_i, \mathbf{x}_j)), \ \ d = 1, \ldots, D. \tag{3}$$

Here, the individual components of the latent function $\mathbf{x}$ are taken to be independent sample paths drawn from a Gaussian process with covariance function $k_x(t_i, t_j)$. Similarly, the components of $\mathbf{f}$ are independent draws from a Gaussian process with covariance function $k_f(\mathbf{x}_i, \mathbf{x}_j)$. These covariance functions, parametrized by parameters $\boldsymbol{\theta}_x$ and $\boldsymbol{\theta}_f$ respectively, play very distinct roles in the model. More precisely, k_x determines the properties of each temporal latent function $x_q(t)$. For instance, the use of an Ornstein-Uhlbeck covariance function yields a Gauss-Markov process for $x_q(t)$, while the squared-exponential kernel gives rise to very smooth and non-Markovian processes. In our experiments, we will focus on the squared exponential covariance function (RBF), the Matern $3/2$ which is only once differentiable, and a periodic covariance function [9, 10] which can be used when data exhibit strong periodicity. These kernel functions take the form:

$$k_{x(\text{rbf})}(t_i, t_j) = \sigma^2_{\text{rbf}} e^{-\frac{(t_i - t_j)^2}{(2l_t^2)}}, \ \ k_{x(\text{mat})}(t_i, t_j) = \sigma^2_{\text{mat}} \left(1 + \frac{\sqrt{3}|t_i - t_j|}{l_t}\right) e^{\frac{-\sqrt{3}|t_i - t_j|}{l_t}},$$

$$k_{x(\text{per})}(t_i, t_j) = \sigma^2_{\text{per}} e^{-\frac{1}{2}\frac{sin^2\left(\frac{2\pi}{T}(t_i - t_j)\right)}{l_t}}. \tag{4}$$

The covariance function k_f determines the properties of the latent mapping $\mathbf{f}$ that maps each low dimensional variable $\mathbf{x}_n$ to the observed vector $\mathbf{y}_n$. We wish this mapping to be a non-linear but smooth, and thus a suitable choice is the squared exponential covariance function

$$k_f(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2_{ard} e^{-\frac{1}{2}\sum_{q=1}^{Q} w_q (x_{i,q} - x_{j,q})^2}, \tag{5}$$

which assumes a different scale w_q for each latent dimension. This, as in the variational Bayesian formulation of the GP-LVM [8], enables an automatic relevance determination procedure (ARD), i.e. it allows Bayesian training to "switch off" unnecessary dimensions by driving the values of the corresponding scales to zero.

The matrix $Y \in \mathbb{R}^{N \times D}$ will collectively denote all observed data so that its nth row corresponds to the data point $\mathbf{y}_n$. Similarly, the matrix $F \in \mathbb{R}^{N \times D}$ will denote the mapping latent variables, i.e. $f_{nd} = f_d(\mathbf{x}_n)$, associated with observations Y from (1). Analogously, $X \in \mathbb{R}^{N \times Q}$ will store all low dimensional latent variables $x_{nq} = x_q(t_n)$. Further, we will refer to columns of these matrices by the vectors $\mathbf{y}_d, \mathbf{f}_d, \mathbf{x}_q \in \mathbb{R}^N$. Given the latent variables we assume independence over the data features, and given time we assume independence over latent dimensions to give

$$p(Y, F, X|\mathbf{t}) = p(Y|F)p(F|X)p(X|\mathbf{t}) = \prod_{d=1}^{D} p(\mathbf{y}_d|\mathbf{f}_d)p(\mathbf{f}_d|X) \prod_{q=1}^{Q} p(\mathbf{x}_q|\mathbf{t}), \tag{6}$$

where $\mathbf{t} \in \mathbb{R}^N$ and $p(\mathbf{y}_d|\mathbf{f}_d)$ is a Gaussian likelihood function term defined from (1). Further, $p(\mathbf{f}_d|X)$ is a marginal GP prior such that

$$p(\mathbf{f}_d|X) = \mathcal{N}(\mathbf{f}_d|\mathbf{0}, K_{NN}), \tag{7}$$

[1] To simplify our notation, we often write $\mathbf{x}$ instead of $\mathbf{x}(t)$ and $\mathbf{f}$ instead of $\mathbf{f}(\mathbf{x})$. Later we also use a similar convention for the kernel functions by often writing them as k_f and k_x.

