Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

Multi-task and Multi-kernel Gaussian Process Dynamical Systems

Dimitrios Korkinof[a,*], Yiannis Demiris[a]

[a] *Department of Electrical & Electronic Engineering, Imperial College London, UK*

Abstract

In this work, we propose a novel method for rectifying damaged motion sequences in an unsupervised manner. In order to achieve maximal accuracy, the proposed model takes advantage of three key properties of the data: their sequential nature, the redundancy that manifests itself among repetitions of the same task, and the potential of knowledge transfer across different tasks. In order to do so, we formulate a factor model consisting of Gaussian Process Dynamical Systems (GPDS), where each factor corresponds to a single basic pattern in time and is able to represent their sequential nature. Factors collectively form a dictionary of fundamental trajectories shared among all sequences, thus able to capture recurrent patterns within the same or across different tasks. We employ variational inference to learn directly from incomplete sequences and perform maximum a-posteriori (MAP) estimates of the missing values. We have evaluated our model with a number of motion datasets, including robotic and human motion capture data. We have compared our approach to well-established methods in the literature in terms of their reconstruction error and our results indicate significant accuracy improvement across different datasets and missing data ratios. Concluding, we investigate the performance benefits of

[*] Corresponding author.
Email addresses: `d.korkinof10@alumni.impetial.ac.uk` (Dimitrios Korkinof), `y.demiris@impetial.ac.uk` (Yiannis Demiris)

the multi-task learning scenario and how this improvement relates to the extent of component sharing that takes place.

Keywords: Gaussian Processes, Variational Bayes, Matrix Decomposition, Factor Models, Data Completion, Human Motion, Gaussian Process Latent Variable Models, Multi-task learning, Unsupervised learning

1. Introduction

Although sensor equipment is becoming increasingly precise and reliable nowadays, it is sometimes impossible to avoid the manifestation of missing data. Even when using highly expensive motion capture systems, the recorded data may suffer from severe corruption and/or may require manual rectification. The problem is intensified in case we employ low-cost equipment, such as the popular Kinect or LEAP sensors.

The technological advances of the past few years have greatly facilitated the collection of large amounts of human motion data from a variety of sources and sensors (i.e. [1]). Despite the significant efforts towards modelling human behaviour [2, 3] and action recognition [4, 5, 6], these efforts are sometimes significantly impaired by inevitable data corruption either too costly or even impossible to repair.

The main approaches employed in the literature for the purpose of data completion are based on either supervised or unsupervised learning. In supervised methods, learning is performed on a set of complete training sequences and rectification of incomplete sequences is achieved by means of the posterior predictive distribution of the trained model. A main disadvantage of this approach is that it may be difficult to obtain a sufficiently large and diverse set of undamaged training sequences. There are several examples of supervised data completion in the literature employing a variety of models, such as Gaussian

Processes (GPs), Hidden Markov Models (HMMs), Linear Dynamical Systems (LDS) and many more.

On the other hand, unsupervised approaches can be trained directly on the incomplete data and thus are a rather more realistic scenario for practical applications. In this case missing data prediction may also be achieved by means of the model predictive distribution. However, for many models this distribution is intractable and we resort to approximations such as variational Bayes (VB) [7], sampling methods [8] or maximum a-posteriori (MAP) [9]. The latter, although being a rough approximation, is easily applicable in a wider range of models than VB, is less computationally intensive than sampling and often provides a good enough estimate. Relevant methods are widely adopted in image processing, such as inpainting[1] [10].

In unsupervised modelling of non-sequential datasets, missing data completion has previously been approached by means of matrix decomposition [11, 12, 13]. A variety of algorithms have been proposed over the years, the great majority of which focuses on point-estimate optimisation imposing various types of regularisation constraints for more effective learning [14, 15] and with many successful applications [16, 17]. Statistical approaches, on the other hand, enjoy a number of comparative merits stemming from the fact that inference, rather than optimisation, is employed. As a result, such methods are less prone to overfitting and have also been reported to achieve superior performance [18].

Modern statistical methods usually focus on image processing [19, 20] or collaborative filtering [21, 22], but few have been formulated to explicitly model temporal data. Furthermore, although unsupervised missing data completion is common in image inpainting, the concept has not been extensively explored

[1]Interpolation of the pixels removed from random locations in the image.

for sequential data, where learning is traditionally performed in a supervised manner.

A model widely adopted for the purpose of data completion is the Gaussian process (GP) [23]. This is mainly due to the fact that GPs are powerful non-parametric models, allowing us to impose high level data characteristics, such as smoothness, trend or periodicity, by selecting the appropriate functional form of the kernel matrix. Alternatively to using a single GP, some researchers have explored linear superpositions of GPs [24], which is shown to increase the model's descriptive power, with further relevant examples found in [25] and [26].

Conventional GPs are discriminative models, with their generative counterpart being the Gaussian process latent variable model (GPLVM), which accounts for the case when no observed covariates are available. The model assumes a lower-dimensional latent input covariate space, giving rise to a powerful non-linear factor model. GPLVMs have been applied with considerable success in data visualisation [27], dimensionality reduction [28], classification [29] and time-series modelling [30]. Their efficacy has also been previously shown in data completion applications [19].

The underlying concept of the GPLVM has initiated vigorous research in the field of Gaussian processes, giving rise to a variety of models able to capture data dynamics in motion and financial time-series [31, 32]. Recently a Bayesian formulation of the GPLVM has been devised in [33] by using inducing variables in order to ameliorate some of the intractabilities during inference, a concept which we extensively utilise in this work.

Further extending the notion of latent covariates towards modelling temporal data, Damianou et. al. [34] introduced a hierarchical model with an intermediate latent layer and a kernel governed by time, dubbed the Gaussian process dynamical system (GPDS) and later developed the deep Gaussian process [35]

Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

by introducing multiple such intermediate latent layers.

In this work, we propose a multi-task, multi-kernel model consisting of Gaussian process dynamical system factors that we shall dub the multi-GPDS. We form a dictionary of non-linear temporal factors shared among multiple tasks in order to facilitate information transfer across tasks which significantly improves reconstruction accuracy. As highlighted in the experimental section, we have observed considerable performance improvement, both compared to popular methods in the literature and compared to training the same model in a single-task fashion.

1.1. Theoretical Background

1.1.1. Gaussian Processes

A Gaussian process is formally defined as a possibly infinite collection of random variables $[f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), ...]$, any finite subset of which follows a joint Gaussian distribution [36].

More specifically, the training set $\{\boldsymbol{y}, \boldsymbol{X}\}$ consists of an observation vector $\boldsymbol{y} = [y_t]_{t=1}^T$ and a covariate matrix $\boldsymbol{X} = [\boldsymbol{x}_t]_{t=1}^T$, where, in general, $\boldsymbol{x}_t$ lies in a Q-dimensional space, $\boldsymbol{x}_t \in \mathbb{R}^Q$. The observations are modeled as follows:

$$y_t \sim \mathcal{N}\left(f(\boldsymbol{x}_t), \sigma^2\right) \tag{1}$$

$$\boldsymbol{f} \sim \mathcal{N}\left(0, \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})\right) \tag{2}$$

where $\boldsymbol{f} = [f_t]_{t=1}^T$ and each random variable $f_t = f(\boldsymbol{x}_t)$ represents the value of the function given input $\boldsymbol{x}_t$.

In that perspective, a Gaussian process can be viewed as a prior distribution over the set of all possible functions $f(\boldsymbol{x})$, with some characteristics dictated by the functional form of the kernel matrix $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})$, which for brevity we will denote as $\boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}}$.

For a set of test covariates $\boldsymbol{X}_*$ (previously unseen inputs), direct inference is tractable and the predictions vector $\boldsymbol{f}_*$ can be found as follows:

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} + \sigma^2 \boldsymbol{I} & \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}^*} \\ \boldsymbol{K}_{\boldsymbol{X}^*\boldsymbol{X}} & \boldsymbol{K}_{\boldsymbol{X}^*\boldsymbol{X}^*} \end{bmatrix}\right) \tag{3}$$

$$\mathbb{E}\left[\boldsymbol{f}_* | \boldsymbol{\mathcal{D}}, \boldsymbol{X}_*\right] = \boldsymbol{K}_{\boldsymbol{X}^*\boldsymbol{X}} \left[\boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} + \sigma^2 \boldsymbol{I}\right]^{-1} \boldsymbol{y} \tag{4}$$

$$\mathbb{V}\left[\boldsymbol{f}_* | \boldsymbol{\mathcal{D}}, \boldsymbol{X}_*\right] = \boldsymbol{K}_{\boldsymbol{X}^*\boldsymbol{X}^*} - \boldsymbol{K}_{\boldsymbol{X}^*\boldsymbol{X}} \left[\boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} + \sigma^2 \boldsymbol{I}\right]^{-1} \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}^*} \tag{5}$$

Being able to perform direct inference for this model is very convenient, although it does not scale well with the size of the training set as it is dominated by the inversion of an $T \times T$ matrix with complexity $\mathcal{O}\left(T^3\right)$.

In order to model multi-dimensional data, we could either impose a common or independent GP priors per dimension, with the latter being preferable to the former [23].

1.1.2. Gaussian Process Latent Variable Models

Let us now suppose a multi-dimensional set of observations $\boldsymbol{Y} \in \mathbb{R}^{T \times D}$ and no known corresponding covariates $\boldsymbol{X}$. We may still assume that the observed data can be accurately represented by a latent covariate space $\boldsymbol{X} \in \mathbb{R}^{T \times Q}$ of lower dimensionality $Q \ll D$ through a non-linear relationship.

Given the latent covariate space $\boldsymbol{X}$, the GPs are postulated to be independent across dimensions, giving rise of the following factorisation.

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{i=1}^{D} p\left(\boldsymbol{y}_i | \boldsymbol{X}\right) \tag{6}$$

where as $\boldsymbol{y}_i$ we denote a vector consisting of the i^{th} column of $\boldsymbol{Y}$. Note that we follow a completely analogous notation for the rest of this work.

