Definition and Composition of Motor Primitives using Latent Force Models and Hidden Markov Models.

Student: Diego Alejandro Agudelo España

Supervisor: Mauricio Alexánder Álvarez López

Universidad Tecnológica de Pereira
Master of Science in Electrical Engineering
Grupo de Investigación en Automática
Pereira, Risaralda
2017

Content

Acknowledgements

I would like to thank Mauricio A. Álvarez for introducing me into this exciting area of research, for the insightful meetings and the continued motivation. I also appreciate the help provided by Cristian Guarnizo and Sebastián Gómez with source code and data sets respectively. And I am deeply grateful with my family and friends for their support and encouragement to pursue my goals.

I would like also to thank Universidad Tecnológica de Pereira for its support through the *Jorge Roa Martínez* scholarship for pursuing a master's degree.

This work has been supported by the project *Human-motion Synthesis through Physically-inspired Machine Learning Models* funded by Universidad Tecnológica de Pereira, with code 6-15-3. I am also thankful with Maestría en Ingeniería Eléctrica and the research group in Automática, from Universidad Tecnológica de Pereira, for additional funding.

Abstract

The movement representation problem is at the core of areas such as robot imitation learning and motion synthesis. In these fields, approaches oriented to the definition of motor primitives as basic building blocks of more complex movements have been extensively used because they cope with the high dimensionality and complexity by using a limited set of adjustable primitives. There is also biological evidence supporting the existence of such primitives in vertebrate and invertebrate motor systems.

Traditional methods for representing motor primitives have been purely data-driven or strongly mechanistic. In the former approach new movements are generated using existing movements and these methods are usually very flexible but their extrapolation capacity is limited by the available training data. On the other hand, strongly mechanistic models have a better generalization ability by relying on a physical description of the modeled system, however, it may be hard to fully describe a real system and the resulting differential equations are usually expensive to solve numerically. Therefore, the motor primitive parameterization used in this work is based on a hybrid model which jointly incorporates the flexibility of the data-driven paradigm and the extrapolation capacity of strongly mechanistic models, namely the latent force model framework.

Moreover, the sequential composition of different motor primitives is also addressed using Hidden Markov Models (HMMs) which allows to process movement realizations efficiently. The resulting joint model is an HMM with latent force models (LFMs) as emission process which is an unexplored combined probabilistic model to the best of our knowledge.

1 Problem statement

Biological evidence suggests that movements performed by living beings and, in particular, human motion can be represented as a combination of a limited number of motor primitives [1–3] which are the basic building blocks of more complex movements similarly as human speech is made up of different phonemes. The combination of these motor primitives can arise at different levels and can happen either simultaneously or sequentially [4].

At the behavioral level motor primitives can be seen as mental templates to achieve motor tasks, these mental templates are composed of sub-movements which can not be further decomposed [5]. Similarly, at the muscle level co-activation among several muscles produces synergies which are necessary to produce the required forces and torques to perform a particular movement. At a neural level motor primitives can be represented by assemblies of neurons which activate specific movement patterns [6]. Alternatively, from the point of view of physics the motor primitives can be kinematic [7] when they encode information about movement direction and velocity or they can be dynamic when information about the necessary forces and torques to perform a movement is encoded.

The main concerns about motor primitives are logically how to define them and how to combine them for building complex movements. Having such a variety of levels motor primitives can emerge from, it is difficult to propose a general representation valid for all levels and scenarios [4]. Therefore, different representations and mechanisms for composition have been proposed for particular tasks.

Particularly, in the context of motor learning in robotics motor primitives have been defined relying on the theory of dynamical systems. The dynamical motor primitives (DMPs) make use of the basic nonlinear dynamical systems behaviors represented by second order differential equations to encode a particular movement [8]. The point attractors are used to represent discrete movements and the limit cycles for representing rhythmic movements. DMPs have some interesting properties such as invariance to translation and scaling, they can be easily extended to multiple degrees of freedom and their superposition flexibility can be used to represent more complex behaviors [9]. In Vecchio et al. [5] an alternative formulation of motion primitives was given using the dynamical systems framework and the term *moveme* was used to denote a motor primitive. DMPs have been used extensively in the context of imitation learning [5,8,10]. However, a relevant limitation of these approaches is that the trajectory represented by DMPs is fixed and non-reactive to variations over the environment they were learnt from [11]. Also, the way DMPs are represented does not allow to take advantage of the potential correlation between the different modeled entities (e.g. robot joints).

On the other hand, data-driven approaches are commonly found in the field of artificial motion generation for defining movement primitives [12]. These methods form new motions

based on existing movements and are inherently different from DMPs since they do not assume physical knowledge of the underlying modeled system. Non-parametric regression techniques have been used under the data-driven paradigm [13] usually exploiting correlations between different parts of the moving entity [14]. Similarly, different dimensionality reduction methods such as PCA, Kernel PCA and Isomap have been applied for deriving movement primitives, for a review see Jenkins et al [15]. Motion interpolation approaches also belong to this category and they have relied on concepts like radial basis functions and splines to fit movement primitives [16]. Since data-driven models relies more heavily on data than in prior assumptions there is a limitation when it is required to perform prediction over regions of unseen data, this means that its extrapolation capacity is limited by the available data.

Although the data-driven paradigm does not assume physical knowledge over the modeled system, it usually incorporates some weak prior assumptions through regularization such as smoothness. Therefore, this approach is sometimes called weakly mechanistic [17] and its counterpart, the strongly mechanistic paradigm, includes physical knowledge in the model as DMPs where second order differential equations were used to model dynamics. Having said that, in this work a novel parameterization of motor primitives is proposed relying on a hybrid model which takes advantage of the flexibility associated to the weakly mechanistic approach jointly with the extrapolation capacity of strongly mechanistic models, namely the latent force model (LFM) framework [18].

The proposed solution represents a contribution along the same line of the switched latent force model (SDLFM) proposed in Álvarez et al. [19]. However, the main difference lies in the composition mechanism for different LFMs (i.e. primitives) because in the SDLFM these primitives are articulated via switching points which become hyper-parameters of a Gaussian process covariance matrix. This covariance matrix grows quadratically on the length of the movement time series and as a consequence, the method is unscalable for long realizations of movement. In the current work Hidden Markov Models (HMM) are explored for composing different LFMs which allows to have a fixed length covariance matrix for each primitive and a simpler representation of the sequential dynamics of motor primitives. HMMs have been used extensively in the context of motor primitives [20–22] to combine them either sequentially or simultaneously [23].

2 Justification

The development of a novel motor primitive parameterization can potentially have an impact in areas such as motor skill learning and artificial motor generation. This is supported by the idea that the use of a limited set of modifiable templates seems to be the only way to cope with the high dimensionality nature of movement representation and robot control problems [24] and the use of these primitives tremendously reduces the number of parameters which need to be estimated for a specific movement.

Moreover, by relying in the LFM framework a novel practical solution to the motor primitive representation problem is proposed. In this way, physical knowledge which arises naturally in movement-related problems, is incorporated in a probabilistic data-driven model via second-order differential equations [18]. This parameterization has the following remarkable properties:

- It encodes a distribution over trajectories (i.e. movements).
- The uncertainty over the learnt movements is quantified.
- Correlations between different outputs (i.e. joints) may be exploited [25].
- Many of the desired properties of a motor primitive representation such as co-activation, modulation, coupling and learnability naturally arise as a consequence of the probabilistic formulation [26] .

The proposed model is also pertinent from the computational efficiency perspective since the jointly use of LFMs and HMMs allows to achieve linear time complexity over the observed movement sequences by segmenting the observations in fixed-length sections.

Finally, the proposed model might have potential applications in other fields such as computational biology where physically-inspired methods are useful. For instance, the modeling of the transcription processes in the cell [27, 28] relies on ordinary differential equations (ODE) and it is a biological application which can get benefit and insights from the proposed solution properties.

3 Objectives

3.1 General objective

To develop a probabilistic model for motor primitive representation and composition using Latent Force Models and Hidden Markov Models.

3.2 Specific objectives

1. To provide a formal definition of the probabilistic generative model using latent force models to parameterize the motor primitives and hidden Markov models for describing the sequential combination of such primitives.

2. To develop an estimation algorithm to infer the probabilistic model from data. This implies to estimate the Latent Force Model hyper-parameters and the Hidden Markov Model parameters from movement observations.

3. To validate the performance of the proposed framework as a motor primitive representation over synthetic and real data.

4 Background

In this section we describe the foundations required to introduce the proposed probabilistic model as well as one of the previous motor primitive definitions which motivated this work. Firstly, we present the dynamical movement primitives (DMPs) which can be regarded as a strategy for movement representation widely used in robotics based on dynamical systems. Our motor primitive parameterization can be seen as a particular extension of DMPs as described in the subsection 4.2. Thereafter, we introduce the latent force model framework (LFM) which is used to define a motor primitive based on Gaussian Processes (GPs), we start with a brief review of GPs and their key results. Finally, we end this section introducing the hidden Markov models (HMMs) which will be used to model the sequential composition between movement primitives in section 5.

4.1 Dynamic Motor Primitives (DMPs)

A common formalization used to represent behaviors of motor primitives relying on nonlinear dynamical systems was introduced by Stefan Schaal [29] in 2002. This formulation is based on the following second order dynamical system.

$$\ddot{y} = \alpha_y(\beta_y(g - y) - \dot{y}), \tag{1}$$

where y represents the system's state, α_y and β_y are constants and g is a known goal state. Given that the dynamical system in (1) is linear, convergence to the unique point attractor g can be guaranteed with proper values for α_y and β_y. However, the convergence behavior of (1) is known to have exponential shape and the ability of supporting more complex trajectories for reaching the goal state is required in the context of movement representation. This is achieved through adding a particular forcing term f in (1) as follows.

$$\ddot{y} = \alpha_y(\beta_y(g - y) - \dot{y}) + f. \tag{2}$$

The remaining question of which form assume for the term f is not trivial and arbitrary complex behavior might arise from this election. Therefore, in order to keep things simple and tractable f is defined in terms of a second dynamical system with state variable x and with the same goal state g. This dynamical system is called the canonical dynamical system [9] and its definition is presented in (3).

$$\dot{v} = \alpha_v(\beta_v(g - x) - v), \quad v = \dot{x}. \tag{3}$$

Note that the dynamical system in (3) is not modulated by any forcing term, which guarantees its convergence to the goal state g if α_v and β_v are set properly. Having defined the canonical

dynamical system, the forcing term in (2) is parameterized as

$$f(x, v, g) = \frac{\sum_{i=1}^{N} \psi_i w_i v}{\sum_{i=1}^{N} \psi_i}, \text{ with } \quad \psi_i = \exp\left\{-h_i \left(\frac{x}{g} - c_i\right)^2\right\}, \tag{4}$$

note that $\{\psi_i\}$ denotes a set of N Gaussian basis functions centered at c_i and with a bandwidth equals to h_i. The weights $\{w_i\}$ regulate the overall contribution of the basis functions to the forcing term f and v is referred in the literature as a diminishing term since it vanishes the effect of the forcing term at the end of the movement.

In summary, under the DMP approach the so-called forcing function f is manipulated to reproduce the desired movement following the observed variable y. This forcing term is in turn governed by the set of weights $\{w_i\}$. Therefore, when a movement needs to be reproduced by a DMP, as it is the case in imitation learning [24], the forcing function parameters w_i need to be estimated from data in order to fit the desired trajectory. Having a demonstration $y_{demo}(t), \dot{y}_{demo}(t)$ and $\ddot{y}_{demo}(t)$ (recall that $\dot{y}_{demo}(t)$ and $\ddot{y}_{demo}(t)$ can be obtained from $y_{demo}(t)$ by differentiation) a well-posed supervised learning problem can be formulated for the forcing term f [9]. Namely, we would like to adjust the weights w_i in such a way that f is as close as possible to f_{target},

$$f_{target} = \ddot{y}_{demo} - \alpha_y(\beta_y(g - y_{demo}) - \dot{y}_{demo}), \tag{5}$$

where α_y, β_y were introduced in equation 1 and g is equal to the last observed trajectory value $g = y_{demo}(T)$ for an observation sequence of length T.

Given that the parameterization of f in terms of the basis functions is linear, the aforementioned learning problem can be solved with standard approaches. However, a non-parametric regression technique based on locally weighted learning (RFWR) [30] has been usually used to solve this problem since it allows to automatically determine the number of basis functions N, their centers c_i and bandwidths h_i [29].

4.2 Latent Force Models (LFMs)

We start the LFMs description by highlighting the similarities with DMPs. LFMs are also formulated using dynamical systems and, as we will see in subsection 4.2.2, the dynamical system assumed under the LFM approach is equivalent to the one assumed in DMPs — see equations (2) and (13). However, the parameterization of the forcing term f is now given in terms of a set of latent stochastic processes, instead of the Gaussian basis functions used in equation (4). Those stochastic processes are specifically Gaussian processes (GPs), so we first give an introduction to what GPs are, how to train them and how they are used to make predictions. In the subsection (4.2.2), we look into the LFM assumptions and main results with more detail and we show the kind of prior induced over functions under different settings of the LFM hyperparameters.