where $K_{NN} = k_f(X, X)$ is the covariance matrix defined by the kernel function k_f and similarly $p(\mathbf{x}_q|\mathbf{t})$ is the marginal GP prior associated with the temporal function $x_q(t)$,

$$p(\mathbf{x}_q|\mathbf{t}) = \mathcal{N}(\mathbf{x}_q|\mathbf{0}, K_t), \tag{8}$$

where $K_t = k_x(\mathbf{t}, \mathbf{t})$ is the covariance matrix obtained by evaluating the kernel function k_x on the observed times $\mathbf{t}$.

Bayesian inference using the above model poses a huge computational challenge as, for instance, marginalization of the variables X, that appear non-linearly inside the kernel matrix K_{NN}, is troublesome. Practical approaches that have been considered until now (e.g. [5, 3]) marginalise out only F and seek a MAP solution for X. In the next section we describe how efficient variational approximations can be applied to marginalize X by extending the framework of [8].

2.1 Variational Bayesian training

The key difficulty with the Bayesian approach is propagating the prior density $p(X|\mathbf{t})$ through the nonlinear mapping. This mapping gives the expressive power to the model, but simultaneously renders the associated marginal likelihood,

$$p(Y|\mathbf{t}) = \int p(Y|F)p(F|X)p(X|\mathbf{t})\mathrm{d}X\,\mathrm{d}F, \tag{9}$$

intractable. We now invoke the variational Bayesian methodology to approximate the integral. Following a standard procedure [11], we introduce a variational distribution $q(\Theta)$ and compute the Jensen's lower bound $\mathcal{F}_v$ on the logarithm of (9),

$$\mathcal{F}_v(q, \boldsymbol{\theta}) = \int q(\Theta) \log \frac{p(Y|F)p(F|X)p(X|\mathbf{t})}{q(\Theta)} \mathrm{d}X\,\mathrm{d}F, \tag{10}$$

where $\boldsymbol{\theta}$ denotes the model's parameters. However, the above form of the lower bound is problematic becauce X (in the GP term $p(F|X)$) appears non-linearly inside the kernel matrix K_{NN} making the integration over X difficult. As shown in [8], this intractability is removed by applying the "data augmentation" principle. More precisely, we augment the joint probability model in (6) by including M extra samples of the GP latent mapping $\mathbf{f}$, known as inducing points, so that $\mathbf{u}_m \in \mathbb{R}^D$ is such a sample. The inducing points are evaluated at a set of pseudo-inputs $\tilde{X} \in \mathbb{R}^{M \times Q}$. The augmented joint probability density takes the form

$$p(Y, F, U, X, \tilde{X}|\mathbf{t}) = \prod_{d=1}^{D} p(\mathbf{y}_d|\mathbf{f}_d)p(\mathbf{f}_d|\mathbf{u}_d, X)p(\mathbf{u}_d|\tilde{X})p(X|\mathbf{t}), \tag{11}$$

where $p(\mathbf{u}_d|\tilde{X})$ is a zero-mean Gaussian with a covariance matrix K_{MM} constructed using the same function as for the GP prior (7). By dropping $\tilde{X}$ from our expressions, we write the augmented GP prior analytically (see [9]) as

$$p(\mathbf{f}_d|\mathbf{u}_d, X) = \mathcal{N}\left(\mathbf{f}_d|K_{NM}K_{MM}^{-1}\mathbf{u}_d, K_{NN} - K_{NM}K_{MM}^{-1}K_{MN}\right). \tag{12}$$