The corresponding probability distribution of $\boldsymbol{y}_i$ is given by:

$$p\left(\boldsymbol{y}_i|\boldsymbol{X}\right) = \mathcal{N}\left(\boldsymbol{y}_i|0, \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} + \sigma^2\boldsymbol{I}\right) \tag{7}$$

1.1.3. Spike-and-slab Gaussian process

The spike and slab GP was first introduced in [24] and is described by eq. 8-12. On a higher level, this is a factor model with loading matrix $\boldsymbol{W}$ and latent factor $\boldsymbol{f}_j$ on which the authors impose a GP prior, with covariate matrix $\boldsymbol{X}$.

$$y_{ti} \sim \mathcal{N}\left(y_{ti}|\sum_{j=1}^{N_h} w_{ij} f_j(\boldsymbol{x}_t), \sigma_i^2\right) \tag{8}$$

$$w_{ij} = \hat{w}_{ij} h_{ij} \tag{9}$$

$$\hat{w}_{ij} \sim \mathcal{N}(\hat{w}_{ij}|0, \gamma_j^{-1}) \tag{10}$$

$$h_{ij} \sim Bernoulli(\pi_j) \tag{11}$$

$$\boldsymbol{f}_j \sim \mathcal{GP}(\boldsymbol{f}_j|0, \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})) \tag{12}$$

An additional contribution of this work is the spike-and-slab prior on the loading matrix, in order to achieve an automatic factor selection functionality and the "coupled" inference performed for estimating the posterior. More specifically, variational posteriors are formulated jointly for the "spike" (h_{ij}) and "slab" $(\hat{w}_{ij})$ variables, as follows: $q(\hat{w}_{nj}, h_{nj}) = q(h_{nj})q(\hat{w}_{nj}|h_{nj})$. As reported in the original work and also confirmed by our own experiments, this "coupled" inference leads to performance benefits.

Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

2. Proposed Approach

2.1. Motivation

Let us consider the case in which we have multiple sequences in our disposal, but perhaps none of them are complete. Each sequence $\boldsymbol{Y}_n \in \mathbb{R}^{T\times D}$ consists of multi-dimensional observations and we constrain them to be of fixed length T and dimension D. The sequences may be of either the same or different tasks, with the assumption that there is at least some redundancy to be harnessed among them.

We envision a model able to take into account the sequential nature of the data and the redundancies that manifest themselves among repetitions of a single task as well as those across different tasks. This can be achieved by supposing that our sequences are a linear superposition of a finite dictionary of not necessarily linear basis functions, shared among all instances. Indeed, this notion is frequently encountered in the literature for the purpose of modelling both image [9, 20] and motion data [37].

In the past, researchers have considered imposing either spherical Gaussian priors [18] or full GP priors [24] in the context of matrix completion and collaborative filtering with considerable success. Indeed, it has been well documented that certain datasets, such as motion and sound, may lie in much lower dimensional manifolds than those they are presented in [38]. Nevertheless, the assumption that this manifold coincides with the time component could be a bit restrictive. In fact, it is quite possible that the optimal manifold is not one dimensional and, even if it is, it might be represented by more complex one-dimensional trajectories.

2.2. Main Contributions

In this work we aim to contribute both from a theoretical and experimental perspective, by proposing a novel non-linear factor model and employing it for

Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

unsupervised time-series reconstruction. More specifically, we outline our main contributions as follows:

GPLVM extension for unsupervised data completion.

As previously outlined, the GPLVM, although highly successful in a variety of applications, has previous only been employed for data completion in a supervised manner. In this work, we extend the model for unsupervised learning. By doing so, we also obviate the need for a training set consisting of complete sequences and enable ourselves to learn directly from incomplete data.
Our approach is analogous to the way the spike-and-slab GP (SS-GP) [24] extends the conventional Gaussian process. More specifically, similarly to the SS-GP, which can be viewed as a factor model of GPs, our approach can be viewed as a factor model of Gaussian Process Dynamical Systems, as will be explained in more detail in the next section.

Reconstruction of time-series with non-uniform gaps. To the best of the authors' knowledge, it is the first time that matrix completion methods have been employed for time-series with large contiguous missing segments. Such methods usually thrive in image inpainting or signal interpolation, but are rarely (if ever) used for completing big continuous gaps.

The scenario we target is a highly realistic one and, demonstrated in this work, is directly applicable to automatically rectifying damaged motion capture sequences.

2.3. Assuptions

Despite being designed to utilise and, in fact, benefit from jointly handling multiple different tasks, our method implies certain assumptions about the properties those tasks should exhibit. We outline some of those properties as follows:

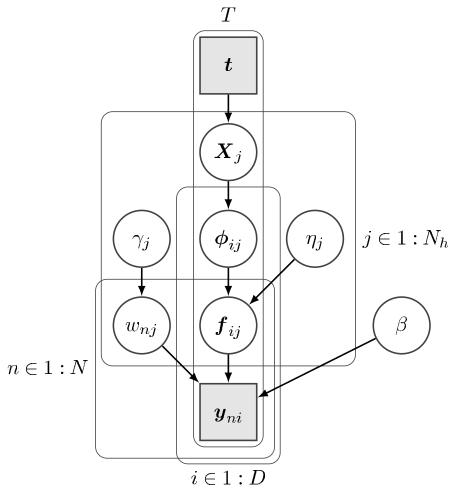


Figure 1: Simplified plate diagram of the proposed model (Multi-GPDS), showing roughly the dependence between variables. Note that $\boldsymbol{y}_{ni}, \boldsymbol{f}_{ij}, \boldsymbol{\phi}_{ij} \in \mathbb{R}^{T\times 1}$ and $\boldsymbol{X}_j \in \mathbb{R}^{T\times Q}$, with Q being the dimensionality of the latent covariate space.

Similarity. Different tasks should be selected so as not to be completely unrelated. This limitation is stemming from the fact that there is no factor sharing across data dimensions, thus tasks should not use disjoint sets of joint positions. In the unlikely event of completely unrelated tasks, the results are not expected to be any worse than a single-task approach, but the method would incur an unnecessarily high computational cost.

Timescale. Selected tasks should have similar time lengths, meaning that none of them should have much shorter or longer completion time.
This limitation is stemming from the fact that we need to fix the length of all sequences. In extreme circumstances, this preprocessing step could cause loss of information or artefacts due to excessive down- or up-sampling respectively.

2.4. Model Formulation

2.4.1. Note on notation

Throughout the rest of this work, we attempt to consistently make use of the following notation, so as to facilitate understanding of the model:

- Indices:
  - n: index of a single sequence (of joint locations in time).
  - i: index of a single dimension in the data.
  - t: index of a single point in time.
  - j: index of a single latent (hidden) model factor.
  - q: index of a single hidden covariate dimension.
- Dimensions:
  - N: total number of sequences in the dataset.
  - D: total number of data dimensions.
  - T: length of motions sequence in time.
  - N_h: total number of latent model factors.
  - Q: total number of latent covariate dimensions.
- Random variables:
  - $\boldsymbol{y}_{ni}$: motion development for sequence n and dimension i; vector of length T time-steps.
  - w_{nj}: loading weight of factor j in sequence n.
  - $\boldsymbol{f}_{ij}$: latent factor j for dimension i; vector of length T time-steps.
  - $\boldsymbol{x}_{tj}$: latent covariate governing factor j at time-step t; vector of length Q dimensions.

As we have described in the previous section, the training set $\{\boldsymbol{Y}_n\}_{n=1}^N$ consists of N sequences of D-dimensional data in time. Please note that we have flattened with respect to the 3D data coordinates (x, y, z) and the skeleton joints, so that $D = 3 \times (number\ of\ joints)$.

We postulate that each sequence can be represented by a dictionary of non-linear basis factors $\boldsymbol{f}_{ij} \in \mathbb{R}^T$ shared among all sequences as follows:

$$\boldsymbol{y}_{ni} \sim \mathcal{N}\left(\boldsymbol{y}_{ni} | \sum_{j=1}^{N_h} w_{nj} \boldsymbol{f}_{ij}, \beta^{-1} \boldsymbol{I}\right) \tag{13}$$

We define the aggregate loadings matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N_h}$, with $[\boldsymbol{W}]_{nj} \triangleq w_{nj}$ on which we impose a spherical Gaussian prior with column-wise sparsity as follows:

$$\boldsymbol{w}_j \sim \mathcal{N}(\boldsymbol{w}_j | 0, \gamma_j^{-1} \boldsymbol{I}) \tag{14}$$

where as $\boldsymbol{w}_j$ we denote the j^{th} column of the loading matrix $\boldsymbol{W}$.

As for the basis functions, we choose a Gaussian process dynamical system (GPDS) prior distributions, one for each latent basis matrix and with a different latent set of covariates $\boldsymbol{X}_j \triangleq \{\boldsymbol{x}_{tj}\}_{t=1}^T$, with $\boldsymbol{x}_{tj} \in \mathbb{R}^Q$. This choice was made so as to achieve a rich non-linear structure.

$$\boldsymbol{f}_{ij} | \boldsymbol{\phi}_{ij} \sim \mathcal{N}\left(\boldsymbol{f}_{ij} | \boldsymbol{\phi}_{ij}, \eta_j^{-1} \boldsymbol{I}\right) \tag{15}$$

$$\boldsymbol{\phi}_{ij} | \boldsymbol{X}_j \sim \mathcal{N}\left(\boldsymbol{\phi}_{ij} | 0, \boldsymbol{K}^{(j)}_{\boldsymbol{X}_j \boldsymbol{X}_j}\right) \tag{16}$$

$$\boldsymbol{x}_{qj} | \boldsymbol{t} \sim \mathcal{N}\left(\boldsymbol{x}_{qj} | 0, \boldsymbol{K}^{(j)}_{\boldsymbol{tt}}\right) \tag{17}$$

Finally, we impose Gamma priors on the precision parameters $\{\gamma_j, \eta_j\}_{j=1}^{N_h}$:

$$\gamma_j \sim \mathcal{G}\left(a_\gamma, \frac{1}{b_\gamma}\right) \tag{18}$$

$$\eta_j \sim \mathcal{G}\left(a_\eta, \frac{1}{b_\eta}\right) \tag{19}$$

and a standard Jeffrey's prior for the precision of the likelihood β: $p(\beta) = \beta^{-1}$ A simplified plate diagram of the proposed model is presented in fig. 1.