4.2.1 Gaussian Processes

A Gaussian process (GP) is defined as a collection of random variables, any finite number of which follows a joint Gaussian distribution [31]. GPs allow to conveniently define a probability distribution directly over functions and perform inference in function space. To denote that the function f is governed by a Gaussian process, we write

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')), \tag{6}$$

where $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ are respectively the mean function and the covariance function which completely specify the GP. Interestingly, any finite set of random variables $\{f(\mathbf{x_i}), f(\mathbf{x_j}), \ldots, f(\mathbf{x_k})\}$ over f is distributed according to

$$\begin{pmatrix} f(\mathbf{x_i}) \\ f(\mathbf{x_j}) \\ \vdots \\ f(\mathbf{x_k}) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(\mathbf{x_i}) \\ m(\mathbf{x_j}) \\ \vdots \\ m(\mathbf{x_k}) \end{pmatrix}, \begin{pmatrix} k(\mathbf{x_i}, \mathbf{x_i}) & k(\mathbf{x_i}, \mathbf{x_j}) & \ldots & k(\mathbf{x_i}, \mathbf{x_k}) \\ k(\mathbf{x_j}, \mathbf{x_i}) & k(\mathbf{x_j}, \mathbf{x_j}) & \ldots & k(\mathbf{x_j}, \mathbf{x_k}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x_k}, \mathbf{x_i}) & k(\mathbf{x_k}, \mathbf{x_j}) & \ldots & k(\mathbf{x_k}, \mathbf{x_k}) \end{pmatrix} \right), \tag{7}$$

as a consequence of the marginal of a GP being a Gaussian distribution.

Having defined a GP prior for f we are now able to sample realizations from it. However, we might be more interested in performing some kind of inference as in the case of computing the posterior over a set of unseen random variables $f(\mathbf{X}^*)$ given a set of observed values $f(\mathbf{X}) = \mathbf{y}$ (i.e. a regression problem). Having $\mathbf{X} \triangleq \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$, $\mathbf{X}^* \triangleq \{\mathbf{x_1^*}, \mathbf{x_2^*}, \ldots, \mathbf{x_M^*}\}$ and assuming a zero-mean GP for f, the joint distribution for $f(\mathbf{X}^*)$ and $f(\mathbf{X})$ is

$$\begin{pmatrix} f(\mathbf{X}) \\ f(\mathbf{X}^*) \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} \mathbf{K}(\mathbf{X}, \mathbf{X}) & \mathbf{K}(\mathbf{X}, \mathbf{X}^*) \\ \mathbf{K}(\mathbf{X}^*, \mathbf{X}) & \mathbf{K}(\mathbf{X}^*, \mathbf{X}^*) \end{pmatrix} \right), \tag{8}$$

where $\mathbf{K}(\mathbf{X}, \mathbf{X}^*)$ denotes the $N \times M$ matrix computed using the covariance function introduced in (6), for all the pairs of observed and unobserved locations. A similar interpretation stands for the other appearances of $\mathbf{K}(.,.)$ with its respective inputs. In order to get the sought posterior we need to include the knowledge about the training data, namely the observed values $\mathbf{y} \triangleq \{y_1, y_2, \ldots, y_N\}$, this is done through the conditioning of the Gaussian distribution on (8) which implies that the posterior for $f(\mathbf{X}^*)$ is also Gaussian

$$\begin{aligned} f(\mathbf{X}^*)|\mathbf{X}, \mathbf{X}^*, \mathbf{y} \sim \mathcal{N}(&\mathbf{K}(\mathbf{X}^*, \mathbf{X})\mathbf{K}(\mathbf{X}, \mathbf{X})^{-1}\mathbf{y}, \\ &\mathbf{K}(\mathbf{X}^*, \mathbf{X}^*) - \mathbf{K}(\mathbf{X}^*, \mathbf{X})\mathbf{K}(\mathbf{X}, \mathbf{X})^{-1}\mathbf{K}(\mathbf{X}, \mathbf{X}^*)). \end{aligned} \tag{9}$$

We have omitted the parameterization of the covariance function so far. Usually a covariance function depends on a set of hyperparameters $\boldsymbol{\theta}$, for instance the squared exponential (SE) covariance function depends on $\boldsymbol{\theta} = \{\sigma, l\}$ as follows

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(\frac{(\mathbf{x} - \mathbf{x}')^2}{2l^2} \right). \tag{10}$$

These hyperparameters are estimated by optimizing the marginal log-likelihood [31] depicted in equation (11).

$$\log p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^\top \mathbf{K}(\mathbf{X},\mathbf{X})^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}(\mathbf{X},\mathbf{X})| - \frac{N}{2}\log 2\pi, \tag{11}$$

where N denotes the dimension of $\mathbf{x}$ and $\mathbf{y}$. Note that the dependence on $\boldsymbol{\theta}$ is introduced through the covariance matrix $\mathbf{K}(\mathbf{X},\mathbf{X})$. This form of estimation is known as *type II maximum likelihood* or *Empirical Bayes*.

4.2.2 Latent Forcing Gaussian Processes

The latent force model framework was introduced in Álvarez et al. [18] motivated by the idea that for some phenomena a weak mechanistic assumption underlies a data-driven model. Therefore, it represents a hybrid model with elements of both, weakly and strongly mechanistic models [17]. The mechanistic assumptions are incorporated using dynamical systems governed by differential equations. Particularly, consider the following second order dynamical system similar to the one presented in (1) but using Leibniz's notation this time.

$$\frac{d^2 y_d(t)}{dt} + C_d \frac{dy_d(t)}{dt} + B_d y_d(t) = 0. \tag{12}$$

C_d and B_d are known as the damper and spring coefficients respectively and $y_d(t)$ is the model's output of interest. Following the same DMP philosophy, the behavior encoded by (12) given values for constants C_d and B_d is well-known and rather limited. The key is to keep the model flexible enough even under circumstances where the mechanistic assumptions are not rigorous fulfilled [17] to fit arbitrary trajectories. Again, this flexibility is achieved by adding a forcing term but, unlike the DMP forcing term in (2), this is governed by a set of latent functions $u_q(t)$ as follows.

$$\frac{d^2 y_d(t)}{dt} + C_d \frac{dy_d(t)}{dt} + B_d y_d(t) = \sum_{q=1}^{Q} S_{d,q} u_q(t). \tag{13}$$

The constants $S_{d,q}$ are known as the sensitivities and they regulate the contribution of the latent forces $u_q(t)$ to the dynamics. Note that the fact of having multiple outputs governed by the same set of latent functions induces correlations between those outputs which can be of great benefit when motor primitives are used to model the movement of a set of robot joints. Also, the possibility of having a larger number of outputs than latent forces allows to use the LFM approach as a dimensionality reduction mechanism [19].

The main difference of the LFM approach in contrast with the classical DMP is that it assumes Gaussian process priors with SE covariance function, which was introduced in

equation (10), over the latent forcing functions, or formally $u_q(t) \sim \mathcal{GP}(0, k_{SE}(t,t'))$. As a consequence of this assumption, it turns out that outputs are jointly governed by a Gaussian Process and its covariance function can be derived analytically. In fact, the cross-covariance between outputs $y_d(t)$ and $y_{d'}(t)$ under the dynamical model of equation (13) can be expressed as

$$k_{y_d,y_{d'}}(t,t') = \sum_{q=1}^{Q} l_q \sqrt{\pi} \frac{S_{d,q} S_{d',q}}{8\omega_d \omega_{d'}} k^{(q)}_{y_d,y_{d'}}(t,t'), \tag{14}$$

where $\omega_d = \sqrt{4B_d - C_d^2}/2$ and $k^{(q)}_{y_d,y_{d'}}(t,t')$ represents the covariance between outputs $y_d(t)$ and $y_{d'}(t)$ under the effect of the latent force $u_q(t)$ and its exact form is given by [18]

$$\begin{aligned} k^{(q)}_{y_d,y_{d'}}(t,t') = h_q(\hat{\gamma_{d'}}, \gamma_d, t, t') + h_q(\gamma_d, \hat{\gamma_{d'}}, t', t) + h_q(\gamma_{d'}, \hat{\gamma_d}, t, t') + h_q(\hat{\gamma_d}, \gamma_{d'}, t', t) \\ -h_q(\hat{\gamma_{d'}}, \hat{\gamma_d}, t, t') - h_q(\hat{\gamma_d}, \hat{\gamma_{d'}}, t', t) - h_q(\gamma_{d'}, \gamma_d, t, t') - h_q(\gamma_d, \gamma_{d'}, t', t), \end{aligned} \tag{15}$$

where $\gamma_d = C_d/2 + j\omega_d$ and $\hat{\gamma}_d = C_d/2 - j\omega_d$ and

$$h_q(\gamma_{d'}, \gamma_d, t, t') = \frac{\Upsilon_q(\gamma_{d'}, t', t) - \exp(-\gamma_d t)\Upsilon_q(\gamma_{d'}, t', 0)}{\gamma_d + \gamma_{d'}}, \tag{16}$$

with

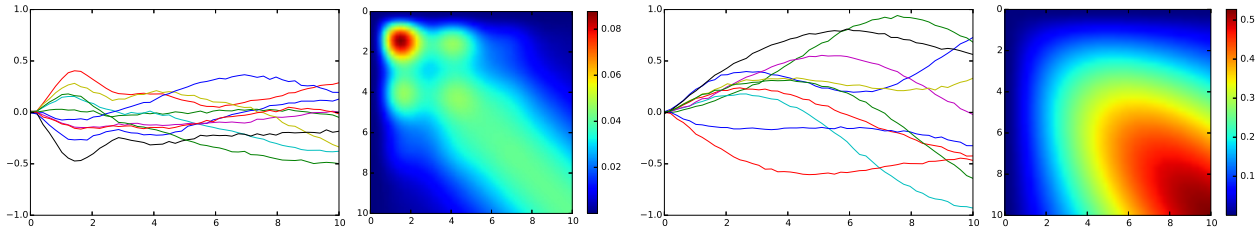
$$\begin{aligned} \Upsilon_q(\gamma_{d'}, t, t') =& 2\exp\left(\frac{l_q^2\gamma_{d'}^2}{4}\right)\exp(-\gamma_{d'}(t-t')) - \exp\left(\frac{-(t-t')^2}{l_q^2}\right)\mathrm{w}(jz_{d',q}(t,t')) \\ &- \exp\left(-\frac{(t')^2}{l_q^2}\right)\exp(-\gamma_{d'}t)\mathrm{w}(-jz_{d',q}(0,t')), \end{aligned} \tag{17}$$

and

$$z_{d',q}(t,t') = \frac{(t-t')}{l_q} - \frac{l_q\gamma_{d'}}{2}. \tag{18}$$

In the above expression l_q denotes the length-scale associated with the *q-th* latent force and $\mathrm{w}(jz)$ denotes the Faddeeva's function defined as $\mathrm{w}(jz) = \exp(z^2)\mathrm{erfc}(z)$ where erfc denotes the complex version of the complementary error function. Notice that, in general, $z_{d',q}(t,t') \in \mathbb{C}$.

Under the LFM approach, the inference of the model from data is performed by a two-stage mechanism following the standard techniques in Gaussian process regression. Firstly, the differential equation parameters, which are now hyperparameters of the second order LFM covariance function introduced in equation (14), are estimated by optimizing the logarithm of the marginal likelihood in (11). Then, the training data is included in the model through the conditioning of the observed random variables which leave us with the GP predictive



(a) (b) (c) (d)

Figure 1: Sampled realizations and resulting covariances for 2 different LFMs. (a) Trajectories drawn from an LFM governed by underdamped dynamics, in (b) the resulting covariance matrix for the sampled points is shown. Similarly, (c) and (d) depict the LFM behavior for an overdamped system instead.

distribution presented in (9). Remarkably, the inference for LFMs in the multiple-output setting is not fundamentally different if the observed and unobserved outputs are stacked as in (8) and the covariance matrices are computed taking into account the cross covariance between outputs.

A LFM is fundamentally a Gaussian process with a particular covariance function which has a physical inspiration associated to dynamical systems. Therefore, sampling from an LFM is equivalent to sampling from a GP with the covariance function introduced in (14). Figure 1 shows realizations sampled for 2 LFMs with different dynamical properties.

Recall that the convergence behavior for a second order system as the one shown in equation (12) is determined by the damping ratio $\eta = \frac{1}{2}C_d/\sqrt{B_d}$. A system with $\eta < 1$ is referred to as an underdamped system and it converges to the point attractor with a decaying oscillatory behavior. In figure 1a some underdamped trajectories were generated setting $\{C_d = 1, B_d = 5\}$ in equation (13). On the other hand, an overdamped system has $\eta > 1$ and its convergence behavior is not oscillatory. A set of overdamped realizations is shown in figure 1c using $\{C_d = 5, B_d = 1\}$ in the second order system. In both cases (i.e. overdamped and underdamped) the realizations were sampled assuming a single latent force with lengthscale $l = 2$ and sensitivity $S_{1,1} = 1$. Note how different the covariance patterns are for different hyperparameter values.

4.3 Hidden Markov Models (HMMs)

The Hidden Markov Models are a family of probabilistic models useful for describing sequential data. When modeling this kind of data one aims to find patterns and insights about the sequential dynamics of the process of interest which is impossible when the observed sequential data is assumed to be a realization of an independent and identically distributed (i.i.d.) process.