A key result in [8] is that a tractable lower bound (computed analogously to (10)) can be obtained through the variational density

$$q(\Theta) = q(F, U, X) = q(F|U, X)q(U)q(X) = \prod_{d=1}^{D} p(\mathbf{f}_d|\mathbf{u}_d, X)q(\mathbf{u}_d)q(X), \tag{13}$$

where $q(X) = \prod_{q=1}^{Q} \mathcal{N}(\mathbf{x}_q|\boldsymbol{\mu}_q, S_q)$ and $q(\mathbf{u}_d)$ is an arbitrary variational distribution. Titsias and Lawrence [8] assume full independence for $q(X)$ and the variational covariances are diagonal matrices. Here, in contrast, the posterior over the latent variables will have strong correlations, so S_q is taken to be a $N \times N$ full covariance matrix. Optimization of the variational lower bound provides an approximation to the true posterior $p(X|Y)$ by $q(X)$. In the augmented probability model, the "difficult" term $p(F|X)$ appearing in (10) is now replaced with (12) and, eventually, it cancels out with the first factor of the variational distribution (13) so that F can be marginalised out analytically.

Given the above and after breaking the logarithm in (10), we obtain the final form of the lower bound (see supplementary material for more details)

$$\mathcal{F}_v(q, \boldsymbol{\theta}) = \hat{\mathcal{F}}_v - \mathrm{KL}(q(X) \parallel p(X|\mathbf{t})), \tag{14}$$

with $\hat{\mathcal{F}}_v = \int q(X) \log p(Y|F)p(F|X)\,\mathrm{d}X\mathrm{d}F$. Both terms in (14) are now tractable. Note that the first of the above terms involves the data while the second one only involves the prior. All the information regarding data point correlations is captured in the KL term and the connection with the observations comes through the variational distribution. Therefore, the first term in (14) has the same analytical solution as the one derived in [8]. (14) can be maximized by using gradient-based methods[2]. However, when not factorizing $q(X)$ across data points yields $O(N^2)$ variational parameters to optimize. This issue is addressed in the next section.

2.2 Reparametrization and Optimization

The optimization involves the model parameters $\boldsymbol{\theta} = (\beta, \boldsymbol{\theta}_f, \boldsymbol{\theta}_t)$, the variational parameters $\{\boldsymbol{\mu}_q, S_q\}_{q=1}^{Q}$ from $q(X)$ and the inducing points[3] $\tilde{X}$.

Optimization of the variational parameters appears challenging, due to their large number and the correlations between them. However, by reparametrizing our $O\left(N^2\right)$ variational parameters according to the framework described in [12] we can obtain a set of $O(N)$ less correlated variational parameters. Specifically, we first take the derivatives of the variational bound (14) w.r.t. S_q and $\boldsymbol{\mu}_q$ and set them to zero, to find the stationary points,

$$S_q = \left(K_t^{-1} + \Lambda_q\right)^{-1} \quad \text{and} \quad \boldsymbol{\mu}_q = K_t \bar{\boldsymbol{\mu}}_q, \tag{15}$$

where $\Lambda_q = -2\frac{\vartheta \hat{F}_v(q,\boldsymbol{\theta})}{\vartheta S_q}$ is a $N \times N$ diagonal, positive matrix and $\bar{\boldsymbol{\mu}}_q = \frac{\vartheta \hat{F}_v}{\vartheta \boldsymbol{\mu}_q}$ is a $N-$dimensional vector. The above stationary conditions tell us that, since S_q depends on a diagonal matrix Λ_q, we can reparametrize it using only the $N-$dimensional diagonal of that matrix, denoted by $\boldsymbol{\lambda}_q$. Then, we can optimise the $2(Q \times N)$ parameters $(\boldsymbol{\lambda}_q, \bar{\boldsymbol{\mu}}_q)$ and obtain the original parameters using (15).