2.5. Variational Posterior Inference

2.5.1. Inducing Variables

Variational inference is, of course, intractable for this model due to the non-linear relationship between $\boldsymbol{X}_j$ and $\boldsymbol{\phi}_{ij}$ in eq. 16. However, tractable inference can be achieved by introducing the concept of "inducing variables" [33], as shown in [34] for the conventional GPDS. This concept is directly applicable to our case as well. More specifically, we can augment our original problem by introducing $M \ll T$ extra auxiliary sample points $\boldsymbol{u}_{ij}$, drawn from the same prior distribution as the original sample $\boldsymbol{\phi}_{ij}$, but located on a different set of latent locations, denoted as $\boldsymbol{Z}_j$:

$$\boldsymbol{u}_{ij} \sim \mathcal{N}(0, \boldsymbol{K}^{(j)}_{\boldsymbol{Z}_j \boldsymbol{Z}_j}) \tag{20}$$

For notational simplicity and in order to explicitly indicate the dimensionality of the corresponding kernel covariance matrices, we shall make the following notational replacements:

- $\boldsymbol{K}^{(j)}_{\boldsymbol{X}_j \boldsymbol{X}_j} \rightarrow \boldsymbol{K}_{TT}$
- $\boldsymbol{K}^{(j)}_{\boldsymbol{X}_j, \boldsymbol{Z}_j} \rightarrow \boldsymbol{K}_{TM}$
- $\boldsymbol{K}^{(j)}_{\boldsymbol{Z}_j, \boldsymbol{Z}_j} \rightarrow \boldsymbol{K}_{MM}$.

Note that we have also dropped the factor indices from the kernel matrices $\{\boldsymbol{K}_{TT}, \boldsymbol{K}_{TM}, \boldsymbol{K}_{MM}\}$ and will henceforth assume they refer to factor j although not explicitly indicated.

Given the inducing variables $\boldsymbol{u}_{dj}$, our initial sample can be directly retrieved as follows:

$$p(\boldsymbol{\phi}_{ij}|\boldsymbol{u}_{ij}, \boldsymbol{X}_j, \boldsymbol{Z}_j) = \mathcal{N}(\boldsymbol{\phi}_{ij}|\boldsymbol{B}_j \boldsymbol{u}_j, \boldsymbol{S}_j) \tag{21}$$

$$\boldsymbol{B}_j = \boldsymbol{K}_{TM}\boldsymbol{K}_{MM}^{-1} \tag{22}$$

$$\boldsymbol{S}_j = \boldsymbol{K}_{TT} - \boldsymbol{K}_{TM}\boldsymbol{K}_{MM}^{-1}\boldsymbol{K}_{MT} \tag{23}$$

by doing so, we no longer need to impose an explicit posterior on the original GP sample $\boldsymbol{\phi}_{ij}$, but rather simply update it through the much smaller auxiliary sample $\boldsymbol{u}_{ij}$ via Type-II maximum likelihood estimation.

2.5.2. Variational Factorisation

Given the introduction of inducing variables, the variational posterior for our model is factorised as follows:

$$\begin{aligned} q(\cdot) = \quad & \prod_{n=1}^{N} q(\boldsymbol{w}_n) \prod_{t=1}^{T} \prod_{i=1}^{D} q(\boldsymbol{f}_{ti}) \\ & \cdot \prod_{i=1}^{D} \prod_{j=1}^{N_h} p(\boldsymbol{\phi}_{ij}|\boldsymbol{u}_{ij}, \boldsymbol{X}_j) q(\boldsymbol{u}_{ij}) \\ & \cdot \prod_{j=1}^{N_h} \prod_{q=1}^{Q} q\left(\boldsymbol{x}_{qj}\right) q\left(\gamma_j\right) q\left(\eta_j\right) q\left(\beta\right) \end{aligned} \tag{24}$$

It should be noted that, inspired by [18], we have chosen a row-wise factorisation for the loadings $\boldsymbol{W}$ and factors $\boldsymbol{F}_i$, different from the column-wise factorisation used in the prior. This is not an intuitive choice since we normally aim to capture dependancies among data-points in time (rows), rather than across factors (columns) [26]. However, variational inference for the column-wise factorisation yields diagonal posterior covariance matrices and fails to represent the intended relationships, while a row-wise factorisation allows for a richer dependency structure and yield more accurate results.

2.5.3. Variational Posterior Distributions

Factors and Loadings Posteriors.

Optimising the lower bound yields analytical updates for most of the random variables in our model. We present our results in this section.

The posterior of the factor loadings $\boldsymbol{w}_n$ is formulated as follows:

$$q(\boldsymbol{w}_n) = \mathcal{N}\left(\boldsymbol{w}_n|\boldsymbol{\mu}_{w_n}, \boldsymbol{\Sigma}_{w_n}\right) \tag{25}$$

$$\boldsymbol{\Sigma}_{w_n} = [\boldsymbol{\Gamma} + \boldsymbol{V}_n]^{-1} \tag{26}$$

$$\boldsymbol{\mu}_{w_n} = \boldsymbol{\Sigma}_{w_n}\boldsymbol{c}_n \tag{27}$$

$$\boldsymbol{V}_n = \langle\beta\rangle \sum_{(t,i)\in\mathcal{O}_n} \left[\langle\boldsymbol{f}_{ti}\rangle\langle\boldsymbol{f}_{ti}\rangle^T + \boldsymbol{\Sigma}_{f_{ti}}\right] \tag{28}$$

$$\boldsymbol{c}_n = \langle\beta\rangle \sum_{(t,i)\in\mathcal{O}_n} y_{nit}\langle\boldsymbol{f}_{ti}\rangle \tag{29}$$

with $\boldsymbol{\Gamma} = diag\left[\langle\boldsymbol{\gamma}\rangle\right]$.

As $\mathcal{O}_n$ we denote the set consisting of the dyads of indices (t, i), which are observed in sequence $\boldsymbol{Y}_n$, the rest of them would be missing.

Note that $\boldsymbol{V}_n$ is a full $N_h \times N_h$, rather than diagonal, matrix and consequently so is the posterior covariance $\boldsymbol{\Sigma}_{w_n}$.

The posterior of the intermediate factor variables $\boldsymbol{f}_{ti}$ can be formulated as follows:

$$q(\boldsymbol{f}_{ti}) = \mathcal{N}\left(\boldsymbol{f}_{ti}|\boldsymbol{\mu}_{f_{ti}}, \boldsymbol{\Sigma}_{f_{ti}}\right) \tag{30}$$

$$\boldsymbol{\Sigma}_{f_{ti}} = [\boldsymbol{H} + \boldsymbol{L}_n]^{-1} \tag{31}$$

$$\boldsymbol{\mu}_{f_{ti}} = \boldsymbol{\Sigma}_{f_{ti}}\left[\boldsymbol{H}\langle\boldsymbol{\phi}_{ti}\rangle + \boldsymbol{z}_n\right] \tag{32}$$

$$\boldsymbol{L}_{ti} = \langle\beta\rangle \sum_{n\in\mathcal{O}_{ti}} \left[\langle\boldsymbol{w}_n\rangle\langle\boldsymbol{w}_n\rangle^T + \boldsymbol{\Sigma}_{w_n}\right] \tag{33}$$

$$\boldsymbol{z}_{ti} = \langle\beta\rangle \sum_{n\in\mathcal{O}_{ti}} y_{nit}\langle\boldsymbol{w}_n\rangle \tag{34}$$

with $\boldsymbol{H} = diag\left[\langle\boldsymbol{\eta}\rangle\right]$.

Equivalently, as $\mathcal{O}_{ti}$ we denote the set consisting of the indices of sequences $\boldsymbol{Y}_n$ for which the time-step t of dimension i is observed.

Gaussian Process Dynamical System Posteriors.

The posteriors of the GPDS variables are slightly more complicated to formulate. As previously mentioned, we follow similar principles to [34] and [33] in devising our own solution.

More specifically, we only need to formulate the posterior of the inducing variables $\boldsymbol{u}_{ij}$, through which we are able to retrieve a posterior estimate for the $\boldsymbol{\phi}_{ij}$'s. Following that, we may then completely disregard the $\boldsymbol{u}_{ij}$'s.

Let us denote the following expectations of the various kernel matrices w.r.t. the latent locations:

$$\psi_0 = tr\left\{\langle\boldsymbol{K}_{TT}\rangle_{q(\boldsymbol{X}_j)}\right\} \tag{35}$$

$$\boldsymbol{\Psi}_1 = \langle\boldsymbol{K}_{TM}\rangle_{q(\boldsymbol{X}_j)} \tag{36}$$

$$\boldsymbol{\Psi}_2 = \langle\boldsymbol{K}_{MT}\boldsymbol{K}_{TM}\rangle_{q(\boldsymbol{X}_j)} \tag{37}$$

As shown in [33], these expectations are tractable for certain kernel functions. For notational simplicity we have dropped the factor index from quantities $\{\psi_0, \boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2\}$ and assume they all refer to factor j despite not explicitly indicated.