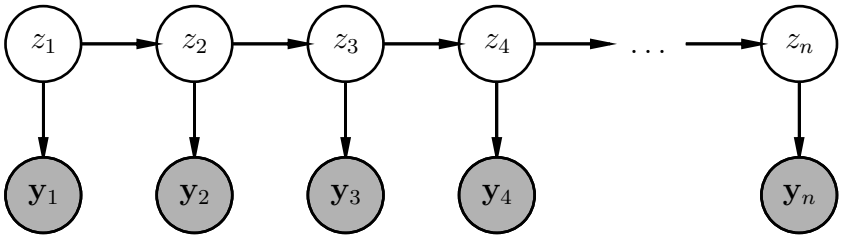


Figure 2: Probabilistic graphical model for HMMs. The unshaded nodes denote hidden random variables whereas the shaded nodes represent observable random vectors.

An HMM is governed by two different processes, a measurement process—also known as emission process—associating an observation $\mathbf{y}_n$ with its corresponding latent variable z_n and a Markovian process for describing the state variable dynamics, this is, the dynamics between the hidden state variables $(z_1, z_2, \ldots, z_n)$. Usually a first-order Markov process is assumed over the state variables. The graphical model for HMMs is depicted in figure 2.

Formally, an HMM models a sequence of observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ by assuming that the observation at index i (i.e. $\mathbf{y}_i$) was produced by a k-valued discrete hidden state z_i and that the sequence of hidden states $\mathbf{Z} = (z_1, z_2, \ldots, z_n)$ was produced by a first-order Markov process. Therefore, the complete-data likelihood for a sequence of length n can be written as:

$$p(\mathbf{Y}, \mathbf{Z}) = p(z_1)p(\mathbf{y}_1|z_1)\prod_{i=2}^{n} p(z_i|z_{i-1})p(\mathbf{y}_i|z_i). \tag{19}$$

Consequently, the marginal likelihood over the observations $\mathbf{Y}$ can be expressed as

$$p(\mathbf{Y}) = \sum_{\mathbf{Z}} p(\mathbf{Y}, \mathbf{Z}). \tag{20}$$

Remarkably, an evidence of the flexibility of HMMs is that the Markovian property does not hold for the observed variables $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ at any order [32].

A common parameterization of an HMM is given by (21)

$$\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\} \quad \text{with} \quad \Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}. \tag{21}$$

Where $\mathbf{A} = \{a_{j,j'}\}$ denotes the hidden state transition matrix, $\boldsymbol{\pi} = \{\pi_j\}$ is the initial hidden state probability mass function and Θ represents the set of emission parameters for each hidden state. The meaning for $a_{j,j'}$ and π_j is

$$p(z_i = j'|z_{i-1} = j) = a_{j,j'}.$$
$$p(z_1 = j) = \pi_j.$$

Making explicit this parameterization in the expression for the complete-data likelihood (19) we have

$$p(\mathbf{Y}, \mathbf{Z}|\zeta) = p(z_1|\boldsymbol{\pi})p(\mathbf{y}_1|z_1, \theta_{z_1}) \prod_{i=2}^{n} p(z_i|z_{i-1}, \mathbf{A})p(\mathbf{y}_i|z_i, \theta_{z_i}). \tag{22}$$

This completes the probabilistic definition of HMMs. We can now formulate a variety of problems about (22) such as how to estimate the model parameters ζ from the observations $\mathbf{Y}$. A comprehensive review of these problems for standard emission processes and their corresponding solutions can be found in [33]. However, the use of novel probabilistic models as observation process broadens the horizon of potential applications and brings new challenges from the perspective of inference. The HMM inference algorithms adapted to the proposed emission process will be covered in section 6.

5 The Model

HMMs have been previously used to model the combination of motor primitives. In Chiappa et al [21] a probabilistic model was proposed to represent movement as a concatenation of noisy transformations of a set of hidden trajectories (motor primitives), these hidden trajectories were in turn modeled with linear Markovian dynamics. This model offers the advantage of estimating autonomously the transition points between hidden trajectories.

HMMs have also been used to model motor primitives under a completely different perspective. Usually motor primitives have been associated with non-overlapping segments of movement trajectories but a different approach was presented by Williams et al [22] where the movement performed to write a handwritten character is modeled as a superposition of concurrent motor primitives using the factorial Hidden Markov model (fHHM) [23] to model each motor primitive as a factor.

In this work HMMs are used differently by introducing LFMs as emission process to represent the motor primitives, which is an unexplored extension to the best of our knowledge. We also introduce the autoregressive HMM in order to guarantee the continuity of the movement representation and for this reason, the autoregressive version of the model is the one used in the experiments described in section 7.

5.1 Hidden Markov Models and Latent Force Models

The proposed model is based on the idea that movement time-series can be represented by a sequence of non-overlapping latent force models. This is motivated by the fact

that movement realizations have some discrete and non-smooth changes on the forces and dynamics which govern the movements. These changes can not be modeled by a single LFM because it is essentially a GP, and as such it generates smooth trajectories, and similarly its hyperparameters—which account for the dynamics as explained in 4.2.2—are fixed.

Moreover, the use of HMM and LFM emissions enables us to capture the existing diversity and redundancy in the dynamics over a movement trajectory. Under this approach the whole trajectory is explained by a limited set of hidden primitives which account for different dynamic regimes. In contrast, the switched dynamical latent force model (SDLFM) [19] assumes a single dynamic regime for a given observation and just models non-smooth changes over the latent forces.

From the scalability point of view, the proposed model offers the advantage of keeping the size of the covariance matrices belonging to different hidden states (LFMs) fixed regardless of the observed trajectory length—which is a drawback for the SDLFM since it uses a single covariance matrix which grows quadratically with the observation sequence length. This is achieved at the expense of constraining the switching points where changes over the latent forces and dynamic regimes occur.

Formally, the overall system is modeled as an HMM where the emission distribution for each hidden state is represented by an LFM. Therefore the complete-data likelihood still fulfills the equation in (22) but the emission process is performed as follows

$$p(\mathbf{y}_i|z_i,\theta_{z_i},\boldsymbol{\chi}) = \mathcal{N}(f_i(\boldsymbol{\chi}), I\sigma^2_{z_i}), \tag{23}$$

$$f_i(x) \sim \mathcal{GP}(0, k_{LFM}(x,x';\theta_{z_i})), \tag{24}$$

where k_{LFM} represents the second order LFM kernel already defined in equation (14) and $\sigma^2_{z_i}$ is the variance of the i.i.d. output Gaussian noise. Notice that the HMM framework allows the observable variables to have a different dimensionality with respect to the latent variables, in this case $\mathbf{y}_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,s})$ is a continuous multivariate vector whereas z_i is univariate and discrete. It should also be noticed that there is an additional variable $\boldsymbol{\chi}$ conditioning the emission process, this variable denotes the set of sample locations $\boldsymbol{\chi} = \{\chi_1, \chi_2, \ldots, \chi_s\}$ where the LFMs are sampled at. Note that this set is independent of the hidden variable values. In consequence, a particular movement sequence represented by the HMM with LFM emission is segmented into equal-length chunks of size $|\boldsymbol{\chi}| = s$, each of which is modeled by a particular LFM.

For the sake of illustration, let's assume that $\sigma^2_{z_i} = 0$, which lead us to $\mathbf{y}_i = (f_i(\chi_1), f_i(\chi_2), \ldots, f_i(\chi_s))$. Using the product rule over the elements of $\mathbf{y}_i$ the emission distribution in (23) can also be written as

$$p(\mathbf{y}_i|z_i,\theta_{z_i},\boldsymbol{\chi}) = \prod_{j=1}^{s} p(f_i(\chi_j)|f_i(\chi_1),\ldots,f_i(\chi_{j-1})). \tag{25}$$

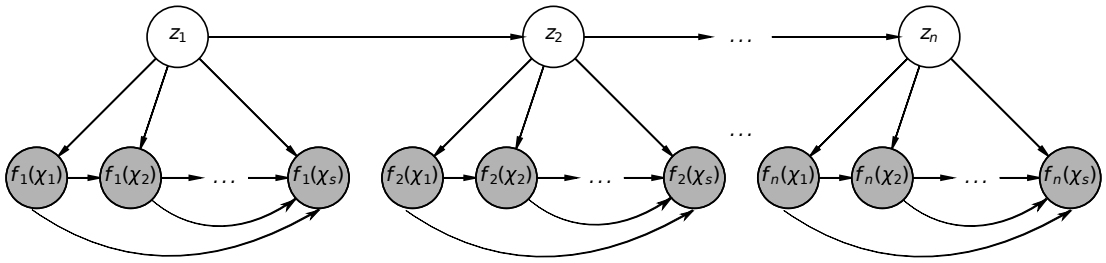


Figure 3: Probabilistic graphical model for the HMM with LFM emission. The sequence of functions $(f_1, f_2, \ldots, f_n)$ is drawn from the LFMs associated to the hidden-state sequence $(z_1, z_2, \ldots, z_n)$. The set $\boldsymbol{\chi} = \{\chi_1, \chi_2, \ldots, \chi_n\}$ denotes the sample locations.

In this way the dependence between hidden variables and the observed variables at different sample locations is explicit and can be better visualized in the probabilistic graphical model depicted in figure 3.

Note that the resulting graph resembles the structure of a Hidden semi-Markov Model (HSMM), which is like an HMM except that each hidden state is allowed to emit a sequence of observations [34]. However, since the sequence's length of emitted observations is fixed for each hidden state, we can still rely on the classical HMM formulation by assuming that each hidden state emits a single multivariate observation. By using an HSMM we can handle variable-length observation sequences at the expense of increasing the computational complexity of the algorithms used to perform inference described in section 6.

Notice that, as a result of the model's formulation on equations (22), (23) and (24), the trajectories that can be generated by the model are not necessarily continuous realizations. This occurs because the observations are assumed to be conditionally independent given the hidden states and therefore, there is not explicit time correlation between successive emissions (i.e. observed segments). Taking into account that movement time series are inherently continuous, this behavior represents a limitation which needs to be addressed in order to build a more accurate model.

We can overcome the aforementioned issue by using an Autoregressive Hidden Markov Model (AHMM) instead. In an AHMM, dependence relations between successive observations can be handled explicitly. In figure 4 a typical graphical model for an AHMM is depicted. Notice that, there are arcs between every pair of successive observations, but this is not a requirement, and higher order dependencies between observed random variables can be added. The extension only affects the computation of the local evidence and the algorithms for estimating the model's parameters remain largely unchanged as we will see in section 6.3.

In order to support the AHMM, the emission process defined in equation (23) needs to be extended to be dependent not only on the corresponding hidden state but on the last

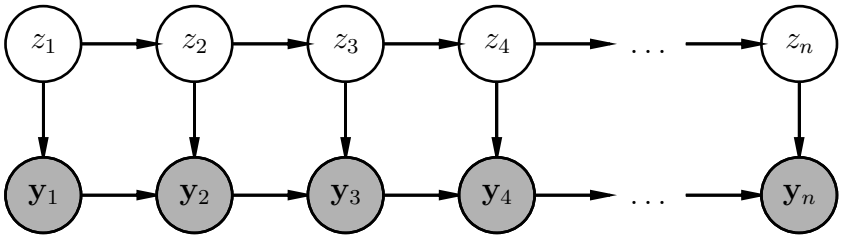


Figure 4: Probabilistic graphical model for an Autoregressive HMM.

observation as well, as it is shown in equations (26) and (27). Particularly, there are two major changes made over the emission process. Firstly, the mean governing the sampled function $f_i(.)$ is not longer zero, now it is equal to $y_{i-1,s}$, which is the last value of the last observation $\mathbf{y}_{i-1}$. Furthermore, in (27), the first value of each observation $f_i(\chi_0)$ is conditioned to be equal to the last value of the last observation $y_{i-1,s}$. Both extensions support the generation of continuous trajectories. The reason why the conditioning by itself is not enough to guarantee continuity and the shift in the mean is also necessary is related with the implicit assumption that the initial conditions for the system in equation (13) are zero, in other words, conditioning the first value of each observation in a value different to its mean is problematic from a numerical point of view given the assumption over the initial conditions. A sampled realization from the extended model can be seen at the bottom of figure 5.

$$p(\mathbf{y}_i|z_i, \mathbf{y}_{i-1}, \theta_{z_i}, \boldsymbol{\chi}) = \mathcal{N}(f_i(\boldsymbol{\chi}), I\sigma^2_{z_i}), \tag{26}$$

$$f_i(x) \sim \mathcal{GP}(y_{i-1,s}, k_{LFM}(x, x'; \theta_{z_i})), \quad f_i(\chi_0) = y_{i-1,s}. \tag{27}$$

In figure 5 there is also a pictorial representation of the proposed HMM model with LFM emission. Notice that each movement segment is governed by a particular LFM which is represented as a spring-damper system, at the same time there is a sequential hidden evolution of the active spring-damper system at a particular segment. This figure is essentially the same graphical model depicted in figure 2 for state-space models augmented with the particular choice of emission process.