2.3 Learning from Multiple Sequences

Our objective is to model multivariate time series. A given data set may consist of a group of independent observed sequences, each with a different length (e.g. in human motion capture data several walks from a subject). Let, for example, the dataset be a group of S independent sequences $\left(Y^{(1)}, ..., Y^{(S)}\right)$. We would like our model to capture the underlying commonality of these data. We handle this by allowing a different temporal latent function for each of the independent sequences, so that $X^{(s)}$ is the set of latent variables corresponding to the sequence s. These sets are a priori assumed to be independent since they correspond to separate sequences, i.e. $p\left(X^{(1)}, X^{(2)}, ..., X^{(S)}\right) = \prod_{s=1}^{S} p(X^{(s)})$, where we dropped the conditioning on time for simplicity. This factorisation leads to a block-diagonal structure for the time covariance matrix K_t, where each block corresponds to one sequenece. In this setting, each block of observations $Y^{(s)}$ is generated from its corresponding $X^{(s)}$ according to $Y^{(s)} = F^{(s)} + \boldsymbol{\epsilon}$, where the latent function which governs this mapping is shared across all sequences and $\boldsymbol{\epsilon}$ is Gaussian noise.

3 Predictions

Our algorithm models the temporal evolution of a dynamical system. It should be capable of generating completely new sequences or reconstructing missing observations from partially observed data. For generating novel sequence given training data the model requires a time vector $\mathbf{t}_*$ as input and computes a density $p(Y_*|Y, \mathbf{t}, \mathbf{t}_*)$. For reconstruction of partially observed data the time stamp information is additionally accompanied by a partially observed sequence $Y_*^p \in \mathbb{R}^{N_* \times D_p}$ from the whole $Y_* = (Y_*^p, Y_*^m)$, where p and m are set of indices indicating the present (i.e. observed) and

[2] See supplementary material for more detailed derivation of (14) and for the equations for the gradients.

[3] We will use the term "variational parameters" to refer only to the parameters of $q(X)$ although the inducing points are also variational parameters.

missing dimensions of Y_* respectively, so that $p \cup m = \{1, \ldots, D\}$. We reconstruct the missing dimensions by computing the Bayesian predictive distribution $p(Y_*^m | Y_*^p, Y, \mathbf{t}_*, \mathbf{t})$. The predictive densities can also be used as estimators for tasks like generative Bayesian classification. Whilst time stamp information is always provided, in the next section we drop its dependence to avoid notational clutter.

3.1 Predictions Given Only the Test Time Points

To approximate the predictive density, we will need to introduce the underlying latent function values $F_* \in \mathbb{R}^{N_* \times D}$ (the noisy-free version of Y_*) and the latent variables $X_* \in \mathbb{R}^{N_* \times Q}$. We write the predictive density as

$$p(Y_*|Y) = \int p(Y_*, F_*, X_*|Y_*, Y) \mathrm{d}F_* \mathrm{d}X_* = \int p(Y_*|F_*) p(F_*|X_*, Y) p(X_*|Y) \mathrm{d}F_* \mathrm{d}X_*. \quad (16)$$

The term $p(F_*|X_*, Y)$ is approximated by the variational distribution

$$q(F_*|X_*) = \int \prod_{d \in D} p(\mathbf{f}_{*,d}|\mathbf{u}_d, X_*) q(\mathbf{u}_d) d\mathbf{u}_d = \prod_{d \in D} q(\mathbf{f}_{*,d}|X_*), \quad (17)$$

where $q(\mathbf{f}_{*,d}|X_*)$ is a Gaussian that can be computed analytically, since in our variational framework the optimal setting for $q(\mathbf{u}_d)$ is also found to be a Gaussian (see suppl. material for complete forms). As for the term $p(X_*|Y)$ in eq. (16), it is approximated by a Gaussian variational distribution $q(X_*)$,

$$q(X_*) = \prod_{q=1}^{Q} q(\mathbf{x}_{*,q}) = \prod_{q=1}^{Q} \int p(\mathbf{x}_{*,q}|\mathbf{x}_q) q(\mathbf{x}_q) \mathrm{d}\mathbf{x}_q = \prod_{q=1}^{Q} \left\langle p(\mathbf{x}_{*,q}|\mathbf{x}_q) \right\rangle_{q(\mathbf{x}_q)}, \quad (18)$$

where $p(\mathbf{x}_{*,q}|\mathbf{x}_q)$ is a Gaussian found from the conditional GP prior (see [9]) and $q(X)$ is also Gaussian. We can, thus, work out analytically the mean and variance for (18), which turn out to be:

$$\mu_{x_{*,q}} = K_{*N} \bar{\mu}_q \quad (19)$$

$$\mathrm{var}(x_{*,q}) = K_{**} - K_{*N}(K_t + \Lambda_q^{-1})^{-1} K_{N*}. \quad (20)$$

where $K_{*N} = k_x(\mathbf{t}_*, \mathbf{t})$, $K_{*N} = K_{*N}^\top$ and $K_{**} = k_x(\mathbf{t}_*, \mathbf{t}_*)$. Notice that these equations have exactly the same form as found in standard GP regression problems. Once we have analytic forms for the posteriors in (16), the predictive density is approximated as

$$p(Y_*|Y) = \int p(Y_*|F_*) q(F_*|X_*) q(X_*) dF_* dX_* = \int p(Y_*|F_*) \left\langle q(F_*|X_*) \right\rangle_{q(X_*)} dF_*, \quad (21)$$

which is a non-Gaussian integral that cannot be computed analytically. However, following the same argument as in [9, 13], we can calculate analytically its mean and covariance:

$$\mathbb{E}(F_*) = B^\top \Psi_1^* \quad (22)$$

$$\mathrm{Cov}(F_*) = B^\top \left(\Psi_2^* - \Psi_1^* (\Psi_1^*)^\top \right) B + \Psi_0^* I - \mathrm{Tr}\left[\left(K_{MM}^{-1} - (K_{MM} + \beta \Psi_2)^{-1} \right) \Psi_2^* \right] I, \quad (23)$$

where $B = \beta \left(K_{MM} + \beta \Psi_2\right)^{-1} \Psi_1^\top Y$, $\Psi_0^* = \langle k_f(X_*, X_*) \rangle$, $\Psi_1^* = \langle K_{M*} \rangle$ and $\Psi_2^* = \langle K_{M*} K_{*M} \rangle$. All expectations are taken w.r.t. $q(X_*)$ and can be calculated analytically, while K_{M*} denotes the cross-covariance matrix between the training inducing inputs $\tilde{X}$ and X_*. The Ψ quantities are calculated analytically (see suppl. material). Finally, since Y_* is just a noisy version of F_*, the mean and covariance of (21) is just computed as: $\mathbb{E}(Y_*) = \mathbb{E}(F_*)$ and $\mathrm{Cov}(Y_*) = \mathrm{Cov}(F_*) + \beta^{-1} I_{N_*}$.

3.2 Predictions Given the Test Time Points and Partially Observed Outputs

The expression for the predictive density $p(Y_*^m | Y_*^p, Y)$ is similar to (16),

$$p(Y_*^m | Y_*^p, Y) = \int p(Y_*^m | F_*^m) p(F_*^m | X_*, Y_*^p, Y) p(X_* | Y_*^p, Y) \mathrm{d}F_*^m \mathrm{d}X_*, \quad (24)$$

and is analytically intractable. To obtain an approximation, we firstly need to apply variational inference and approximate $p(X_*|Y_*^p, Y)$ with a Gaussian distribution. This requires the optimisation of a new variational lower bound that accounts for the contribution of the partially observed data Y_*^p. This lower bound approximates the true marginal likelihood $p(Y_*^p, Y)$ and has exactly analogous form with the lower bound computed only on the training data Y. Moreover, the variational optimisation requires the definition of the variational distribution $q(X_*, X)$ which needs to be optimised and is fully correlated across X and X_*. After the optimisation, the approximation to the true posterior $p(X_*|Y_*^p, Y)$ is given from the marginal $q(X_*)$. A much faster but less accurate method would be to decouple the test from the training latent variables by imposing the factorisation $q(X_*, X) = q(X)q(X_*)$. This is not used, however, in our current implementation.