Optimising the lower bound yields the following solution for the posterior of the inducing variables:

$$q\left(\boldsymbol{u}_{ij}\right) = \mathcal{N}\left(\boldsymbol{u}_{ij}|\boldsymbol{\mu}_{u_{ij}}, \boldsymbol{\Sigma}_{u_{ij}}\right) \tag{38}$$

$$\boldsymbol{\Sigma}_{u_{ij}} = \boldsymbol{K}_{MM}^T\boldsymbol{A}_j\boldsymbol{K}_{MM} \tag{39}$$

$$\boldsymbol{\mu}_{u_{ij}} = \boldsymbol{K}_{MM}^T\boldsymbol{A}_j\boldsymbol{\Psi}_1^T\boldsymbol{\mu}_{f_{ij}} \tag{40}$$

Based on the posterior of $\boldsymbol{u}_{ij}$, we obtain the mean of $\boldsymbol{\phi}_{ij}$, as follows:

$$\boldsymbol{\mu}_{\phi_{ij}} = \langle \eta_j \rangle \boldsymbol{\Psi}_1 \boldsymbol{A}_j \boldsymbol{\Psi}_1^T \boldsymbol{\mu}_{f_{ij}} \tag{41}$$

with $\boldsymbol{A}_j = \left(\boldsymbol{K}_{MM} + \langle \eta_j \rangle \boldsymbol{\Psi}_2\right)^{-1}$. Another quantity necessary for the rest of the inference is the full posterior expectation of $\left\langle \boldsymbol{\phi}_{ij}^T \boldsymbol{\phi}_{ij} \right\rangle$, which is as follows:

$$\begin{aligned} \left\langle \boldsymbol{\phi}_{ij}^T \boldsymbol{\phi}_{ij} \right\rangle = \; & \psi_0 - tr\left\{\boldsymbol{K}_{MM}^{-1} \boldsymbol{\Psi}_2\right\} + tr\left\{\boldsymbol{\Psi}_2 \boldsymbol{A}_j\right\} && (42) \\ & + \langle \eta_j \rangle^2 \boldsymbol{\mu}_{f_{ij}}^T \Psi_1 A_j^T \Psi_2 A_j \Psi_1^T \boldsymbol{\mu}_{f_{ij}} && (43) \end{aligned}$$

For the above result we have used eq. 21 as well as the following:

$$\langle \boldsymbol{B}_j \rangle = \boldsymbol{\Psi}_1 \boldsymbol{K}_{MM}^{-1} \tag{44}$$

$$\left\langle \boldsymbol{B}_j^T \boldsymbol{B}_j \right\rangle = \boldsymbol{K}_{MM}^{-T} \boldsymbol{\Psi}_2 \boldsymbol{K}_{MM}^{-1} \tag{45}$$

$$tr\left\{\langle \boldsymbol{S}_j \rangle\right\} = \psi_0 - tr\left\{\boldsymbol{K}_{MM}^{-1} \boldsymbol{\Psi}_2\right\} \tag{46}$$

Precision Variables Posteriors.

The posterior expectations of the precision variables β, $\{\gamma_j\}_{j=1}^{N_h}$ and $\{\eta_j\}_{j=1}^{N_h}$ are given as follows:

$$\langle \beta \rangle = \frac{\#\mathcal{O}}{\sum_{(n,t,i)\in\mathcal{O}} \left\langle \left(y_{nit} - \boldsymbol{w}_n^T \boldsymbol{f}_{ti}\right)^2 \right\rangle} \tag{47}$$

where $\mathcal{O}$ is the set of triplets (n, t, i) of observed data and $\#\mathcal{O}$ is the cardinality of this set.

$$\langle \gamma_j \rangle = \frac{2a + N}{2b + \left\langle \boldsymbol{w}_j^T \boldsymbol{w}_j \right\rangle} \tag{48}$$

and

$$\langle \eta_j \rangle = \frac{2a + TD}{2b + \sum_{i=1}^{D} \left\langle \left(\boldsymbol{f}_{ij} - \boldsymbol{\phi}_{ij}\right)^T \left(\boldsymbol{f}_{ij} - \boldsymbol{\phi}_{ij}\right) \right\rangle} \tag{49}$$

The posterior expectations required for the above updates are given below:

$$\left\langle \boldsymbol{w}_j^T \boldsymbol{w}_j \right\rangle = \sum_{n=1}^{N} \left(\left[\boldsymbol{\mu}_{w_n}\right]_j^2 + \left[\boldsymbol{\Sigma}_{w_n}\right]_{jj} \right) \tag{50}$$

$$\left\langle \boldsymbol{f}_{ij}^T \boldsymbol{f}_{ij} \right\rangle = \sum_{t=1}^{T} \left(\left[\boldsymbol{\mu}_{f_{ti}}\right]_j^2 + \left[\boldsymbol{\Sigma}_{f_{ti}}\right]_{jj} \right) \tag{51}$$

2.5.4. Hyper-parameter Optimisation

The hyper-parameters that are optimised rather than inferenced are the ones pertaining to the individual GPDS factors. In order to estimate them, we follow closely the approach introduced in [34].

The optimisation itself involves Type-II maximum likelihood with respect to:

1. The kernel hyper-parameters of latent kernel matrices $\boldsymbol{K}_{TT}$, $\boldsymbol{K}_{TM}$ and $\boldsymbol{K}_{MM}$.
2. The time kernel matrix $\boldsymbol{K}_{tt}$.
3. The inducing locations $\boldsymbol{Z}_j$.
4. The parameters of the latent locations posterior distribution $q(\boldsymbol{X}_j) = \mathcal{N}\left(\boldsymbol{X}_j|\cdot\right)$.

As the posteriors $q(\boldsymbol{X}_j)$ are also Gaussian, we typically need to estimate means and covariances, which involves $\mathcal{O}\left(T^2\right)$ number of parameters per factor. Due to the high number of parameters this is both computationally intensive and slow to converge. For that reason, the authors in [34] have reparametrised the problem using a Gaussian approximation [7], involving $\mathcal{O}(T)$ number of parameters and as a result the approach scales much better with the number of training data.

In brief, the parameters to be estimated by gradient ascent methods are $\boldsymbol{\Theta}_j = \left\{\boldsymbol{\theta}_j^{(x)}, \boldsymbol{\theta}_j^{(t)}, \boldsymbol{Z}_j, \boldsymbol{\lambda}_j, \boldsymbol{\mu}_{X_j}\right\}$. The likelihood terms dependent on those parameters can be computed by integrating out all other random variables analytically.

To be more specific we begin by considering only the terms containing $\boldsymbol{\Theta}_j$ given in eq. 54. Then we compute the expectation w.r.t. $\boldsymbol{f}_{ij}$, yielding eq. 55. Following that, we can integrate out the rest of the variables, which yields the likelihood to be optimised in eq. 56. Finally, the Gaussian integral I_{ij} is given in eq. 57.

$$\begin{aligned}\left\langle \left(y_{nit} - \boldsymbol{w}_n^T \boldsymbol{f}_{ti}\right)^2 \right\rangle &= \left(y_{nit} - \langle \boldsymbol{w}_n \rangle^T \langle \boldsymbol{f}_{ti} \rangle\right)^2 + &(52)\\ &+tr\left\{\boldsymbol{\mu}_{w_n}\boldsymbol{\mu}_{w_n}^T \boldsymbol{\Sigma}_{f_{ti}} + \boldsymbol{\Sigma}_{w_n}\boldsymbol{\mu}_{f_{ti}}\boldsymbol{\mu}_{f_{ti}}^T + \boldsymbol{\Sigma}_{w_n}\boldsymbol{\Sigma}_{f_{ti}}\right\} &(53)\end{aligned}$$

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Theta}_j) &\propto \sum_{i=1}^{N_y} \int q(\boldsymbol{\Omega}_{ij}) q(\boldsymbol{f}_{ij}) \ln \frac{\mathcal{N}\left(\boldsymbol{f}_{ij} | \boldsymbol{\phi}_{ij}, \eta_j^{-1}\boldsymbol{I}\right) p(\boldsymbol{f}_{ij}, \boldsymbol{u}_{ij}, \boldsymbol{X}_j | \boldsymbol{t})}{q(\boldsymbol{\phi}_{ij}, \boldsymbol{u}_{ij}, X_j)} d\boldsymbol{f}_{ij} d\boldsymbol{\Omega}_{ij} &(54)\\ &= \sum_{i=1}^{N_y} \int p(\boldsymbol{\phi}_{ij}|\cdot) q(\boldsymbol{X}_j) q(\boldsymbol{u}_{ij}) \ln \frac{\mathcal{N}\left(\boldsymbol{\mu}_{f_{ij}} | \boldsymbol{\phi}_{ij}, \langle \eta_j \rangle^{-1} \boldsymbol{I}\right) p(\boldsymbol{X}_j) p(\boldsymbol{u}_{ij} | \boldsymbol{Z}_j)}{q(\boldsymbol{X}_j) q(\boldsymbol{u}_{ij})} d\boldsymbol{\Omega}_{ij} - \\ &- \frac{\langle \eta_j \rangle}{2} \sum_{\forall (t,i)} [\boldsymbol{\Sigma}_{f_{ti}}]_{jj} &(55)\\ &= \sum_{i=1}^{N_y} \ln I_{ij} - \frac{1}{2}\langle \eta_j \rangle \sum_{\forall (t,i)} [\boldsymbol{\Sigma}_{f_{ti}}]_{jj} - \frac{1}{2}\langle \eta_j \rangle \psi_0 + \frac{1}{2}\langle \eta_j \rangle tr\left\{\boldsymbol{K}_{MM}^{-1}\boldsymbol{\Psi}_2\right\} \\ &-KL(q(\boldsymbol{X}_j) \parallel p(\boldsymbol{X}_j)) &(56)\end{aligned}$$

where, for brevity, $\boldsymbol{\Omega}_{ij} = \left\{\boldsymbol{\phi}_{ij}, \boldsymbol{u}_{ij}, \boldsymbol{X}_j\right\}$.

$$\begin{aligned}I_{ij} = &\int e^{\left\langle \ln \mathcal{N}\left(\boldsymbol{\mu}_{f_{ij}} | \boldsymbol{B}_j \boldsymbol{u}_{ij}, \langle \eta_j \rangle^{-1} \boldsymbol{I}\right)\right\rangle} \mathcal{N}(\boldsymbol{u}_{ij}|0, \boldsymbol{K}_{MM}) d\boldsymbol{u}_{ij} = \\ &= \frac{\langle \eta_j \rangle^{\frac{N}{2}} |\boldsymbol{K}_{MM}|^{\frac{1}{2}}}{|\boldsymbol{K}_{MM} + \langle \eta_j \rangle \boldsymbol{\Psi}_2|^{\frac{1}{2}}} e^{-\frac{1}{2}\boldsymbol{\mu}_{f_{ij}}^T \left[\langle \eta_j \rangle \boldsymbol{I} - \langle \eta_j \rangle^2 \boldsymbol{\Psi}_1 (\boldsymbol{K}_{MM} + \langle \eta_j \rangle \boldsymbol{\Psi}_2)^{-1} \boldsymbol{\Psi}_1^T\right] \boldsymbol{\mu}_{f_{ij}}} &(57)\end{aligned}$$

3. Empirical Evaluation

3.1. Experimental Setting

Table 1: Dataset details. Note that the motion capture sequences are provided in 3D join position coordinates.

	#Drawing 8s [39]	#Dance Primitives [40]	#HDM05 [41]
#Sequences	24	449	168
#Actions	1	6	8
#Repetitions	24	{57,83,84}	{16,20,52}
#Timepoints	121	73	70
#Dimensions	4	23×3	31×3

3.1.1. Comparisons

For the purpose of empirically evaluating our approach we have compared against popular models in the literature, some used as originally proposed and others adapted to unsupervised sequence completion.