6 Algorithms

Having defined precisely what an HMM is from a probabilistic point of view in section (4.3), it is important to consider some relevant problems which need to be addressed in order to

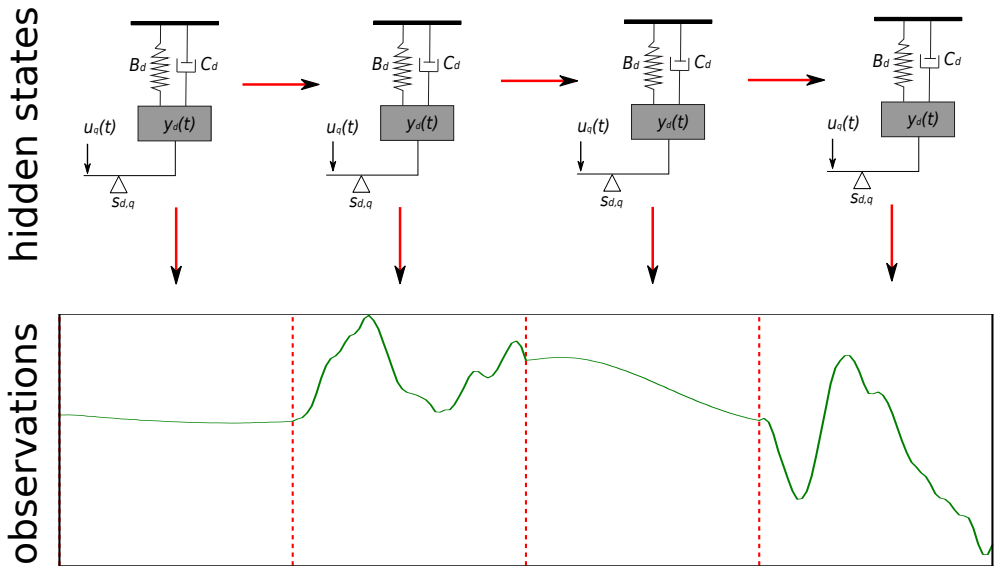


Figure 5: HMM with LFM emissions. Top: sequential evolution of different LFMs represented as spring-damper systems. Bottom: Movement observation sequence as a concatenation of LFM realizations.

use an HMM in a real setting as in the motor primitives case. These problems can be also be considered as inference problems since unobserved quantities are inferred from the observed ones. Formally, Given an observation sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ and a particular model $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$, the main challenges are [33]:

1. How to efficiently compute the observation likelihood given the model, $p(\mathbf{Y}|\zeta)$?
2. How do we choose a sequence of hidden states which maximizes $p(\mathbf{Z}|\mathbf{Y}, \zeta)$?
3. How are the model parameters estimated in such a way that $p(\mathbf{Y}|\zeta)$ is maximized?

The described problems are well known and their solutions for particular choices of emission process and for some variations of the classical HMM architecture depicted on figure 2 have been proposed [33]. Specifically, the Forward-Backward algorithm was proposed as a solution for the first problem, the Viterbi algorithm for the second one and the Baum-Welch algorithm for the third problem. In this chapter we will see that the standard algorithms for solving the aforementioned problems remain largely unchanged by setting LFMs as observation model. Therefore, we give a brief description of the algorithms for the first and second problem, and a more detailed analysis of the third challenge is presented since it is where the major changes occur due to the novel observation model.

6.1 Observation Likelihood Computation

Algorithm 1: Forward algorithm

Input : Model parameters $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$.
Observation sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$.

Output: A matrix α where $\alpha[t][i]$ denotes the joint probability $p(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t, z_t = i)$.

```
for i = 1, ..., K do
    α[1][i] = p(y_1 | z_1 = i) * π(i)
for t = 2, ..., N do
    for j = 1, ..., K do
        α[t][j] = 0
        for i = 1, ..., K do
            α[t][j] = α[t][j] + α[t − 1][i] * A(i, j)
        α[t][j] = α[t][j] * p(y_t | y_{t−1}, z_t = j)
```

Let $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ be an observation sequence and $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$ a particular model under consideration. In order to compute the observation likelihood $p(\mathbf{Y}|\zeta)$ efficiently, we will use the forward-backward [33] algorithm with some minor modifications to support the autoregressive extension. To achieve this we define the functions $\alpha(z_n)$ and $\beta(z_n)$ as follows. Note that throughout section 6 we will omit the explicit parametrization on ζ to keep the notation uncluttered.

$$\alpha(z_n) \equiv p(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n, z_n). \tag{28}$$

$$\beta(z_n) \equiv p(\mathbf{y}_{n+1}, \mathbf{y}_{n+2}, \ldots, \mathbf{y}_N | z_n, \mathbf{y}_n). \tag{29}$$

Using the product rule and the conditionally independence assumptions made in an AHMM the following recurrence relations for $\alpha(z_n)$ and $\beta(z_n)$ can be derived [32] with their respective base cases.

$$\alpha(z_n) = p(\mathbf{y}_n|\mathbf{y}_{n-1}, z_n) \sum_{z_{n-1}} \alpha(z_{n-1}) p(z_n|z_{n-1}) \quad \text{with} \quad \alpha(z_1) = p(\mathbf{y}_1|z_1)\boldsymbol{\pi}(z_1), \tag{30}$$

$$\beta(z_n) = \sum_{z_{n+1}} \beta(z_{n+1}) p(\mathbf{y}_{n+1}|\mathbf{y}_n, z_{n+1}) p(z_{n+1}|z_n) \quad \text{with} \quad \beta(z_N) = 1. \tag{31}$$

We are now able to formulate the observation likelihood computation in terms of either, $\alpha(.)$ or $\beta(.)$, as follows,

$$p(\mathbf{Y}) = \sum_{z_N} \alpha(z_N) = \sum_{z_1} \beta(z_1) p(\mathbf{y}_1|z_1)\boldsymbol{\pi}(z_1). \tag{32}$$

Remarkably, the introduced recursions allow us to compute the observation likelihood in linear time complexity with respect to the observation's sequence length. This is achieved using a dynamic-programming strategy for computing iteratively the α recursion known as the forward algorithm — See algorithm No. 1 — and a similar one for the β recursion known as the backward algorithm. The forward-backward procedure is also useful in the task of computing the posterior distributions of the hidden variables given the observations, $p(z_n|\mathbf{Y})$, as we will see in section 6.3. It is important to point out that, in order to actually implement the forward-backward algorithm, it is necessary to use scaled versions of $\alpha(z_n)$ and $\beta(z_n)$ because of floating point underflow issues [32].

6.2 Most Probable Hidden State Sequence

Algorithm 2: Viterbi algorithm

Input : Model parameters $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$.
Observation sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$.
Output: A sequence of hidden states $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$ which maximizes $p(\mathbf{Z}|\mathbf{Y}, \zeta)$.

for $i = 1, \ldots, K$ **do**
  $\delta[1][i] = p(\mathbf{y}_1|z_1 = i) * \boldsymbol{\pi}(i)$ ▷ Meaning of δ in equation (34)
for $t = 2, \ldots, N$ **do**
  for $j = 1, \ldots, K$ **do**
    $\delta[t][j] = 0$
    for $i = 1, \ldots, K$ **do**
      if $\delta[t][j] < \delta[t-1][j] * \boldsymbol{A}(i, j)$ **then**
        $\delta[t][j] = \delta[t-1][j] * \mathbf{A}(i, j)$
        $\psi[t][j] = i$
    $\delta[t][j] = \delta[t][j] * p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t = j)$
/* Path backtracking */
$z_N = \arg\max_i \delta[N][i]$
for $t = N - 1, \ldots, 1$ **do**
  $z_t = \psi[t+1][z_{t+1}]$

Consider the problem of finding the sequence of motor primitives which governs a given movement realization. Taking into account the model formulation presented in section 5.1, this problem can be rephrased as finding the sequence of hidden states $\mathbf{Z} = (z_1, z_2, \ldots, z_N)$ from which the sequence of observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$ was emitted. Keep in mind that, as a consequence of the probabilistic formulation, a distribution is induced over such sequence. Then, a possible way to answer the stated question is to find the sequence of states

associated with the largest probability mass — a MAP estimate of $p(\mathbf{Z}|\mathbf{Y}, \zeta)$.

$$\arg\max_{\mathbf{Z}} p(Z|\mathbf{Y}, \zeta) = \arg\max_{\mathbf{Z}} \frac{p(\mathbf{Z}, \mathbf{Y}|\zeta)}{p(\mathbf{Y}|\zeta)} = \arg\max_{\mathbf{Z}} p(\mathbf{Z}, \mathbf{Y}|\zeta). \tag{33}$$

In order to maximize (33) we can do better than an exhaustive search over $\mathbf{Z}$ by following a similar idea from the α recursion computation in section 6.1. Defining $\delta_n(i)$ to be the probability associated with the most probable hidden-state sequence which explains the first n observations and whose last hidden state is i,

$$\delta_n(i) = \max_{z_1, z_2, \ldots, z_{n-1}} p(z_1, z_2, \ldots, z_n = i, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n), \tag{34}$$

the recurrence relation in (35) can be derived [33] for $\delta_n(i)$, and it can be implemented efficiently using dynamic programming to get an $O(K^2N)$ algorithm known as the Viterbi algorithm — See algorithm No. 2. Note that the Viterbi algorithm resembles the forward algorithm (algorithm No. 1) but the summation over the last hidden variable is turned into a maximization operation. A new variable ψ is also introduced to keep track of the sequence of states which maximizes the expression in (35) at every step n. Thus, the final answer is reconstructed via a backward traversal over ψ.

$$\delta_n(i) = \left\{ \max_j \quad \delta_{n-1}(j) p(z_n = i | z_{n-1} = j) \right\} p(\mathbf{y}_n | \mathbf{y}_{n-1}, z_n = i). \tag{35}$$

Recall that the sequence given by the Viterbi algorithm is optimal in a global sense, at the sequence level, rather than at the single state level. Therefore, the group of states which individually are the most probable based on the observations does not necessarily correspond with the most probable sequence of hidden states [33].

6.3 Parameter Estimation

The problem of estimating the model parameters $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$ given a sequence of observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$ is addressed in this section. A commonly used approach for training an HMM consists in choosing the parameters ζ in such a way that $p(\mathbf{Y}|\zeta)$ is maximized. This is known as a maximum-likelihood estimate for ζ.

It turns out that the form of $p(\mathbf{Y}|\zeta)$ (equation (32)) can not be directly maximized in a closed-form way. We then rely on the Expectation-Maximization (EM) algorithm [35] to estimate ζ, which is an iterative procedure used for parameter estimation of probabilistic models with latent variables. The EM algorithm alternates between two steps after which the estimate is updated. In the E step, the posterior distribution for the latent variable $p(\mathbf{Z}|\mathbf{Y}, \zeta^{old})$ is computed using the current estimate of parameters ζ^{old}. In the M step, the

expectation of the log-likelihood of the complete data $p(\mathbf{Z}, \mathbf{Y}|\zeta^{new})$ is computed with respect to the posterior $p(\mathbf{Z}|\mathbf{Y}, \zeta^{old})$, next this expected value is maximized with respect to ζ^{new} to obtain a new estimate for ζ. The two-step algorithm is repeated until some suitable convergence criteria is reached.