4 Handling Very High Dimensional Datasets

Our variational framework avoids the typical cubic complexity of Gaussian processes allowing relatively large training sets (thousands of time points, N). Further, the model scales only linearly with the number of dimensions D. Specifically, the number of dimensions only matters when performing calculations involving the data matrix Y. In the final form of the lower bound (and consequently in all of the derived quantities, such as gradients) this matrix only appears in the form $YY^\top$ which can be precomputed. This means that, when $N \ll D$, we can calculate $YY^\top$ only once and then substitute Y with the SVD (or Cholesky decomposition) of $YY^\top$. In this way, we can work with an $N \times N$ instead of an $N \times D$ matrix. Practically speaking, this allows us to work with data sets involving millions of features. In our experiments we model directly the pixels of HD quality video, exploiting this trick.

5 Experiments

We consider two different types of high dimensional time series, a human motion capture data set consisting of different walks and high resolution video sequences. The experiments are intended to explore the various properties of the model and to evaluate its performance in different tasks (prediction, reconstruction, generation of data).

5.1 Human Motion Capture Data

We followed [14, 15] in considering motion capture data of walks and runs taken from subject 35 in the CMU motion capture database. We treated each motion as an independent sequence. The data set was constructed and preprocessed as described in [15]. This results in 2,613 separate 59-dimensional frames split into 31 training sequences with an average length of 84 frames each.

The model is jointly trained, as explained in section 2.3, on both walks and runs, i.e. the algorithm learns a common latent space for these motions. At test time we investigate the ability of the model to reconstruct test data from a previously unseen sequence given partial information for the test targets. This is tested once by providing only the dimensions which correspond to the body of the subject and once by providing those that correspond to the legs. We compare with results in [15], which used MAP approximations for the dynamical models, and against nearest neighbour. We can also indirectly compare with the binary latent variable model (BLV) of [14] which used a slightly different data preprocessing. We assess the performance using the cumulative error per joint in the scaled space defined in [14] and by the root mean square error in the angle space suggested by [15]. Our model was initialized with nine latent dimensions. We performed two runs, once using the Matern covariance function for the dynamical prior and once using the RBF. From table 1 we see that the variational Gaussian process dynamical system considerably outperforms the other approaches. The appropriate latent space dimensionality for the data was automatically inferred by our models. If an RBF covariance governed the dynamics the model retained four dimensions whereas the model using the Matern kept only three. The other latent dimensions were completely switched off by the ARD parameters. The best performance for the legs and the body reconstruction was achieved by the VGPDS model that used the Matern and the RBF covariance function respectively.

Table 1: Errors obtained for the motion capture dataset considering nearest neighbour in the angle space (NN) and in the scaled space(NN sc.), GPLVM, BLV and VGPDS. CL / CB are the leg and body datasets as preprocessed in [14], L and B the corresponding datasets from [15]. SC corresponds to the error in the scaled space, as in Taylor et al. while RA is the error in the angle space. The best error per column is in bold.

Data	CL	CB	L	L	B	B
Error Type	SC	SC	SC	RA	SC	RA
BLV	11.7	**8.8**	-	-	-	-
NN sc.	22.2	**20.5**	-	-	-	-
GPLVM (Q = 3)	-	-	11.4	3.40	16.9	2.49
GPLVM (Q = 4)	-	-	9.7	3.38	20.7	2.72
GPLVM (Q = 5)	-	-	13.4	4.25	23.4	2.78
NN sc.	-	-	13.5	4.44	20.8	2.62
NN	-	-	14.0	4.11	30.9	3.20
VGPDS (RBF)	-	-	8.19	3.57	**10.73**	**1.90**
VGPDS (Matern 3/2)	-	-	**6.99**	**2.88**	14.22	2.23

5.2 Modeling Raw High Dimensional Video Sequences

For our second set of experiments we considered video sequences. Such sequences are typically preprocessed before modeling to extract informative features and reduce the dimensionality of the problem. Here we work directly with the raw pixel values to demonstrate the ability of the VGPDS to model data with a vast number of features. This also allows us to directly sample video from the learned model.