To be more specific, we aim at quantitatively assessing the efficacy of two main aspects of our model:

- The imposition of GPDS priors for dictionary components
- The column-wise prior and row-wise posterior specification of the factor loading weights (section 2.5.2)

For that purpose we compare against the following methods:

- **Interpolation**: The simplest way to conduct missing data reconstruction. This method is aimed to serve as a baseline for all evaluated methods and highlight the necessity of more sophisticated approaches under the current setting. The particular interpolation method we have used follows the spring metaphor[2]. It should also be noted, that interpolation is the only

[2]More specifically, we have made use of method 4 from the code published at `http://uk.mathworks.com/matlabcentral/fileexchange/4551-inpaint-nans`.

one of the methods that does not require fixed-length sequences. Despite that, we have used the same exact data in order to yield comparable results.

- **SVT** [42] and **FPC** [43]: The methods are dubbed singular value thresholding (SVT) and fixed point completion (FPC). They are both non-Bayesian and non-sequential models employing point-wise optimisation for the purpose of matrix completion.

- **SS-GP** [24]: We briefly introduced this model in section 1.1.3. In order to make it applicable to our multi-task scenario, we have adapted it to the likelihood of eq. 13, but have kept the original prior specification, variational factorisation and joint spike-and-slab posterior inference. The adapted model specification is given in eq. 58-61.

  The main factors distinguishing the multi-GPDS from the SS-GP, can be summarised as follows:

  – We use ARD, instead of spike-and-slab, priors. This is not expected to introduce any performance benefits.

  – We impose GPDS priors on the temporal factors $\boldsymbol{f}_{ij}$, instead of simple GPs. As previously discussed the GPDS prior offers one more latent hierarchy level and higher expressive power.

  – The variational posteriors of the factor loading matrix $\boldsymbol{W}$ are fully factorised in the SS-GP: $\prod_n \prod_j q(w_{nj})$. Inspired by [18], we have chosen partially factorised posteriors: $\prod_n q(\boldsymbol{w}_n)$. As mentioned before, this results in full (rather than diagonal) posterior covariance matrices and captures richer dependencies, but also introduces higher computational burden.

Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

$$w_{nj} = \hat{w}_{nj} h_{nj} \tag{58}$$

$$\hat{w}_{nj} = \mathcal{N}(w_{nj}|0, \gamma_j^{-1}) \tag{59}$$

$$h_{nj} = Bernoulli(\pi_j) \tag{60}$$

$$\boldsymbol{f}_{ij} \sim \mathcal{GP}(\boldsymbol{f}_{ij}|0, \boldsymbol{K}(\boldsymbol{t}, \boldsymbol{t})) \tag{61}$$

- **IBP-GP**: A model otherwise identical to the SS-GP with the difference that it assumes an Indian Buffet process prior (IBP) [44] on the factor loadings to induce sparsity. We have not found an equivalent method in the bibliography (using GP priors on the factors), however the extension is straightforward. Note that we adopt the same principle to the SS-GP during inference by using a coupled variational posterior, which resulted in some performance gains. Due to the non-parametric IBP prior we expect this method to perform better compared to its parametric counterpart (SS-GP).

$$w_{nj} = h_{nj} \hat{w}_{nj} \tag{62}$$

$$h_{nj}|\boldsymbol{u} \sim Bernoulli(\pi_j(\boldsymbol{u})) \tag{63}$$

$$\pi_j(\boldsymbol{u}) = \prod_{k=1}^{j} u_k \tag{64}$$

$$u_k \sim Beta(\gamma, 1) \tag{65}$$

$$\hat{w}_{nj} \sim \mathcal{N}(\hat{w}_{nj}|0, \gamma_j^{-1}) \tag{66}$$

- **VBMC** [18]: Variational Bayesian Matrix Completion (VBMC) is a model assuming spherical Gaussian priors for the purpose of general matrix completion. In order to use this method in our setting we have flattened the time and joints dimensions. This model is able to capture richer dependency structures by coupling factors and loadings, which is achieved by factorising the prior row-wise and the posterior column-wise, leading to full, instead of diagonal, posterior covariance matrices. Note that although this is not an explicitly sequential model, it implicitly models all dependancies by means of a joint posterior distribution over time and joints. We expect it to work well, but not better than the Multi-GPDS, due to the fact that spherical Gaussian priors are generally less expressive that GPDS priors.

- **Multi-GP**: A model otherwise the same as the Multi-GPDS with the difference that it assumes GP, instead of GPDS priors, on the factors. We have compared against this model in order to illustrate the benefits of introducing a more complex GPDS prior on the factors.

- **AR1**: A model otherwise the same as the Multi-GPDS with the difference that it assumes an auto-regressive prior AR_1 on the factors. We also impose a joint Normal-Gamma prior on the mean and precision of the autoregressive prior. This model has been chosen for comparison in order to highlight the necessity of non-linear GPDS priors on the factor components.

$$f_{1ij} \sim \mathcal{N}\left(f_{1ij}|\mu_{ij}, \lambda_{ij}^{-1}\right) \tag{67}$$

$$f_{tij} \sim \mathcal{N}\left(f_{tij}|f_{(t-1)ij}, \lambda_{ij}^{-1}\right) \tag{68}$$

$$\mu_{ij}, \lambda_{ij} \sim \mathcal{N}\left(\mu_{ij}|\mu_0, (\beta_0\lambda_{ij})^{-1}\right) \mathcal{G}\left(\lambda_{ij}|a_0, \frac{1}{b_0}\right) \tag{69}$$

3.1.2. Data Removal Scheme

In this work we have utilised complete data sequences, whose details we provide in Table 1. From the complete sequences we remove consecutive segments in a manner that is designed to be as realistic and challenging as possible. Our sole purpose in using complete sequences is to have access to the corresponding ground-truth necessary for quantitatively assessing the reconstruction accuracy.

In the data removal scheme we aim at removing a variable random number of segments, of variable random length. More specifically, throughout our experimental evaluation, we perform data removal according to the algorithm outlined in the following paragraph.

Description of data removal algorithm.

Inputs:

- $r \in \mathbb{R}$: Desired missing data ratio.
- $\lambda \in \mathbb{N}$: Intensity of Poisson distribution for the number of missing data segments.
- $T \in \mathbb{N}$: Length of a single data sequence.

Outputs:

- $\boldsymbol{q} \in \mathbb{N}$: number of missing segments in the sequence.
- $\{m_k\}_{k=1}^{q} \in \mathbb{N}$: vector containing the lengths of missing data segments.
- $\{s_k\}_{k=1}^{q} \in \mathbb{N}$: vector containing the starting points of the missing data segments.

Algorithm:

- Calculate the total number of data-points N_r to be removed from the sequence according to the desired missing data ratio r:

  $N_r = rT$, where T is the length of the sequence.

- Draw the number of missing segments q according to a truncated (strictly positive) Poisson distribution with intensity λ:

  $p(q|\lambda) \propto \delta(q > 0) Poisson(\lambda)$

  This is possible by simply resampling in case of a zero draw (rejection sampling).

- Draw a vector $\boldsymbol{\ell}$ of length q from a uniformly distribution:

  $\boldsymbol{\ell} \sim \mathcal{U}([0,1])$

- Normalise $\boldsymbol{\ell}$ so that it sums to one: $\sum_{k=1}^{q} \ell_k = 1$.

  ℓ_k now represents the proportion of N_r that will be removed.

- `for` $k = 1 \ldots q$:

  - Calculate the length m_{nk} of the k^{th} missing data segment of the sequence, as follows:

    $m_k = \lfloor N_r \ell_k \rfloor$, where $\lfloor \cdot \rfloor$ is the 'floor' operator.

  - Estimate the length b_k of a bounding box for the missing k^{th} segment, as follows:

    $b_k = \lceil T \ell_k \rceil$, where $\lceil \cdot \rceil$ is the 'ceiling' operator and both b_k and m_k represent the same proportion of T and N_r respectively: $\frac{b_k}{T} = \frac{m_k}{N_r} = \ell_k$.

  - Draw a starting position s_k, uniformly at random, within the bounding box of length b_k to place the k^{th} missing segment within it: $s_k \sim \mathcal{U}\left[0, b_k - m_k\right]$

    In order to guarantee the non-overlapping placement of all segments we enforce the following constraint on the starting points: $0 \leq s_k < b_k - m_k$.

3.1.3. Data Processing

In order to ensure a realistic experimental scenario we conduct processing of the sequences after the data removal.

All evaluated algorithms, except the interpolation, require sequences of the same length, hence we fix the sequences to their average length by subsampling the longer or linearly interpolating the shorter ones. Note that this process achieves an effect similar to speed normalisation, but does not align the sequences in time and does not account for different part of the sequence being executed at different speeds.

Finally, prior to presenting the data to each algorithm, we have normalised the incomplete sequences by removing their mean per dimension and scaling according to their total standard deviation. The normalisation is performed based only on the observed data-points.

3.1.4. Implementation and Setting

In our implementation we attempt to ensure consistency, impartiality and reproducibility by following a number of principles as outlined below.

For the purpose of consistency, we have used the same GP library [27] and the same kernel structure in time for all methods (details will follow).

In order to account for the influence of random initialisation, all presented results are calculated by accumulating statistics over 30 independent repetitions, with different random initialisations and random data removal per repetition. In order to ensure impartiality, we seed the random number generators identically across methods and use the same or completely analogous random initialisation schemes.

For all experiments we present our results for missing data ratios ranging from 50% to 80% with increments of 10%. With bold lettering we mark the best statistically significant results, determined by applying the Student-t test.

We provide all hyper-parameters necessary for reproducing our experiments below:

- λ: Poisson intensity, or else the expected number of missing data segments. This parameter is set to $\lambda = 2$ for all experiments.
- N_h: Number of latent factors or initial rank for the VBMC, SVT and FPC

methods. In the Robotic experiment, $N_h = 32$ for all methods except the the Multi-GPDS for which $N_h = 32$. In the MoCap experiments we use $N_h = 64$ for all methods.