To start with the EM derivation for our model we need to introduce the *1-of-K* encoding for the hidden random variables $\{z_i\}$ to ease the math. Under this notation, each variable z_i is represented as a binary vector having a single one in the position denoted by its value. Thus, $z_i = j$ means that $z_{i,j} = 1$ when it is *1-of-K* encoded. Rewriting the probabilities involving the hidden variables in terms of the introduced encoding we get,

$$p(z_1|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{z_{1,k}}, \tag{36}$$

$$p(\mathbf{y}_i|\mathbf{y}_{i-1}, z_i) = \prod_{k=1}^{K} p(\mathbf{y}_i|\mathbf{y}_{i-1}, \theta_k)^{z_{i,k}}, \tag{37}$$

$$p(z_i|z_{i-1}, \mathbf{A}) = \prod_{k=1}^{K}\prod_{j=1}^{K} a_{j,k}^{z_{i,k}z_{i-1,j}}. \tag{38}$$

E step

In this stage of the EM algorithm we focus on how to compute the posterior of hidden variables given the observations and a particular model, or more formally $p(\mathbf{Z}|\mathbf{Y}, \zeta^{old})$. With regard to HMMs, this step reduces to compute the expectations $\langle z_{i,k}\rangle$ and $\langle z_{i,k}z_{i-1,j}\rangle$ since these are the only expectations required to perform the M step. To illustrate how to compute them, recall that using the binary random variables properties we can rewrite such expectations as,

$$\langle z_{n,k}\rangle = p(z_n = k|\mathbf{Y}, \zeta^{old}). \tag{39}$$

$$\langle z_{n-1,i}z_{n,j}\rangle = p(z_n = j, z_{n-1} = i|\mathbf{Y}, \zeta^{old}). \tag{40}$$

It proves to be the case that these posteriors can be computed from the α and β recursions introduced in (30) and (31). This means that the E step can be executed in linear time with respect to the number of observations with a single run of the forward-backward algorithm described in section 6.1. Using the Bayes theorem jointly with the conditional independence properties for HMMs, $\langle z_{n,k}\rangle$ and $\langle z_{n-1,i}z_{n,j}\rangle$ can be written in terms of $\alpha(z_n)$ and $\beta(z_n)$ —which were defined in (28) and (29)— as it is shown in (41) and (42) [32]. Notice that the conditioning on ζ^{old} is implicit for the sake of readability.

$$\langle z_{n,k}\rangle = \frac{\alpha(z_n = k)\beta(z_n = k)}{\sum_{l=1}^{K}\alpha(z_N = l)}. \tag{41}$$

$$\langle z_{n-1,i} z_{n,j} \rangle = \frac{\alpha(z_{n-1} = i)\, a_{i,j}\, p(\mathbf{y}_n | \mathbf{y}_{n-1}, z_n) \beta(z_n = j)}{\sum_{l=1}^{K} \alpha(z_N = l)}. \tag{42}$$

M step

Let $Q(\zeta, \zeta^{old})$ denote the expectation of the log-likelihood of the complete data. Using (36), (37) and (38) in the expression for the complete-data likelihood for HMMs (equation (22)), and applying the linearity of expectation we have,

$$\begin{aligned} Q(\zeta, \zeta^{old}) &= \sum_{\mathbf{Z}} \ln p(\mathbf{Y}, \mathbf{Z}|\zeta) p(\mathbf{Z}|\mathbf{Y}, \zeta^{old}) \\ &= \langle \ln p(\mathbf{Y}, \mathbf{Z}|\zeta) \rangle \\ &= \left\langle \sum_{k=1}^{K} z_{1,k} \ln \pi_k + \sum_{i=2}^{N} \sum_{k=1}^{K} \sum_{j=1}^{K} z_{i,k} z_{i-1,j} \ln a_{j,k} + \sum_{i=1}^{N} \sum_{k=1}^{K} z_{i,k} \ln p(\mathbf{y}_i | \mathbf{y}_{i-1}, \theta_k) \right\rangle \\ &= \sum_{k=1}^{K} \langle z_{1,k} \rangle \ln \pi_k + \sum_{i=2}^{N} \sum_{k=1}^{K} \sum_{j=1}^{K} \langle z_{i,k} z_{i-1,j} \rangle \ln a_{j,k} + \sum_{i=1}^{N} \sum_{k=1}^{K} \langle z_{i,k} \rangle \ln p(\mathbf{y}_i | \mathbf{y}_{i-1}, \theta_k), \end{aligned} \tag{43}$$

where the $\langle . \rangle$ operator denotes expectation with respect to the distribution $p(\mathbf{Z}|\mathbf{Y}, \zeta^{old})$. Observe that the expectations involving hidden variables are only $\langle z_{i,k} \rangle$ and $\langle z_{i,k} z_{i-1,j} \rangle$. The way those expectations are computed is described in the E step, but in the M step they are assumed as constants.

The M step is complete when we find values for $\zeta = \{\mathbf{A}, \boldsymbol{\pi}, \Theta\}$ that maximize the function $Q(\zeta, \zeta^{old})$. A useful fact is that the parameters $\mathbf{A}$, $\boldsymbol{\pi}$ and Θ are not coupled in the optimization task, and we can therefore optimize them independently by picking the proper term for each parameter in (43). Remarkably, the terms to optimize for the set of parameters $\{\mathbf{A}, \boldsymbol{\pi}\}$ are independent of the particular form of the emission process $p(\mathbf{y}_i | \mathbf{y}_{i-1}, z_i)$. As a result, the M-step update equations for the parameters associated to the hidden dynamics $\{\mathbf{A}, \boldsymbol{\pi}\}$ are unchanged by the use of LFMs as emission process.

It can be shown that $Q(\zeta, \zeta^{old})$ is maximized with respect to the parameters accounting for the hidden dynamics $\{\mathbf{A}, \boldsymbol{\pi}\}$ when we update them as follows [33],

$$\pi_k = \frac{\langle z_{1,k} \rangle}{\sum_{j=1}^{K} \langle z_{1,j} \rangle} = \langle z_{1,k} \rangle \tag{44}$$

$$a_{j,k} = \frac{\sum_{i=2}^{N} \langle z_{i-1,j} z_{i,k} \rangle}{\sum_{l=1}^{K} \sum_{i=2}^{N} \langle z_{i-1,j} z_{i,l} \rangle} = \frac{\sum_{i=2}^{N} \langle z_{i-1,j} z_{i,k} \rangle}{\sum_{i=2}^{N} \langle z_{i-1,j} \rangle} \tag{45}$$

In order to update the emission process parameters $\Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}$, only the last term

of equation (43) must be taken into account, namely

$$Q(\zeta, \zeta^{old})_{\text{emission term}} = \sum_{k=1}^{K}\sum_{i=1}^{N}\langle z_{i,k}\rangle \ln p(\mathbf{y}_i|\mathbf{y}_{i-1}, \theta_k). \tag{46}$$

This term is basically a weighted sum of Gaussian log-likelihood functions, each of which

Algorithm 3: Objective function for estimating the emission parameters

Input : Emission parameters $\Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}$.
Observation sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$.
A matrix γ where $\gamma[t][i]$ denotes denotes $\langle z_{t,i}\rangle$

Output: A real number ν equals to the emission term in (46).

```
for k = 1, ..., K do
    for n = 1, ..., N do
        ν = ν + γ[n][k] * ln p(y_n | y_{n-1}, θ_k)        ▷ See equation (47)
```

has the form already shown in section 4.2.1 — equation (11). Re-writing that result for the proposed emission process we have

$$\ln p(\mathbf{y}_i|\mathbf{y}_{i-1}, \theta_k) = -\frac{1}{2}(\mathbf{y}_i - \boldsymbol{\mu})^T\mathbf{K}_{\theta_k}^{-1}(\mathbf{y}_i - \boldsymbol{\mu}) - \frac{1}{2}\ln|\mathbf{K}_{\theta_k}| - \frac{S}{2}\ln 2\pi, \tag{47}$$

where $\mathbf{K}_{\theta_k} = \mathbf{K}(\boldsymbol{\chi}, \boldsymbol{\chi}; \theta_k) + I\sigma_k^2$ denotes the kernel matrix computed using the LFM covariance function over the sample locations $\boldsymbol{\chi}$ added with the diagonal matrix accounting for the output noise variance. $\boldsymbol{\mu}$ represents the process mean which is set to be constant and equal to the last value of the last observation $\boldsymbol{\mu} = y_{i-1,s}\mathbf{1}$ (where $\mathbf{1}$ denotes the all-one vector). Recall that S stands for the dimensionality of $\boldsymbol{\chi}$ and $\mathbf{y}_i$.

The estimation of the LFM emission parameters Θ (also known as hyperparameters in the GPs literature) is performed through the optimization of the stated emission term and gradient-based methods can be used for this. It can be shown that the derivatives of (46) with respect to the kernel hyperparameters in Θ can be computed according to (48) and (49) [31]

$$\frac{\partial Q(\zeta, \zeta^{old})}{\partial \theta_j} = \sum_{i=1}^{N}\langle z_{i,j}\rangle \frac{\partial \ln p(\mathbf{y}_i|\mathbf{y}_{i-1}, \theta_j)}{\partial \theta_j}. \tag{48}$$

$$\frac{\partial \ln p(\mathbf{y}_i|\mathbf{y}_{i-1}, \theta_j)}{\partial \theta_j} = \frac{1}{2}\operatorname{Tr}\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \mathbf{K}_{\theta_j}^{-1})\frac{\partial \mathbf{K}_{\theta_j}}{\partial \theta_j}\right) \text{ with } \boldsymbol{\alpha} = \mathbf{K}_{\theta_j}^{-1}(\mathbf{y}_i - \boldsymbol{\mu}). \tag{49}$$

Note that the above result holds for any kind of covariance function (kernel) we choose. The use of the LFM covariance function introduced in equation (14) affects the factor $\partial \mathbf{K}_{\theta_j}/\partial\theta_j$ depending on the particular hyperparameter for which we are computing the gradient. To be specific, for the LFM kernel we need to take derivatives with respect to the following hyperparameters: The damper constants $\{C_d\}$, the spring constants $\{B_d\}$, the sensitivities $\{S_{q,d}\}$, the latent forces lengthscales $\{l_q\}$ and the output noise variance σ^2. Recall that we have a set of these hyperparameters for each hidden state in our model — See section 4.2.2 for details about the meaning of these hyperparameters.

As it can be seen from (15), the expression for the LFM covariance function is rather involved and so its gradients, which are not derived in this document. In the experiments reported in section 7 the numerical approximation to the gradient was used as an alternative to the symbolic derivation and it proved to work well for learning the emission parameters. The objective function for estimating these emission parameters is sketched in the algorithm No. 3.

After the E step and the M step are executed together, it is guaranteed that the likelihood with respect to the parameters does not decrease. Therefore, we will eventually stop at a local maximum ζ^*. This form of estimation applied to HMMs is known as the Baum-Welch algorithm [33], and an implementation of this procedure for an autoregressive HMM with LFM emission is outlined in the algorithm No. 4.

Algorithm 4: Baum-Welch algorithm for an AHMM with LFM emission

Input : Observation sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N)$.
Output: Estimated model parameters $\zeta^* = \{\mathbf{A}^*, \boldsymbol{\pi}^*, \Theta^*\}$

Initialize the model parameters $\zeta^* = \{\mathbf{A}^*, \boldsymbol{\pi}^*, \Theta^*\}$
$\rho_{old} = 0$
while *True* **do**
  $\alpha =$ `ForwardAlgorithm`$(\mathbf{Y}, \zeta^*)$ ▷ See Algorithm 1
  $\beta =$ `BackwardAlgorithm`$(\mathbf{Y}, \zeta^*)$
  $\rho = 0$ ▷ Let ρ be $p(\mathbf{Y}|\zeta^*)$
  for $i = 1, \ldots, K$ **do**
    $\rho = \rho + \alpha[N][i]$
  if $|\rho - \rho_{old}| < \epsilon$ **then**
    return $\zeta^* = \{\mathbf{A}^*, \boldsymbol{\pi}^*, \Theta^*\}$ ▷ ϵ is set to be a sensible threshold value
  $\rho_{old} = \rho$
  `/* E step */`
  for $t = 1, \ldots, N$ **do**
    for $i = 1, \ldots, K$ **do**
      $\gamma[t][i] = (\alpha[t][i] * \beta[t][i])/\rho$ ▷ Let $\gamma[t][i]$ denotes $\langle z_{t,i} \rangle$
  for $t = 1, \ldots, N-1$ **do**
    for $i = 1, \ldots, K$ **do**
      for $j = 1, \ldots, K$ **do**
        $\xi[t][i][j] = \mathbf{A}(i,j) * p(\mathbf{y}_{t+1}|\mathbf{y}_t, z_{t+1} = j)$ ▷ $\xi[t][i][j]$ represents $\langle z_{t,i} z_{t+1,j} \rangle$
        $\xi[t][i][j] = (\xi[t][i][j] * \alpha[t][i] * \beta[t+1][j])/\rho$
  `/* M step */`
  for $i = 1, \ldots, K$ **do**
    $\boldsymbol{\pi}^*(i) = \gamma[1][i]$
  for $i = 1, \ldots, K$ **do**
    for $j = 1, \ldots, K$ **do**
      $n, d = 0$
      for $t = 1, \ldots, N-1$ **do**
        $n = \xi[t][i][j] + n$
        $d = \gamma[t][i] + d$
      $\mathbf{A}^*(i,j) = n/d$
  $\Theta^* =$ `GradientBasedOptimizer`$(\texttt{EmissionObjective}, \Theta^*, \gamma, \mathbf{Y})$ ▷ See algorithm 3

7 Experiments

In this section the capability of the proposed model for learning motor primitives and identifying them over unseen trajectories is evaluated. In the first part this validation is done using synthetic trajectories, which were produced using the joint model as a probabilistic generative model. On section 7.2, the model is evaluated over two real data sets, namely the CMU-MOCAP database and a robot data set recorded from multiple striking movements in a table tennis game. We also report a preliminary experiment using the model in a classification task to demonstrate its potential in this area.

7.1 Synthetic data

A particular instance of the model was used for sampling 20 trajectories with 20 segments each. This data-set was divided in two parts. One half was used for inferring the model (i.e. learning the motor primitives) and the rest was used for validation (i.e. motor primitive identification). Recall that, in order to produce continuous trajectories, the synthetic trajectories were generated in such a way that the constant mean of each segment is equal to the last value of the last segment — See equation (27). To generate the synthetic trajectories, we consider an HMM with three hidden states with transition matrix $\mathbf{A}$ and initial state probability mass function $\boldsymbol{\pi}$. The same parameters inferred from training data are shown alongside (i.e. $\mathbf{A}^*$ and $\boldsymbol{\pi}^*$).

$$\mathbf{A} = \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.3 & 0.2 & 0.5 \end{bmatrix}, \boldsymbol{\pi} = \begin{bmatrix} 0.1 \\ 0.3 \\ 0.6 \end{bmatrix}, \qquad \mathbf{A}^* = \begin{bmatrix} 0.83 & 0.08 & 0.09 \\ 0.63 & 0.27 & 0.1 \\ 0.27 & 0.25 & 0.48 \end{bmatrix}, \boldsymbol{\pi}^* = \begin{bmatrix} 0.0 \\ 0.4 \\ 0.6 \end{bmatrix}.$$

Regarding the emission process, a LFM with 4 outputs ($D = 4$), a single latent force ($Q = 1$) and sample locations set $\boldsymbol{\chi} = \{0.1, \ldots, 5.1\}$ with $|\boldsymbol{\chi}| = 20$ was chosen. The sensitivities were fixed to one and they were not estimated. The actual emission parameters and the corresponding inferred values are depicted in table 1, where the spring and damper constants are indexed by the output index.