Firstly, we used the model to reconstruct partially observed frames from test video sequences[4]. For the first video discussed here we gave as partial information approximately 50% of the pixels while for the other two we gave approximately 40% of the pixels on each frame. The mean squared error per pixel was measured to compare with the $k-$nearest neighbour (NN) method, for $k \in (1, .., 5)$ (we only present the error achieved for the best choice of k in each case). The datasets considered are the following: firstly, the 'Missa' dataset, a standard benchmark used in image processing. This is 103,680-dimensional video, showing a woman talking for 150 frames. The data is challenging as there are translations in the pixel space. We also considered an HD video of dimensionality 9×10^5 that shows an artificially created scene of ocean waves as well as a $230,400-$dimensional video showing a dog running for 60 frames. The later is approximately periodic in nature, containing several paces from the dog. For the first two videos we used the Matern and RBF kernel respectively to model the dynamics and interpolated to reconstruct blocks of frames chosen from the whole sequence. For the 'dog' dataset we constructed a compound kernel $k_x = k_{x(\text{rbf})} + k_{x(\text{periodic})}$, where the RBF term is employed to capture any divergence from the approximately periodic pattern. We then used our model to reconstruct the last 7 frames extrapolating beyond the original video. As can be seen in table 2, our method outperformed NN in all cases. The results are also demonstrated visually in figure 1 and the reconstructed videos are available in the supplementary material.

Table 2: The mean squared error per pixel for VGPDS and NN for the three datasets (measured only in the missing inputs). The number of latent dimensions selected by our model is in parenthesis.

	Missa	Ocean	Dog
VGPDS	2.52 ($Q = 12$)	9.36 ($Q = 9$)	4.01 ($Q = 6$)
NN	2.63	9.53	4.15

As can be seen in figure 1, VGPDM predicts pixels which are smoothly connected with the observed image, whereas the NN method cannot fit the predicted pixels in the overall context.

As a second task, we used our generative model to create new samples and generate a new video sequence. This is most effective for the 'dog' video as the training examples were approximately periodic in nature. The model was trained on 60 frames (time-stamps $[t_1, t_{60}]$) and we generated

[4] 'Missa' dataset: cipr.rpi.edu. 'Ocean': cogfilms.com. 'Dog': fitfurlife.com. See details in supplementary.

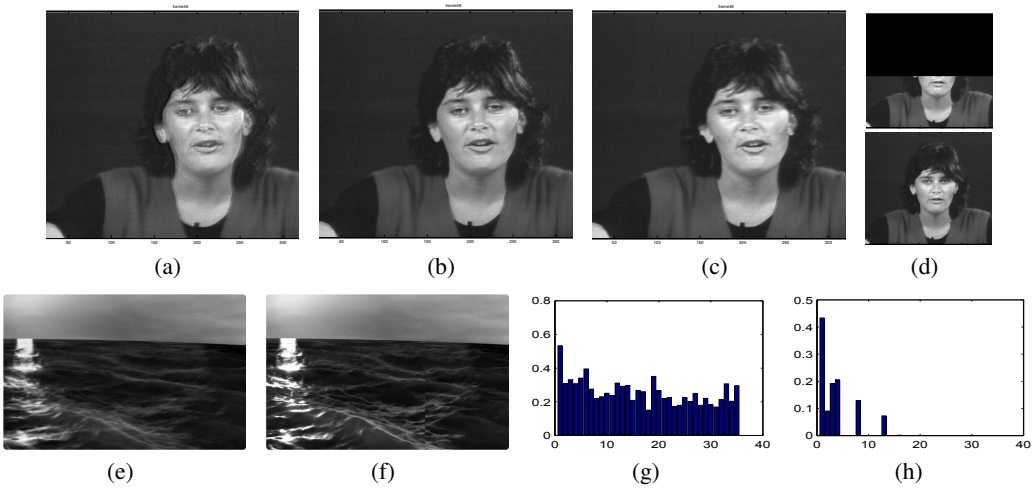

Figure 1: (a) and (c) demonstrate the reconstruction achieved by VGPDS and NN respectively for the most challenging frame (b) of the 'missa' video, i.e. when translation occurs. (d) shows another example of the reconstruction achieved by VGPDM given the partially observed image. (e) (VGPDS) and (f) (NN) depict the reconstruction achieved for a frame of the 'ocean' dataset. Finally, we demonstrate the ability of the model to automatically select the latent dimensionality by showing the initial lengthscales (fig: (g)) of the ARD kernel and the values obtained after training (fig: (h)) on the 'dog' data set.

the new frames which correspond to the next 40 time points in the future. The only input given for this generation of future frames was the time stamp vector, $[t_{61}, t_{100}]$. The results show a smooth transition from training to test and amongst the test video frames. The resulting video of the dog continuing to run is sharp and high quality. This experiment demonstrates the ability of the model to reconstruct massively high dimensional images without blurring. Frames from the result are shown in figure 2. The full video is available in the supplementary material.