- Q: Dimension of the GPDS latent covariate space $\boldsymbol{X}$. We use $Q = 2$ for the Robotic dataset and $Q = 1$ for the MoCap datasets.
- N_{GD}: Maximum number of gradient descent iterations. This is applicable for all gradient-based algorithms (SVT and FPC), where $N_{GD} = 1000$, and those employing gradient descent for kernel hyper-parameter optimisation (SS-GP, IBP-GP, Multi-GP and Multi-GPDS), where $N_{GD} = 10$.
- N_{freq}: Kernel update frequency. We update the kernel hyper-parameters every N_{freq} iterations in order to reduce computational costs. $N_{freq} = 10$ in the Robotic experiment and $N_{freq} = 5$ in the MoCap experiments.
- Kernel Functions: We have used a Square Exponential (SE) kernel for $\boldsymbol{K}_{xx}$ and a composite Radial Basis Function + White noise + Bias term for $\boldsymbol{K}_{tt}$.

Note that in the case of the robotic experiment, we observed that the evaluated methods performed better when optimising the kernel parameters more rarely, every 10 instead of every 5 VB iterations. Also, due to the relatively simpler nature of the robotic data, less factors were sufficient and most methods achieved optimal performance for 32 factors, whereas the Multi-GPDS performed best with 16 factors.

Finally, our results and source code necessary to reproduce all experiments are available through the first author's website `http://www.korkinof.com` and the Personal Robotics Lab website `http://www3.imperial.ac.uk/personalrobotics`.

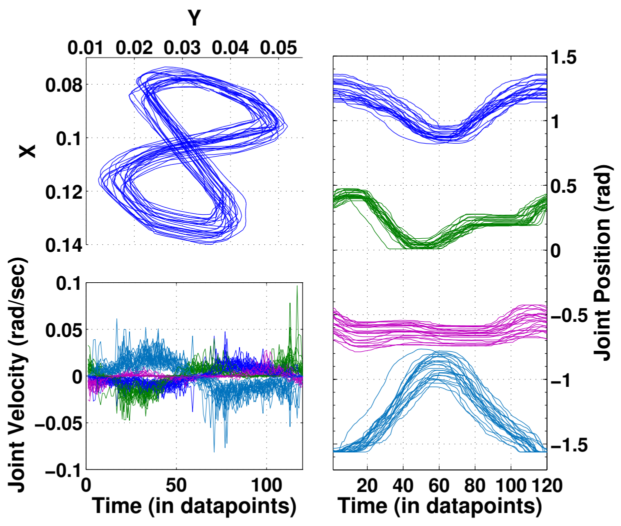


Figure 2: **Drawing *lazy figure* 8s**: Upper left: End effector position on the experiment's drawing plane. Lower left: Angular velocities (rad/s) per joint. Right: Angular positions (rad) per joint. (figure best viewed in colour)

3.2. Robotic Data

This dataset consists of joint angle positions of a NAO humanoid robot (academic edition) recorded while drawing lazy figure 8s. The dataset includes 24 distinct demonstrations presented to the NAO robot by means of kinesthetics[3] and part of it has previously been used in [39] for online trajectory learning. As can be seen in fig. 2, the dataset is noisy, each demonstration has different points of origin and there is significant variation among demonstrations. The task itself entails challenges for learning algorithms and for that reason is widely considered a classical benchmark [38].

The results pertaining to this experiment are presented in Table 2. As may be observed, the Multi-GPDS achieves overall the best reconstruction accuracy, with the exception of the highest missing data ratio (80%) where its performance is statistically equivalent to the AR1. In all other cases the Multi-GPDS

[3]Manual movement of the joints in the desired positions during the demonstration.

Table 2: **Drawing lazy figure 8s**: Reconstruction root mean square error for all evaluated methods. Each column represents results for a different missing data ratio. The scale of the results is 10^{-2}.

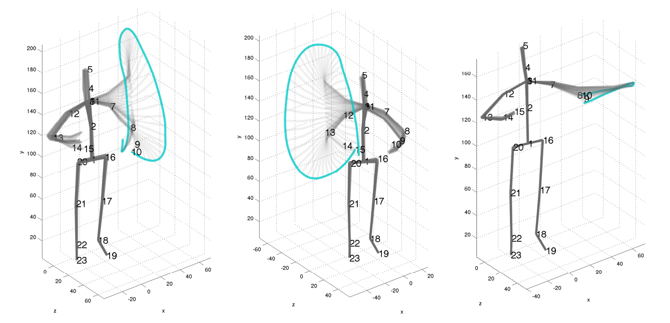
	50%	**60%**	**70%**	**80%**
Interp.	5.85(0.24)	9.71(0.52)	11.1(0.63)	13.1(1.18)
SVT	4.24(0.55)	5.56(0.78)	7.86(1.26)	9.54(1.30)
FPC	4.29(0.55)	5.54(0.76)	7.80(1.24)	9.54(1.27)
SS-GP	2.73(0.24)	3.47(0.52)	4.79(0.63)	6.56(1.19)
IBP-GP	2.92(0.54)	3.73(0.79)	4.87(0.75)	5.79(0.81)
Multi-GP	2.66(0.30)	3.29(0.37)	4.07(0.42)	4.66(0.32)
VBMC	2.89(0.28)	3.71(0.37)	5.30(0.98)	7.05(0.90)
AR1	2.50(0.25)	3.00(0.31)	3.64(0.43)	**4.24(0.31)**
Multi-GPDS	**2.20(0.24)**	**2.59(0.28)**	**3.32(0.49)**	**4.28(0.42)**

Table 3: **Dance Primitives Dataset**: Reconstruction RMSE for all evaluated methods. Column represents results for a different missing data ratios.

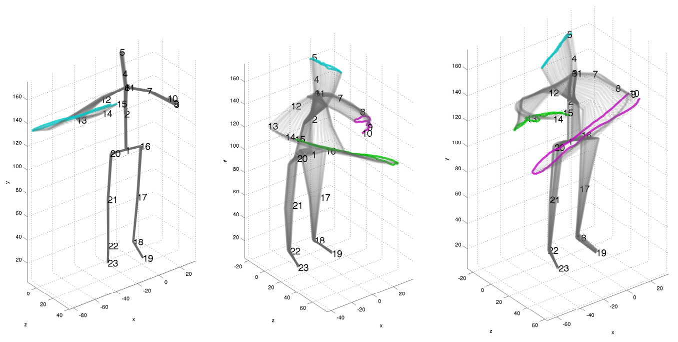
	50%	**60%**	**70%**	**80%**
Interp.	2.75(0.20)	4.06(0.36)	5.63(0.32)	7.01(0.31)
SVT	2.17(0.05)	2.73(0.06)	3.58(0.08)	4.90(0.11)
FPC	1.45(0.05)	1.93(0.07)	2.72(0.08)	4.18(0.12)
SS-GP	1.10(0.03)	1.34(0.04)	1.71(0.06)	2.47(0.18)
IBP-GP	1.07(0.03)	1.30(0.04)	1.62(0.06)	2.30(0.19)
Multi-GP	1.08(0.03)	1.43(0.05)	1.99(0.06)	3.12(0.10)
VBMC	**1.02(0.02)**	1.27(0.04)	1.66(0.05)	2.50(0.07)
AR1	1.08(0.02)	1.38(0.04)	1.78(0.05)	2.34(0.07)
Multi-GPDS	**1.00(0.02)**	**1.20(0.04)**	**1.47(0.05)**	**1.84(0.08)**

outperforms AR1 10%-14%.

With regard to the rest of the evaluated algorithms, the Multi-GP performs quite well in this experiment, followed by the SS-GP and the VBMC. The IBP-GP underperforms for this dataset, as the number of factors selected appears to be too low for the non-parametric prior. Finally, the SVT and FPC perform almost identically and significantly better than interpolation, but worse that all other methods.



(a) LHRB (b) RHRB (c) LHE



(d) RHE (e) ATR (f) CTR

Figure 3: **Dance primitives**: Figures (a)-(f) show each separate action, which include the following: (a): Left hand rotation (LHR), (b): Right hand rotation (RHR), (c): Left hand extension (LHE) (d): Right hand extension (RHE) (e): Anti-clockwise torso rotation (ATR) and (f): Clockwise torso rotation (CTR) (figures best seen in colour)

3.3. Motion Capture Data

3.3.1. Dance Primitives

This dataset consists of 6 human dance motion primitives and has previously been used in learning dance sequences by means of context-free stochastic grammars [45, 40]. The data is recorded as time-series using an OptiTrack 8-camera motion capture system.

There are 449 sequences in total, belonging to 8 distinct actions with not necessarily the same number of repetition per action class. The data is given in raw 3D positions in time and includes only upper body motion. We illustrate the actions in fig. 3, where we also provide a short task description and corresponding abbreviation for quick reference. For a better understanding of the dataset we refer to the following video shared by the authors in [40]: `http://www.youtube.com/watch?v=S99ViThK050`.

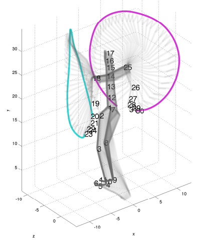
Reconstruction errors for all evaluated methods are presented in Table 3. We observe that the differences among algorithms tend to be more prominent in higher missing data ratios, whereas their performance almost converges in lower ratios. Compared to the second best performing method, the Multi-GPDS achieves the globally best reconstruction accuracy, with differences ranging from 2.5% in lower missing data ratios, up to 20% for the highest ratio of 80%.

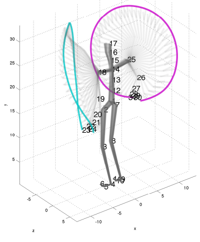
As far as the other methods are concerned, the IBP-GP and SS-GP perform very well in this experiment, but their performance is being matched and surpassed in lower missing data ratios by the VBMC. It becomes obvious that the GP prior has an advantage in more severely corrupt sequences. Finally, the gradient-based methods SVT and FPC perform considerably worse, with the FPC outperforming the SVT by a large margin.

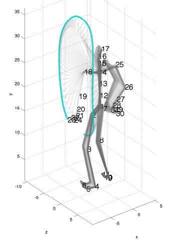
Visual results of the reconstruction achieved my the multi-GPDS for this dataset can be seen in the following video: `https://youtu.be/s9Eni_H4yQM`.