The inference of the model from the training synthetic data set was carried out using the algorithm described in section 6.3. Notably, the validation of the inferred model can not be done by merely looking to the estimated parameters and comparing them to the actual values since they are not, in general, identifiable parameters. For instance, the hidden states might be arbitrarily permuted after the inference process. However, the estimated parameters are shown here in such a way that there is a one-to-one correspondence between hidden states before and after the inference to ease comparison.

	Hidden State 1	Hidden State 2	Hidden State 3
Spring const.	{3., 1, 2.5, 10.}	{1., 3.5, 9.0, 5.0}	{5., 8., 4.5, 1.}
Spring const.*	{3.21, 1.05, 2.67, 11.09}	{0.67, 1.72, 3.39, 2.26}	{6.92, 10.66, 6.32, 2.3}
Damper const.	{1., 3., 7.5, 10.}	{3., 10., 0.5, 0.1}	{6., 5., 4., 9.}
Damper const.*	{1.2, 3.36, 8.39, 9.61}	{0.5, 2.62, 1.07, 0.27}	{6.09, 5.54, 4.47, 9.76}
Lengthscale	10	2	5
Lengthscale*	85.77	180.48	159.96

Table 1: LFM emission parameters.

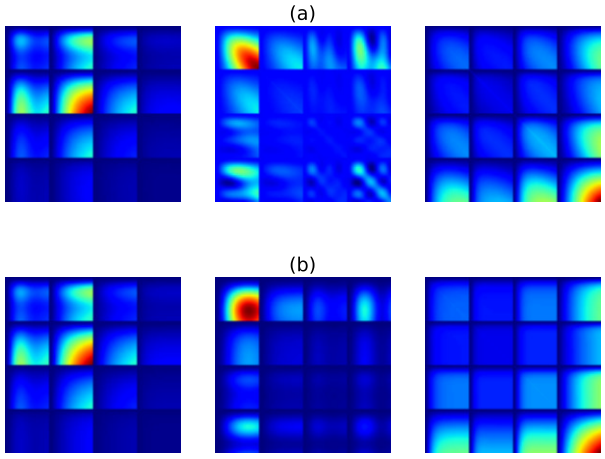


Figure 6: Actual and inferred toy experiment covariance matrices for the 3 hidden states. Top row **(a)**: LFM covariance matrices used for generating the synthetic trajectories. Bottom row**(b)**: Estimated LFM covariance matrices.

It can be argued that there is a similar pattern with respect to the original emission values, particularly for hidden states 1 and 3. The hidden state 2 exhibits a more diverse pattern in comparison to its original parameters. Nevertheless, when the inferred LFM covariances are plotted (see figure 6) for each hidden state, it is easy to see that the underlying correlations between outputs were successfully captured by the inferred emission process. The poor lengthscale estimation in table 1 can be explained by the fact that the range covered by the sample locations set $\boldsymbol{\chi}$ (i.e. from 0.1 to 5.1) is reduced for this property to be noticeable and inferable.

A sample synthetic trajectory can be seen in figure 7. This is a trajectory used to validate the model's capability to detect motor primitives (i.e. hidden states) given the inferred model parameters, which is achieved using the Viterbi algorithm [33] — See algorithm No.

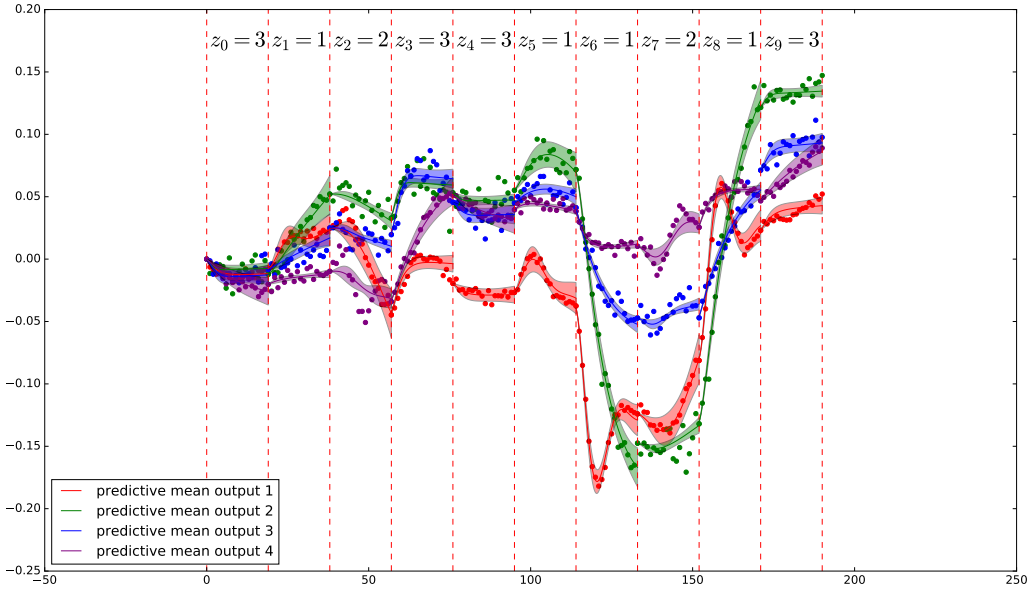


Figure 7: Primitives identification over a synthetic trajectory. In the top, the most probable hidden state sequence $[z_0, z_1, \ldots, z_9]$ given by the inferred model is shown. The predictive mean after conditioning over the hidden state sequence and observations is also depicted with error bars accounting for two standard deviations. The x-axis denotes the number of samples.

2. The resulting hidden state sequence is shown on the top of figure 7, and it turned out to be exactly the same sequence used for generation, which was assumed to be unknown. Moreover, the predictive mean of $p(\mathbf{x}_i|z_i, \mathbf{x}_{i-1}, \theta_{z_i}, \boldsymbol{\chi})$ (equation 26) after conditioning over the most probable hidden state sequence and the observed multiple-output trajectory is also shown with the corresponding error bars.

The Viterbi algorithm was executed for each validation trajectory and the resulting hidden state sequences were compared against the actual values used during the data-set generation. The correct hidden state was recovered with a success rate of **95**% failing only in **10** out of **200** validation segments.

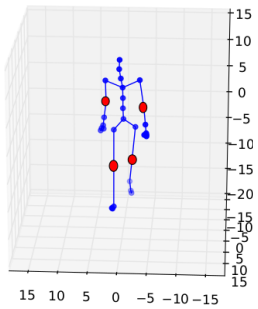


Figure 8: Chosen MOCAP joints for real data experiment

7.2 Real data

7.2.1 CMU-MOCAP

The real-data experiment consists in inferring a set of motor primitives from a group of realizations of the same particular behavior and assessing whether or not, a similar pattern is recovered over the inferred activated motor primitives for unseen realizations of the same behavior. To achieve this, the CMU motion capture database (CMU-MOCAP) is used. The chosen behavior is the walking action because it exhibits a rich redundancy over dynamics as a consequence of the cyclic nature of gait. Specifically, the subject No. 7 was used with trials $\{01, 02, 03, 06, 07, 08, 09\}$ for training and trials $\{10, 11\}$ for validation. In order to take advantage of the multiple-output nature of LFMs a subset of four joints was selected for the validation. The chosen joints are both elbows and both knees since they are relevant for the walking behavior and their potential correlations might be exploited by the model. In figure 8 the selected joints, whose angle variations over time are used in this experiment, are highlighted with red markers.

The model formulation given in section 5 implies fixing some model parameters a priori such as the number of hidden primitives, the number of latent forces and the sample locations set $\boldsymbol{\chi}$ used in the emission process in equation 26. Using a number of hidden states equals to three was enough for capturing the dynamics of the chosen behavior. Experiments with a higher number of hidden states were carried out with equally good results but the local variations associated to each observation started to be captured by the model thanks to the higher number of available hidden states. Similarly, the use of a high number of latent forces makes the model more flexible from the point of view of regression at the expense of favoring overfitting. By using three latent forces a good trade-off is obtained between the regression capability and the intended behavior generalization at the motor primitive level. Finally, the sample locations set $\boldsymbol{\chi}$ was defined to cover the interval $[0.1, 5.1]$ with 20 sample locations equally spaced (i.e. $|\boldsymbol{\chi}| = 20$). This choice was motivated by the functional division of a gait cycle into eight phases [36]: initial contact, loading response, mid stance, terminal

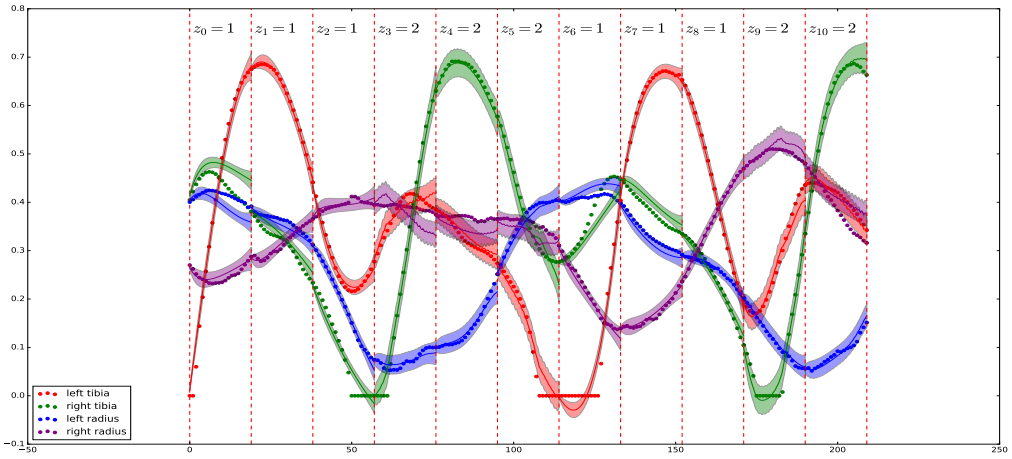


Figure 9: Primitives identification over a real walking trajectory. In the top the most probable hidden state sequence $[z_0, z_1, \ldots, z_{10}]$ given by the inferred model is shown. The predictive mean with error bars is depicted for the four joints. The x-axis denotes the number of samples (frame rate = 120).

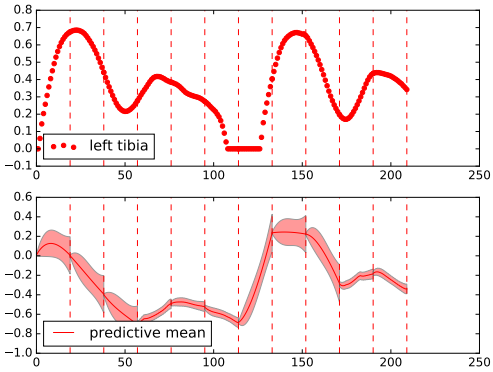
stance, pre-swing, initial swing, mid swing, and terminal swing. Having $|\boldsymbol{\chi}|= 20$ a complete gait cycle over the selected observations is made up of roughly seven segments which is in accordance with the functional subdivision given that the initial contact can be considered an instantaneous event.

The resulting Viterbi sequences over training and validation observations are shown in table 2. From this table a cyclic sequential pattern over the identified motor primitives can be observed with a period of seven segments as expected. The first three segments of a gait cycle correspond to activations of the hidden state one, and the remaining gait cycle segments are generally explained by the hidden state two, although the last cycle segment exhibits high variability. Remarkably, the discussed pattern was also obtained over the unseen trajectories (i.e. No. 10, 11) which suggests that the sequential dynamics of the motor primitives associated to the walking behavior of subject seven were learned by the proposed model.