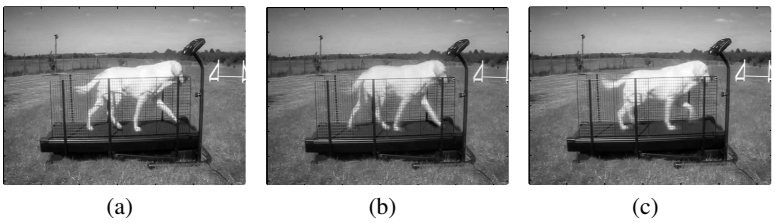

Figure 2: The last frame of the training video (a) is smoothly followed by the first frame (b) of the generated video. A subsequent generated frame can be seen in (c).

6 Discussion and Future Work

We have introduced a fully Bayesian approach for modeling dynamical systems through probabilistic nonlinear dimensionality reduction. Marginalizing the latent space and reconstructing data using Gaussian processes results in a very generic model for capturing complex, non-linear correlations even in very high dimensional data, without having to perform any data preprocessing or exhaustive search for defining the model's structure and parameters.

Our method's effectiveness has been demonstrated in two tasks; firstly, in modeling human motion capture data and, secondly, in reconstructing and generating raw, very high dimensional video sequences. A promising future direction to follow would be to enhance our formulation with domain-specific knowledge encoded, for example, in more sophisticated covariance functions or in the way that data are being preprocessed. Thus, we can obtain application-oriented methods to be used for tasks in areas such as robotics, computer vision and finance.

References

[1] N. D. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *Journal of Machine Learning Research*, vol. 6, pp. 1783–1816, 2005.

[2] N. D. Lawrence, "Gaussian process latent variable models for visualisation of high dimensional data," in *In NIPS*, p. 2004, 2004.

[3] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models," in *In NIPS*, pp. 1441–1448, MIT Press, 2006.

[4] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 283–298, Feb. 2008.

[5] N. D. Lawrence, "Hierarchical Gaussian process latent variable models," in *In International Conference in Machine Learning*, 2007.

[6] J. Ko and D. Fox, "GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models," *Auton. Robots*, vol. 27, pp. 75–90, July 2009.

[7] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," *JMLR W&CP*, vol. 5, pp. 567–574, 2009.

[8] M. Titsias and N. D. Lawrence, "Bayesian Gaussian process latent variable model," *Journal of Machine Learning Research - Proceedings Track*, vol. 9, pp. 844–851, 2010.

[9] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[10] D. J. C. MacKay, "Introduction to Gaussian processes," in *Neural Networks and Machine Learning* (C. M. Bishop, ed.), NATO ASI Series, pp. 133–166, Kluwer Academic Press, 1998.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st ed. 2006. corr. 2nd printing ed., Oct. 2007.

[12] M. Opper and C. Archambeau, "The variational Gaussian approximation revisited," *Neural Computation*, vol. 21, no. 3, pp. 786–792, 2009.

[13] A. Girard, C. E. Rasmussen, J. Quiñonero-Candela, and R. Murray-Smith, "Gaussian process priors with uncertain inputs - application to multiple-step ahead time series forecasting," in *Neural Information Processing Systems*, 2003.

[14] G. W. Taylor, G. E. Hinton, and S. Roweis, "Modeling human motion using binary latent variables," in *Advances in Neural Information Processing Systems*, p. 2007, MIT Press, 2006.

[15] N. D. Lawrence, "Learning for larger datasets with the Gaussian process latent variable model," *Journal of Machine Learning Research - Proceedings Track*, vol. 2, pp. 243–250, 2007.