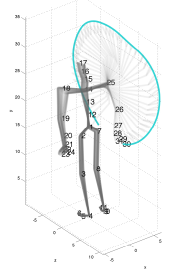
3.3.2. HDM05 Dataset

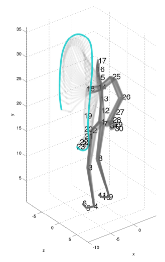
The HDM05 database [41] contains motion capture data recorded at the Hochschule der Medien (HDM) in the year 2005 under the supervision of Bernhard Eberhardt. It consists of around 70 motions classes, with up to 50 repetitions of each action from a variety of actors and recorded using a Vicon system. In its entirety, the database contains over 1500 systematically recorded and documented motion sequences with a total duration of over 3 hours.

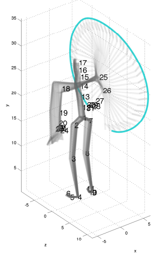

(a) BHRB

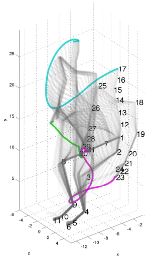

(b) BHRF


(c) LHRB


(d) RHRB


(e) LHRF


(f) RHRF


(g) SoC

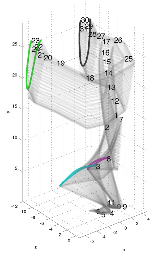

(h) SHE

Figure 4: **HDM05 Dataset**: Figures (a)-(h) show each separate action, which include the following: (a): Both hand rotation backwards (BHRB), (b): Both hand rotation forwards (BHRF), (c): Left hand rotation backwards (LHRB), (d): Right hand rotation backwards (RHRB), (e): Left hand rotation forwards (LHRF), (f): Right hand rotation forwards (RHRF), (g): Sitting down on chair (SoC) and (h) Squat with both hands extended (SHE) (figures best seen in colour)

For the purpose of this experiment we have selected the 8 actions illustrated in fig. 4. We have included primitive actions, closely related to the ones used in our "dance primitives" experiment, such as LHRB, RHRB, LHRF and RHRF (for these abbreviations we refer to fig. 4), as well as more complex actions consisting of a combination of those primitive actions and involving both hands, such as BHRB and BHRF. Finally, two actions involving full body motion (SoC and SHE) have been included in order to increase the difficulty of the experiment. Additionally, as tasks (g) and (h) are relatively unrelated to tasks (a)-(f), we expect limited factor sharing to take place across those two groups.

Table 4: **HDM05 Dataset**: Reconstruction RMSE for all evaluated methods. Each column represents results for a different missing data ratio.

	50%	**60%**	**70%**	**80%**
Interp.	0.71(0.05)	1.04(0.06)	1.46(0.08)	2.000(0.113)
SVT	0.70(0.01)	0.92(0.01)	1.27(0.03)	1.90(0.03)
FPC	0.67(0.01)	0.90(0.02)	1.27(0.03)	1.92(0.03)
SS-GP	0.46(0.01)	0.55(0.02)	0.69(0.02)	1.04(0.05)
IBP-GP	0.45(0.01)	0.53(0.02)	0.64(0.02)	0.96(0.04)
Multi-GP	0.44(0.00)	0.55(0.01)	0.71(0.01)	1.03(0.02)
VBMC	0.52(0.00)	0.67(0.01)	0.94(0.02)	1.53(0.03)
AR1	0.52(0.01)	0.61(0.01)	0.71(0.01)	0.85(0.01)
Multi-GPDS	**0.390(0.01)**	**0.45(0.01)**	**0.53(0.01)**	**0.71(0.04)**

We present our results in Table 4. We may observe that the Multi-GPDS is able to achieve the best reconstruction accuracy by a margin ranging from 11%-23% compared to the second best performing method.

As for the other methods, the autoregressive model (AR1) performs best in highest missing data ratios, but is outperformed by the IBP-GP for lower ratios. The IBP-GP exhibits consistently better performance than the SS-GP, revealing the benefits of the non-parametric prior being employed. Finally, the Multi-GP is also consistently better than the VBMC in this experiment indicating the importance of the GP prior. Finally, the gradient-based methods, SVT and FPC, achieved similar but considerably worse performance.

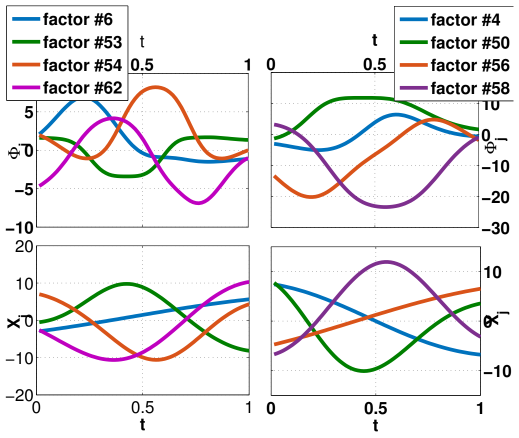
Visual results from this experiment can be seen in the following video: `https://youtu.be/X97PYgJYEbk`.

3.4. Discussion

3.4.1. Model Properties

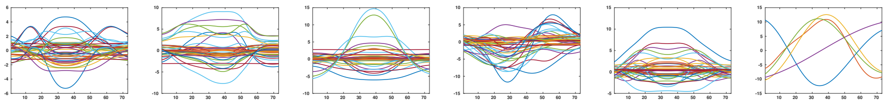
In this section we comment upon the properties of the Multi-GPDS, as they manifest themselves during our empirical evaluation, and investigate the extent to which our model meets its aims.

Temporal Factors.

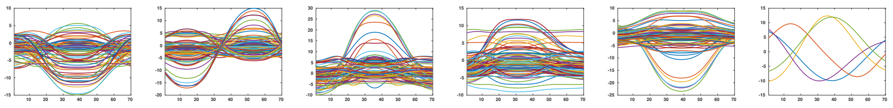


(a) Dance primitives dataset. (b) HDM05 dataset.

Figure 5: Illustration of several dominant posterior Multi-GPDS covariates and factors in time. We can see that the GPDS prior yields highly non-linear factor and is ultimately a significantly more descriptive prior on the temporal factors.



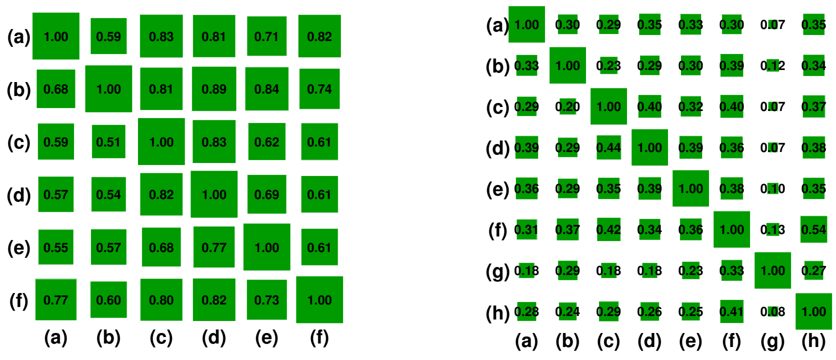
(a) KL-$\boldsymbol{\phi}_3$ (b) KL-$\boldsymbol{\phi}_{22}$ (c) KL-$\boldsymbol{\phi}_{39}$ (d) KL-$\boldsymbol{\phi}_{50}$ (e) KL-$\boldsymbol{\phi}_{52}$ (f) KL-$\boldsymbol{X}$



(g) HDM-$\boldsymbol{\phi}_7$ (h) HDM-$\boldsymbol{\phi}_{24}$ (i) HDM-$\boldsymbol{\phi}_{29}$ (j) HDM-$\boldsymbol{\phi}_{50}$ (k) HDM-$\boldsymbol{\phi}_{58}$ (l) HDM-$\boldsymbol{X}$

Figure 6: Most frequently shared factors between tasks for both datasets. The result presented is from a single run, as the factors are unidentifiable across runs.

The purpose of employing GPDS priors on the model factors was to achieve a non-linear sequential structure with high descriptive power. In order to illustrate the effect of the GPDS prior, we have formed a set consisting of the single most dominant factor per sequence and then randomly chosen 4 factors from this set to be presented in in fig. 5. Finally, we picked the most frequently shared factors between pairs of action to be depicted in fig. 6.



(a) Dance primitives dataset. (b) HDM05 dataset.

Figure 7: Factor sharing confusion matrices. In order to avoid cluttering these two subfigures, we have numbered the action classes in the order they appear in figures 3 and 4.

Table 5: **HDM05 Dataset**: Reconstruction RMSE per action class for the Multi-GPDS trained under a single-task learning (STL) and multi-task learning (MTL) settings.

	50%	**60%**	**70%**	**80%**
	fig. 4a: Both hand rotation backward (BHRB)			
STL	0.63(0.08)	0.81(0.09)	1.09(0.15)	1.22(0.188)
MTL	**0.12(0.01)**	**0.17(0.01)**	**0.24(0.019)**	**0.45(0.07)**
	fig. 4g: Sit-down on chair (SoC)			
STL	0.78(0.04)	0.93(0.05)	1.15(0.10)	**1.63(0.12)**
MTL	**0.49(0.04)**	**0.67(0.04)**	**0.87(0.06)**	**1.61(0.22)**

We observe that the latent covariates $\boldsymbol{X}_j$ form smooth trajectories generally more complex than linear, but are unimodal trajectories in time. The resulting temporal factors $\boldsymbol{\phi}_{ij}$ are considerably more non-linear and complex and in some occasions multi-modal. Their functional form resembles the one achieved by time-varying length-scales.

Factor Sharing and Multi-Task Learning.

As previously mentioned, formulating our model in terms of a dictionary shared among all sequences, allows for multi-task learning and ad-hoc component sharing determined by the magnitude of the factor loading weights.

In this section we attempt to quantify the factor sharing achieved by the

Multi-GPDS, by measuring the mean square error of the absolute factor loadings $\boldsymbol{w}_n$ between tasks. More specifically the factor sharing between task i and task j, denoted as $1/d_{ij}$ is calculated as follows:

$$d_{ij} = \frac{1}{|\mathcal{I}_i|\,|\mathcal{I}_j|} \sum_{n \in \mathcal{I}_i} \sum_{n' \in \mathcal{I}_j} ||\boldsymbol{w}_n| - |\boldsymbol{w}_{n'}|| \qquad (70)$$

where $\mathcal{I}_i$ is the set of all sequence indices belonging to task i and $|\mathcal{I}_i|$ is the cardinality of this set. The similarities are then normalised by dividing with their maximum value and are presented in the form of a confusion matrix in fig. 7.