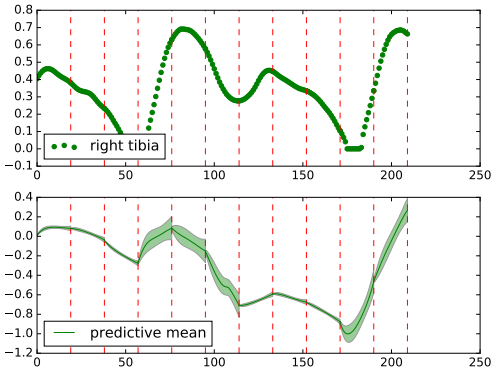
To illustrate how the time series corresponding to the chosen joints look like, in figure 9 the identified motor primitives over the realization No. 10 are shown along the resulting fit after conditioning over the most probable hidden state sequence (Viterbi sequence). An additional experiment was carried out with the aim of assessing the extrapolation capacity of the proposed approach. This capability is inherent to the LFMs given that they are hybrid models incorporating elements from the strongly-mechanistic approach to modeling [18]. In this experiment we also intended to test the model's ability to perform transfer learning over

Trial Number	**Motor primitives identified**
01	1, 1, 1, 2, 2, 2, 1 | 1, 1, 1, 2, 2, 2, 1 | 1, 1
02	1, 1, 1, 2, 2, 2, 1 | 1, 1, 1, 2, 2, 2, 1 | 1, 1
03	1, 1, 1, 2, 2, 2, 2 | 1, 1, 1, 2, 2, 2, 2 | 2, 1, 1, 1, 2, 2, 2
06	1, 1, 1, 2, 2, 2, 2 | 1, 1, 1, 2, 2, 2, 2 | 1, 1
07	1, 1, 1, 2, 2, 2, 2 | 1, 1, 1, 2, 2, 2, 2 | 2, 1, 1, 1, 2
08	1, 1, 1, 2, 2, 2, 2 | 2, 1, 1, 1, 1, 2
09	1, 1, 1, 2, 1, 2, 1 | 1, 1, 2, 2
10*	1, 1, 1, 2, 2, 2, 1 | 1, 1, 2, 2
11*	1, 1, 1, 2, 2, 2, 2 | 1, 1, 1, 2, 2, 2, 2 | 2, 1, 1, 1, 2

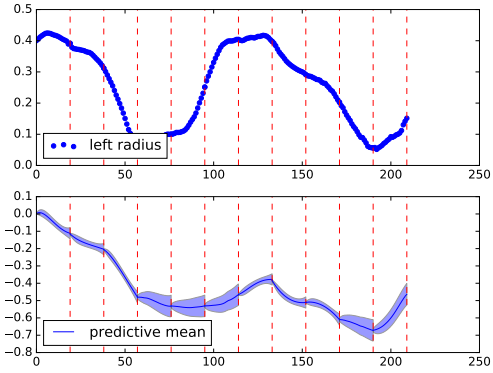
Table 2: Motor primitives identified over walking realizations.



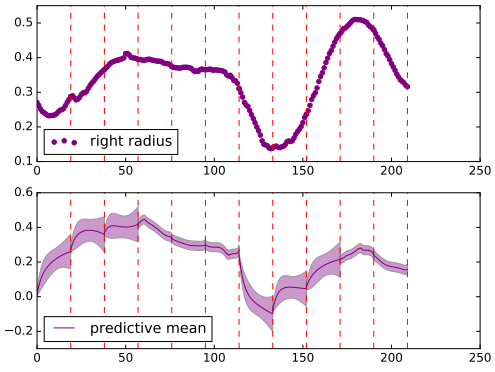
(a) left tibia.



(b) right tibia.



(c) left radius.



(d) right radius.

Figure 10: Missing output experiment for each of the chosen joints. For each joint the plot on top shows the actual observations. The predicted output based on the other observed outputs (i.e. the predicted output is assumed entirely unobserved) is shown on the bottom for each joint. The shaded area for predictions denotes ± 2 standard deviations. The x-axis for all plots denotes the number of samples (frame rate = 120).

different outputs.

Regarding the experiment setting, we used the validation human-walking realization depicted in figure 9 corresponding to the time evolution of both elbow joints and both knee joints from the CMU-MOCAP subject No. 7. After learning the model from the training realizations of the last experiment, we assumed one of the outputs of the validation realization as missing and we tried to predict it only using the other three observed outputs as input of the trained model. The model structure (i.e. the number of latent forces, the number of hidden states and the sample locations set) was set in exactly the same way used in the first real-data experiment described in the second paragraph of section 7.2.

The resulting prediction assuming each output as unobserved is shown in figure 10. Remarkably, the increasing and decreasing tendencies for each joint were learned by the model which is worthy of attention taking into account that the modeled output was not measured at any sample location. However, looking carefully to the magnitude of the predictions it is clear that the scales of the original outputs were not correctly recovered, and this makes any error-based measure prohibitively large. One of the reasons for this may be related with the autoregressive condition in equation (27) which makes the GP mean of every segment dependent on the last segment, as a result the prediction errors in each segment propagate to all upcoming segments if there is not observed data to alleviate this effect. Additionally, under the current formulation of the model, the GP mean of the first segment is not being learned and therefore it's always assumed to be zero which is an inaccurate assumption.

Classification of MOCAP behaviors: LFMs have been used successfully to recognize different human actions [37] in a setting where the learned LFM hyperparameters are feed into a discriminative classifier. With this experiment we pretend to extend the previous work by taking into account the sequential evolution of the LFM hyperparameters in the classification setting, this information is available thanks to the joint HMM-LFM formulation.

The model definition given in section 5 for the motor primitives provides a way of defining a probability distribution over movement realizations. Interestingly, this fact allows us to take our model to the classification domain and validate it as a probabilistic generative model assessing how well it can represent different behaviors for their discrimination. Recall that, in the generative approach to classification, we estimate the posterior probabilities $p(C_k|\mathbf{Y})$ via the class-conditional densities $p(\mathbf{Y}|C_k)$ using Bayes theorem, where $\mathbf{Y}$ represents the observed instance we want to classify and C_k denotes the different categories in the task. The assigned category is then computed using decision theory based on the posterior probabilities $p(C_k|\mathbf{Y})$.

In this experiment we aim at distinguishing between two classes, namely the walking behavior (C_1) and the running behavior (C_2). As input $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$ we have a realization

		Outcome		
		C_1	C_2	Total
Actual value	C_1	11	1	12
	C_2	0	5	5
	Total	11	6	17

Table 3: Confusion matrix for the classification experiment between the walking and the running behavior.

of either behavior. As we did in section 7.2.1, the behaviors are characterized by the time series corresponding to both elbows and both knees. In contrast with section 7.2.1, we take the MOCAP realizations of subject 35 because it has multiple realizations of both behaviors. Trials $\{2, 3, 4, 5, 6\}$ are used for training an instance of the proposed model for C_1, trials $\{17, 18, 19, 20, 21\}$ represent the training set for the model instance of behavior C_2. The rest of the walking and running realizations are used for validation (12 realizations for C_1 and 5 for C_2). The model structure is set similarly as we did in section 7.2.1 (number of latent forces = number of hidden states = 3, $|\chi| = 20$). The distance between consecutive sample locations was fixed to the inverse of the MOCAP frame rate.

A particular movement trajectory $\mathbf{Y}$ was assigned to the behavior C_k with the largest $p(\mathbf{Y}|C_k)$ since the prior $p(C_k)$ was assumed to be uniform. The resulting confusion matrix after training both models and classifying the validation movements is shown in the table No. 3. We got an accuracy of 0.94 and only one validation instance of the walking behavior was misclassified.

The obtained results unveils the potential application of the proposed model in a different problem such as classification. We therefore suggest the possibility of exploring this domain as an alternative to extend this work—See section 8.

7.2.2 Robot Data - Table Tennis Striking Movements

In this experiment we analyze the set of inferred motor primitives for multiple striking movements performed by a robot in a table tennis game. The data was recorded with a haptic input device (Barrett WAM arm) while a human teacher was demonstrating how to perform movements for hitting a table tennis ball by taking the robot arm with his own hands—Figure 11 shows the setting used to record the data set. Remarkably, this kind of experiments can benefit fields such as robot imitation learning and human movement analysis by bringing new insights.

One instance of our model was trained from 7 recorded outputs corresponding to the different degrees of freedom of the robot arm (shoulder yaw, shoulder pitch, shoulder roll, elbow joint,

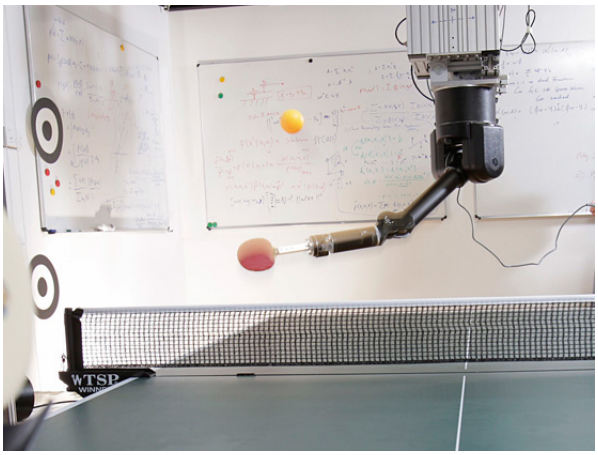


Figure 11: Robot setting for capturing striking movements in a table tennis game[1].

wrist yaw, wrist pitch, wrist roll). The size of the sample location set χ was set to 20 samples as we did in the MOCAP experiment in section 7.2.1, and we chose the distance between consecutive samples to be equal to the inverse of the robot recording frequency (0.2 seconds due to the subsampling). The number of available hidden states was 5 which proved to be enough since only 4 hidden states were actually used by the Viterbi algorithm. Regarding the number of latent forces, we carried out this experiment with 3 of them due mainly to computational reasons because for each new latent force $O(D \times K)$ new hyperparameters must be added making the training time prohibitively long for a large number of latent forces.

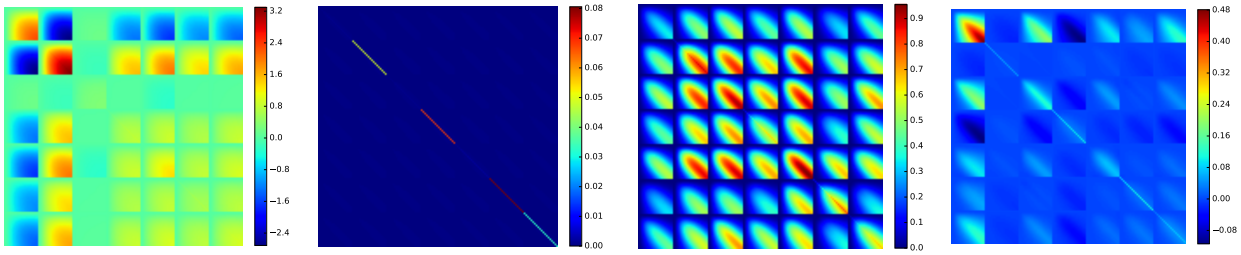
The resulting most likely hidden state sequence (Viterbi sequence) inferred by the model for the striking movements realization is the following

$$\begin{aligned}[\mathbf{2},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{2},\mathbf{2},\mathbf{4},\mathbf{0},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{1},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{0},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\\ \mathbf{4},\mathbf{2},\mathbf{4},\mathbf{4},\mathbf{0},\mathbf{2},\mathbf{4},\mathbf{2},\mathbf{4},\mathbf{2},\mathbf{4},\mathbf{2},\mathbf{2},\mathbf{2},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4},\mathbf{4}].\end{aligned} \tag{50}$$

In figure (13) the recorded outputs are shown colored according to the Viterbi sequence shown above, which means that every hidden state is associated with a different color. Moreover, In figure 12 the joint covariance pattern for the 7 outputs can be observed for each of the hidden states except for the hidden state number 3 which was excluded of the Viterbi sequence in (50).

We are now able to analyze which covariance pattern governed each of the segments in the striking realizations in order to gain insights about the nature of this particular movement. One remarkable fact is that the hidden state number 4 (the blue one) was the most common in the movement, governing about 2/3 of the entire movement. Looking into this covariance pattern in figure 12d we see that the strongest correlations were captured relative to the first output (shoulder yaw), in other words, the covariances across the first output row and first

[1]Source: https://ei.is.tuebingen.mpg.de/research_groups/robot-learning



(a) (b) (c) (d)

Figure 12: Inferred multiple-output covariances associated with the different hidden states for the striking movements. (a) joint covariance of hidden state 0, (b) joint covariance of hidden state 1, (c) joint covariance of hidden state 2, (d) joint covariance of hidden state 4. The hidden state 3 is omitted since it does not appear in the obtained Viterbi sequence (50).

output column. This is consistent with the fact that the shoulder yaw is the main degree of freedom for lateral displacements of the arm, which is in turn crucial for striking the ball. Marginally, the shoulder yaw has the highest variance in the hidden state 4 which is again explainable from the large lateral displacements. Another interesting observation from this covariance pattern, which can also be derived by inspection of figure 13, is that outputs 1 and 3 are highly proportional whereas outputs 1 and 4 have, in general, inverse proportionality. Take into account that a striking movement while the robot arm is turned to the right (or left), in waiting position, implies a reduction in the shoulder yaw angle (output 1) and an extension of the elbow joint (output 4), which might explain the inverse proportionality between these outputs. The second most common hidden state is the number 2 whose joint covariance matrix can be observed in figure 12c. The main property of this covariance pattern is that it encodes a direct proportionality between all the outputs involved in the movement. This is particularly true for the outputs 2,3 and 5 (shoulder pitch, shoulder roll and wrist yaw). According to the visualization of the striking movements this covariance pattern occurs when the ball is hit close to the plane passing through the center of the table and there is a short lateral displacement, this is why the shoulder yaw (output number 1) has a small marginal variance in this state.

Regarding the hidden state number 1 (See figure 12b) it exhibits a particular behavior because the covariance structure is diagonal, which means that the outputs are being represented as if they were independent of each other. It turns out that this hidden state explains a single segment of the Viterbi sequence where an anomalous event occurred: the robot arm stopped moving and stayed in resting position for a while—See the cyan segment in figure 13. All the outputs therefore remained constant and the variances in the diagonal of the covariance matrix in 12b account for the changes due to independent output noise processes. This highlights the possibility of using the proposed model as an anomalous behavior detector.

Finally, the hidden state number 0 is scarce in the Viterbi sequence with only three occurrences in the whole observation. It is worth of attention the inverse relationship between outputs 1 and 2 derived from the corresponding upper-left submatrix in figure 12a. Even though this is a property which clearly differentiates this hidden state from the others, it proved to be difficult to notice this property in the visualization of its reduced number of occurrences in the strikings movement. According to the observed occurrences, this hidden state seems to correspond with the behavior which brings the arm back to its resting position after hitting the ball, but this only happened for a subset of the strikes.

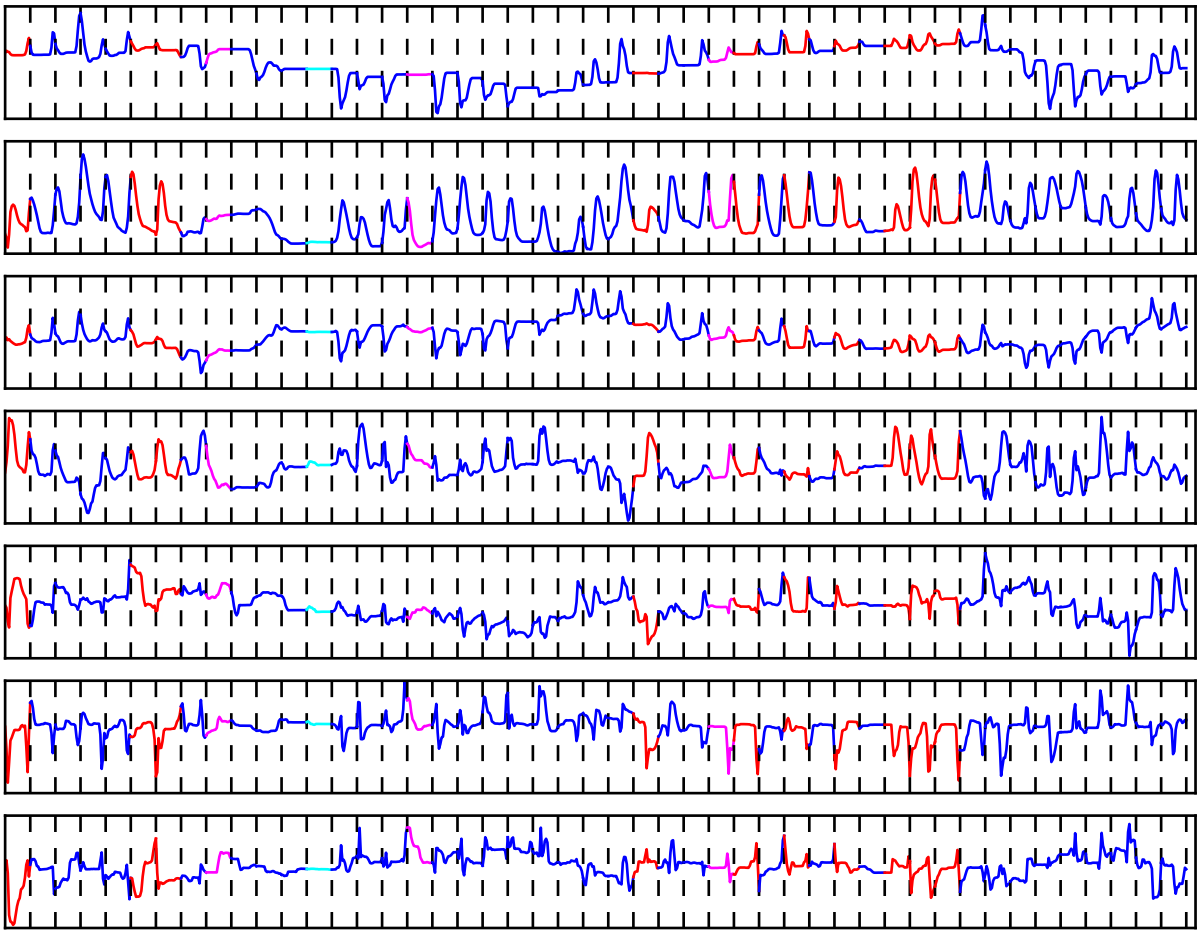


Figure 13: Robot arm outputs over time while performing striking movements. From top to bottom the outputs correspond to: shoulder yaw, shoulder pitch, shoulder roll, elbow joint, wrist yaw, wrist pitch and wrist roll. The outputs are colored according the Viterbi sequence in (50) (magenta: hidden state 0, cyan: hidden state 1, red: hidden state 2, blue: hidden state 4). The vertical dashed lines denote the chosen segmentation.

8 Conclusions

In this work a novel probabilistic parameterization of motor primitives and their sequential composition is proposed relying on LFMs and HMMs. The resulting joint model is an HMM with an LFM as observation model (i.e. emission process), which embodies the following properties:

- **Discrete changes modeling over dynamics and forces**: Movement realizations have some discrete and non-smooth changes on the forces and dynamical systems which govern them. Under the proposed approach, these changes can be captured at certain time locations depending on the particular choice of the sample locations set (equation (23)).
- **Redundancy learning over dynamics**: Having a set of primitives (hidden states) means that the diversity of the dynamic regimes present in a movement realization can be captured. Moreover, given that this set is finite, the existing redundancy in the dynamics can also be inferred by the model.
- **Scalability**: The size of the covariance matrices belonging to different hidden states (LFMs) is kept fixed regardless of the observed trajectory length. Therefore, movement realizations can be processed in linear time with respect to the realization length. In contrast with the switched dynamical LFM (SDLFM) [19] where the covariance matrix grows quadratically with the observation length.
- **Correlations between outputs**: The assumption of having the same set of latent forces governing the modeled outputs (See section 4.2.2) in the emission process allows us to do transfer learning between outputs. This means that, for a given output, the information at other outputs can help to improve prediction.

We also showed how to estimate the model's parameters from data using the EM algorithm (section 6.3). Remarkably, the HMM inference algorithms remain largely unchanged by the LFM observation model and, likewise, the training algorithm except for estimating the emission parameters.

Through synthetic and real data experiments, the model's capability to identify the occurrence of motor primitives after a training stage was successfully validated. In particular, we demonstrated how the model can successfully derive a set of motor primitives for the human walking behavior and detect the same pattern of primitives over unseen realizations (See section 7.2.1). In section 7.2.2 we also showed how the set of inferred motor primitives for a particular behavior—table tennis striking movements—can be analyzed for extracting knowledge and insights about the movement.

8.1 Research Outcomes

In terms of scientific production we achieved:

- D. Agudelo-España, M. A. Álvarez, and Á. A. Orozco, *Definition and Composition of Motor Primitives Using Latent Force Models and Hidden Markov Models*. Cham: Springer International Publishing, 2017, pp. 249–256. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-52277-7_31.

  This paper was presented orally at the 21st Iberoamerican Congress on Pattern Recognition (CIARP 2016) in Lima, Peru.

- The software package used to implement the proposed model and to execute all the experiments reported in this document is written in Python and can be freely downloaded from https://github.com/DiegoAE/HMMLFM.

8.2 Future Work

We consider the following possible ways of extending this work:

- Alternative formulations which support variable-length segments are suggested to increase the model's flexibility. In particular, Hidden Semi-Markov Models (HSMMs) represent a particular way to achieve this.

- Automatic strategies to set the model structure (e.g. number of latent forces, number of hidden states) based on data should be devised.

- A further step in the validation might involve using the model as a probabilistic generative model in a classification task as in the case of identifying pathological walking or distinguishing between different walking styles. A preliminary successful experiment in this direction was carried out in section 7.2.1.

- In order to perform human motion synthesis we require learning further from the training data, in addition to the movement realizations dynamics. A possible extension is also learn a parametric form for the LFMs mean function.

Bibliography

[1] E. Bizzi, A. d'Avella, P. Saltiel, and M. Tresch, "Book review: Modular organization of spinal motor systems," *The Neuroscientist*, vol. 8, no. 5, pp. 437–442, 2002.

[2] A. d'Avella, P. Saltiel, and E. Bizzi, "Combinations of muscle synergies in the construction of a natural motor behavior," *Nature neuroscience*, vol. 6, no. 3, pp. 300–308, 2003.

[3] A. d'Avella and E. Bizzi, "Shared and specific muscle synergies in natural motor behaviors," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 8, pp. 3076–3081, 2005.

[4] T. Flash and B. Hochner, "Motor primitives in vertebrates and invertebrates," *Current opinion in neurobiology*, vol. 15, no. 6, pp. 660–666, 2005.

[5] F. Meier, E. Theodorou, and S. Schaal, "Movement segmentation and recognition for imitation learning," in *International Conference on Artificial Intelligence and Statistics*, 2012, pp. 761–769.

[6] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction," *Science*, vol. 233, no. 4771, pp. 1416–1419, 1986.

[7] I. V. Grinyagin, E. V. Biryukova, and M. A. Maier, "Kinematic and dynamic synergies of human precision-grip movements," *Journal of neurophysiology*, vol. 94, no. 4, pp. 2284–2294, 2005.

[8] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," Tech. Rep., 2002.

[9] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *Robotics Research. The Eleventh International Symposium*. Springer, 2005, pp. 561–572.

[10] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[11] E. A. Rückert, G. Neumann, M. Toussaint, and W. Maass, "Learned graphical models for probabilistic planning provide a new class of movement primitives," 2013.

[12] O. C. Jenkins and M. J. Matarić, "Deriving action and behavior primitives from human motion data," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3. IEEE, 2002, pp. 2551–2556.

[13] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 2, pp. 283–298, 2008.

[14] K. Pullen and C. Bregler, "Motion capture assisted animation: Texturing and synthesis," in *ACM Transactions on Graphics (TOG)*, vol. 21, no. 3. ACM, 2002, pp. 501–508.

[15] O. C. Jenkins and M. J. Mataric, "Automated derivation of behavior vocabularies for autonomous humanoid motion," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, 2003, pp. 225–232.

[16] R. Urtasun, P. Glardon, R. Boulic, D. Thalmann, and P. Fua, "Style-based motion synthesis," in *Computer Graphics Forum*, vol. 23, no. 4. Wiley Online Library, 2004, pp. 799–812.

[17] M. A. Álvarez, D. Luengo, and N. D. Lawrence, "Linear latent force models using gaussian processes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 11, pp. 2693–2705, 2013.

[18] M. A. Alvarez, D. Luengo, and N. D. Lawrence, "Latent force models," in *International Conference on Artificial Intelligence and Statistics*, 2009, pp. 9–16.

[19] M. Alvarez, J. R. Peters, N. D. Lawrence, and B. Schölkopf, "Switched latent force models for movement segmentation," in *Advances in neural information processing systems*, 2010, pp. 55–63.

[20] M. Brand and A. Hertzmann, "Style machines," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 183–192.

[21] S. Chiappa and J. R. Peters, "Movement extraction by detecting dynamics switches and repetitions," in *Advances in neural information processing systems*, 2010, pp. 388–396.

[22] B. Williams, M. Toussaint, and A. J. Storkey, "Modelling motion primitives and their timing in biologically executed movements," in *Advances in neural information processing systems*, 2008, pp. 1609–1616.

[23] Z. Ghahramani and M. I. Jordan, "Factorial hidden markov models," *Machine learning*, vol. 29, no. 2-3, pp. 245–273, 1997.

[24] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[25] M. A. Alvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *arXiv preprint arXiv:1106.6251*, 2011.

[26] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in neural information processing systems*, 2013, pp. 2616–2624.

[27] N. D. Lawrence, G. Sanguinetti, and M. Rattray, "Modelling transcriptional regulation using gaussian processes," in *Advances in Neural Information Processing Systems*, 2006, pp. 785–792.

[28] G. Sanguinetti, A. Ruttor, M. Opper, and C. Archambeau, "Switching regulatory models of cellular stress response," *Bioinformatics*, vol. 25, no. 10, pp. 1280–1286, 2009.

[29] S. Schaal, "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics," in *Adaptive Motion of Animals and Machines*. Springer, 2006, pp. 261–280.

[30] S. Schaal, C. G. Atkeson, and S. Vijayakumar, "Scalable techniques from nonparametric statistics for real time robot learning," *Applied Intelligence*, vol. 17, no. 1, pp. 49–60, 2002.

[31] C. E. Rasmussen, "Gaussian processes for machine learning," 2006.

[32] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[33] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[34] K. P. Murphy, "Hidden semi-markov models (hsmms)," 2002.

[35] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[36] J. Perry, J. R. Davids *et al.*, "Gait analysis: Normal and pathological function." *Journal of Pediatric Orthopaedics*, vol. 12, no. 6, p. 815, 1992.

[37] Z. C. Li, R. Chellali, and Y. Yang, *Latent Force Models for Human Action Recognition*. Cham: Springer International Publishing, 2016, pp. 237–246.