The factor sharing property the Multi-GPDS, not only leads to smaller, more efficient models able to take advantage of commonalities, but also leads to significant performance gains as may be seen in Table 5. More specifically, we observe an impressive 3-5 times better reconstruction accuracy in task BHRB (fig. 4a) and up to 40% in task SoC (fig. 4g).

Tasks for which we present results were purposefully chosen in order to highlight the fact that those with higher similarity to others (such as BHRB, fig. 7b) have naturally benefitted more by the multi-task learning employed. In contrast, for relatively more unique tasks (such as SoC, fig. 7b) this benefit is less prominent, but is nevertheless still significant.

4. Conclusion

In this work we propose a novel multi-kernel GPDS factor model for rectifying severely damaged sequences. Our aims are threefold: 1) To utilise the redundancy manifesting itself among different demonstrations of the same task, 2) To allow multi-task learning among similar tasks and 3) To take into account the sequential nature of the data. We have devised efficient variational Bayesian inference and have subjected our model to rigorous empirical evaluation on one robotic and two motion capture datasets. We achieve significant improvements in terms of reconstruction accuracy compared to popular methods in the literature. Finally we investigate some interesting properties of our model as they arise from the empirical evaluation.

Acknowledgements

The authors would like to thank Dr. Ayse Kucukyilmaz for her helpful comments and suggestions and Dr. Kyuhwa Lee for kindly providing us the motion capture dataset, along with the rest of the members of the Personal Robotics Lab at Imperial College London for their help and support.

Dr. Korkinof is supported by the State Scholarship Foundation of Greece (IKY) through the EU NSRF 2007-2013. This work was supported in part by the EU FP7 project WYSIWYD under Grant 612139.

[1] K. Altun, B. Barshan, O. Tunçel, Comparative study on classifying human activities with miniature inertial and magnetic sensors, Pattern Recognition 43 (10) (2010) 3605–3620.

[2] P.-C. Chung, C.-D. Liu, A daily behavior enabled hidden markov model for human behavior understanding, Pattern Recognition 41 (5) (2008) 1572–1580.

[3] L. Chen, H. Wei, J. Ferryman, A survey of human motion analysis using depth imagery, Pattern Recognition Letters 34 (15) (2013) 1995–2006.

[4] L. Liu, L. Shao, P. Rockett, Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition, Pattern recognition 46 (7) (2013) 1810–1818.

[5] H. Chen, G. Wang, J.-H. Xue, L. He, A novel hierarchical framework for human action recognition, Pattern Recognition 55 (2016) 148–159.

[6] M. Barnachon, S. Bouakaz, B. Boufama, E. Guillou, Ongoing human action recognition with motion capture, Pattern Recognition 47 (1) (2014) 238–247.

[7] M. Opper, C. Archambeau, The variational gaussian approximation revisited, Neural computation 21 (3) (2009) 786–792.

[8] S. Brooks, A. Gelman, G. Jones, X.-L. Meng, Handbook of Markov Chain Monte Carlo, CRC Press, 2011.

[9] M. Zhou, H. Yang, G. Sapiro, D. B. Dunson, L. Carin, Dependent hierarchical beta process for image interpolation and denoising, in: Int. Conf. on Artificial Intelligence and Itatistics (AISTATS), 2011, pp. 883–891.

[10] J. Paisley, M. Zhou, G. Sapiro, L. Carin, Nonparametric image interpolation and dictionary learning using spatially-dependent dirichlet and beta process priors, in: IEEE Int. Conf. on Image Processing (ICIP), IEEE, 2010, pp. 1869–1872.

[11] R. Ma, N. Barzigar, A. Roozgard, S. Cheng, Decomposition approach for low-rank matrix completion and its applications, IEEE Trans. on Signal Processing 62 (7) (2014) 1671–1683.

[12] A. Cichocki, H. Lee, Y.-D. Kim, S. Choi, Non-negative matrix factorization with α-divergence, Pattern Recognition Letters 29 (9) (2008) 1433–1440.

[13] Y. Liu, L. Jiao, F. Shang, A fast tri-factorization method for low-rank matrix recovery and completion, Pattern Recognition 46 (1) (2013) 163–173.

[14] G. Marjanovic, V. Solo, On lq optimization and matrix completion, IEEE Trans. on Signal Processing 60 (11) (2012) 5714–5724.

[15] E. J. Candès, T. Tao, The power of convex relaxation: Near-optimal matrix completion, IEEE Trans. on Information Theory 56 (5) (2010) 2053–2080.

[16] D. S. Kalogerias, A. P. Petropulu, Matrix completion in colocated mimo radar: Recoverability, bounds & theoretical guarantees, IEEE Trans. on Signal Processing 62 (2) (2014) 309–321.

[17] R. Parhizkar, A. Karbasi, S. Oh, M. Vetterli, Calibration using matrix completion with application to ultrasound tomography, IEEE Trans. on Signal Processing 61 (20) (2013) 4923–4933.

[18] S. D. Babacan, M. Luessi, R. Molina, A. K. Katsaggelos, Sparse bayesian methods for low-rank matrix estimation, IEEE Trans. on Signal Processing 60 (8) (2012) 3964–3977.

[19] M. Zhou, C. Wang, M. Chen, et al., Nonparametric bayesian matrix completion, in: IEEE Sensor Array and Multichannel Signal Processing Workshop, 2010, pp. 213–216.

[20] M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, L. Carin, Nonparametric bayesian dictionary learning for analysis of noisy and incomplete images, IEEE Trans. on Image Processing 21 (1) (2012) 130–144.

[21] J. Sun, D. Parthasarathy, K. Varshney, Collaborative kalman filtering for dynamic matrix factorization, Signal Processing, IEEE Transactions on 62 (14) (2014) 3499–3509. doi:10.1109/TSP.2014.2326618.

[22] G. Chen, F. Wang, C. Zhang, Collaborative filtering using orthogonal nonnegative matrix tri-factorization, Information Processing & Management 45 (3) (2009) 368–379.

[23] C. E. Rasmussen, Gaussian processes for machine learning.

Preprint version; final version available at http://www.sciencedirect.com/science/article/pii/S0031320316304010
Pattern Recognition (2017)
Published by: Elsevier
DOI: 10.1016/j.patcog.2016.12.014

[24] M. K. Titsias, M. Lázaro-Gredilla, Spike and slab variational inference for multi-task and multiple kernel learning., in: Advances in Neural Information Processing Systems (NIPS), Vol. 24, 2011, pp. 2339–2347.

[25] A. Wilson, Z. Ghahramani, D. A. Knowles, Gaussian process regression networks, in: Int. Conf. on Machine Learning (ICML), 2012, pp. 599–606.

[26] J. Luttinen, A. Ilin, Variational gaussian-process factor analysis for modeling spatio-temporal data., in: Advances in Neural Information Processing Systems (NIPS), 2009, pp. 1177–1185.

[27] N. D. Lawrence, Gaussian process latent variable models for visualisation of high dimensional data., in: Advances in Neural Information Processing Systems (NIPS), Vol. 2, 2003, p. 5.

[28] N. Lawrence, Probabilistic non-linear principal component analysis with gaussian process latent variable models, The Journal of Machine Learning Research (JMLR) 6 (2005) 1783–1816.

[29] R. Urtasun, T. Darrell, Discriminative gaussian process latent variable model for classification, in: Int. Conf. on Machine Learning (ICML), ACM, 2007, pp. 927–934.

[30] N. D. Lawrence, A. J. Moore, Hierarchical gaussian process latent variable models, in: Int. Conf. on Machine Learning (ICML), 2007, pp. 481–488.

[31] M. A. Alvarez, D. Luengo, N. D. Lawrence, Latent force models, in: Int. Conf. on Artificial Intelligence and Statistics (AISTATS), 2009, pp. 9–16.

[32] J. Hartikainen, M. Seppänen, S. Särkkä, State-space inference for non-linear latent force models with application to satellite orbit prediction, in: Int. Conf. on Machine Learning (ICML), 2012, pp. 903–910.

[33] M. K. Titsias, N. D. Lawrence, Bayesian gaussian process latent variable model, in: International Conference on Artificial Intelligence and Statistics, 2010, pp. 844–851.

[34] A. C. Damianou, M. K. Titsias, N. D. Lawrence, Variational gaussian process dynamical systems., in: Advances in Neural Information Processing Systems (NIPS), 2011, pp. 2510–2518.

[35] A. Damianou, N. Lawrence, Deep gaussian processes, in: Int. Conf. on Artificial Intelligence and Statistics (AISTATS), 2013, pp. 207–215.

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

[37] C. Vollmer, J. P. Eggert, H.-M. Gross, Modeling human motion trajectories by sparse activation of motion primitives learned from unpartitioned data, in: Advances in Artificial Intelligence, 2012, pp. 168–179.

[38] S. Vijayakumar, A. D'souza, S. Schaal, Incremental online learning in high dimensions, Neural Computation 17 (2) (2005) 2602–2634.

[39] D. Korkinof, Y. Demiris, Online quantum mixture regression for trajectory learning by demonstration, in: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2013, pp. 3222 – 3229.

[40] K. Lee, Y. Su, T.-K. Kim, Y. Demiris, A syntactic approach to robot imitation learning using probabilistic activity grammars, Robotics and Autonomous Systems 61 (12) (2013) 1323–1334.

[41] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, A. Weber, Documentation mocap database hdm05.

[42] J.-F. Cai, E. J. Candès, Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM Journal on Optimization 20 (4) (2010) 1956–1982.

[43] S. Ma, D. Goldfarb, L. Chen, Fixed point and bregman iterative methods for matrix rank minimization, Mathematical Programming 128 (1-2) (2011) 321–353.

[44] T. L. Griffiths, Z. Ghahramani, Infinite latent feature models and the indian buffet process, in: Advances in Neural Information Processing Systems (NIPS), Vol. 18, 2005, pp. 475–482.

[45] K. Lee, T. K. Kim, Y. Demiris, Learning action symbols for hierarchical grammar induction, in: Int. Conf. on Pattern Recognition, 2012, pp. 3778–3782.