Dimensionality Reduction for Probabilistic Movement Primitives

Adrià Colomé[1], Gerhard Neumann[2], Jan Peters[2,3] and Carme Torras[1]

Abstract— **Humans as well as humanoid robots can use a large number of degrees of freedom to solve very complex motor tasks. The high-dimensionality of these motor tasks adds difficulties to the control problem and machine learning algorithms. However, it is well known that the intrinsic dimensionality of many human movements is small in comparison to the number of employed DoFs, and hence, the movements can be represented by a small number of synergies encoding the couplings between DoFs. In this paper, we want to apply Dimensionality Reduction (DR) to a recent movement representation used in robotics, called Probabilistic Movement Primitives (ProMP). While ProMP have been shown to have many benefits, they suffer with the high-dimensionality of a robotic system as the number of parameters of a ProMP scales quadratically with the dimensionality. We use probablistic dimensionality reduction techniques based on expectation maximization to extract the unknown synergies from a given set of demonstrations. The ProMP representation is now estimated in the low-dimensional space of the synergies. We show that our dimensionality reduction is more efficient both for encoding a trajectory from data and for applying Reinforcement Learning with Relative Entropy Policy Search (REPS).**

I. Introduction

Movement Primitives (MPs) are a common approach to characterize motion in robotics, usually as a compact representation that easily permits learning tasks based on changes in their parameters. Over the last years, Dynamic Movement Primitives (DMPs) have been widely used for motion representation and learning [1], [2]. However, DMPs are a deterministic approach to motion representation, thus they are not capable of representing motion variability. In contrast, the recently proposed ProMP [3] approach is capable of capturing the variance of a set of demonstrations of the same task to a robot and reproducing the trajectory with the same variance over time, thanks to a stochastic model-based linear feedback controller.

However, when learning a certain task with a robot, the number of Degrees of Freedom (DoF) of the robot is typically much higher than the dimensionality of the task space. In classic body mechanics, the *Bernstein Problem* [4] is known as the problem of coordinating the abundant number of actuators (muscles) of a human body, which our nervous system cannot initially handle, such that a certain motion is performed [5]. In addition, some tasks have environment descriptors, such as goal position, via-point, etc., called context variables, that may vary and need to be taken into account for a better motion characterization.

In this paper, we address the problem of fitting a low-dimensional, probabilistic representation to a set of demonstrations of a task. In order to achieve such representation, we will modify the ProMP representation such that the movement is represented in a linear subspace of the full configuration space. Allowing for context variables, we will derive an Expectation-Maximization (EM) closed-form solution to find the mean, covariance and context dependencies of the ProMP in a latent low-dimensional subspace, its linear embedding in the joint space (with a coupling matrix) and the state covariance.

Dimensionality reduction over the DoF of robots is a common approach for grasping and hand motion [6], [7]. However, it has been less used for arm robot skills. As curse of dimensionality affects the performance of most RL algorithms when applied to robots with a relatively high number of DoF, such as the NAO robot in Fig. 1, new approaches to Reinforcement Learning (RL) with Dimensionality Reduction (DR) are being developed [8], [9]. Both approaches are based on Principal Component Analysis (PCA) or its probabilistic version (PPCA). In this paper, we directly extract the latent space with EM, without requiring PCA. PCA is just used as initialization of the EM approach.

Fig. 1. NAO robot interactuating with people. Picture provided by the Ave Maria Foundation in Sitges.

Using EM instead of PCA in combination with the ProMP approach comes with an important benefit, since the ProMP provides a time-dependent variance profile. While the variance profile is important in many applications, it

[acolome@iri.upc.edu, neumann@ias.tu-darmstadt.de, mail@jan-peters.net, torras@iri.upc.edu]. This research was carried out during a stay of A. Colomé at the Intelligent Autonomous Systems Lab in Darmstadt. This work is partially funded by CSIC Project MANIPlus (201350E102) and the European Union under grant agreement #270327 (CompLACS). Adrià Colomé is also supported by the Spanish Ministry of Education, Culture and Sport via a FPU doctoral grant (AP2010-1989).

[1] Inst. de Robòtica i Inf. Ind., CSIC-UPC, Barcelona, Spain.

[2] Intelligent Autonomous System Lab, TUD, Darmstadt, Germany.

[3] Max Planck Institute for Intelligent Systems, Tübingen, Germany.

also contains relevant information for the dimensionality reduction technique. Time points with a low variance are important for the movement and the PCA approach is not allowed to distort these time points. However, for time points with a large variance, a larger reproduction error of the DR technique is acceptable.

II. Probabilistic Movement Primitives (ProMP)

ProMP are a general approach to learn and encode a set of similar motion trajectories that present time-dependent variances over time as seen in Fig. 2. Given a number of basis functions per DoF N_f, ProMP use a $N_f \times 2$ time-dependent matrix $\mathbf{\Phi}_t = [\phi_t, \dot{\phi}_t]$ to encode position and velocity, ϕ_t being the vector of normalized kernel basis functions (e.g., uniformly distributed Gaussian basis function over time), thus the position and velocity state vector $\mathbf{y}_t$ can be represented as

$$\mathbf{y}_t = \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \mathbf{\Phi}_t^T \boldsymbol{\omega} + \boldsymbol{\epsilon}_y, \tag{1}$$

where $\boldsymbol{\epsilon}_y \sim \mathcal{N}(0, \mathbf{\Sigma}_y)$ is a zero-mean Gaussian noise and the weights $\boldsymbol{\omega}$ are also treated as random variables with a distribution

$$p(\boldsymbol{\omega}) = \mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega). \tag{2}$$

This distribution can be fitted, given a set of demonstration trajectories $\boldsymbol{\tau}_k = \{\mathbf{y}_t^k\}_{t=1..N_t}$, $k = 1..N_k$, by getting the weights $\boldsymbol{\omega}_k$ of each demonstration with least squares. Subsequently, the parameters of the distribution $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega, \mathbf{\Sigma}_y\}$, $\mathbf{\Sigma}_y$ being the state covariance, are fitted by a maximum likelihood estimate, i.e., we compute the sample mean and the sample covariance [10] of $\boldsymbol{\omega}$. Then the probability of observing a trajectory $\boldsymbol{\tau}$ can be expressed as the product of all timestep probabilities $p\left(\mathbf{y}_t; \boldsymbol{\theta}\right)$

$$p(\boldsymbol{\tau}; \boldsymbol{\theta}) = \prod_t \int \mathcal{N}(\mathbf{y}_t|\mathbf{\Phi}_t^T\boldsymbol{\omega}, \mathbf{\Sigma}_y)\mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega)d\boldsymbol{\omega} \tag{3}$$

Due to the probabilistic representation, the ProMP approach can represent motion variability while keeping other MP properties such as rescalation and linear representation w.r.t. parameters. It also allows for other operations such as modulation via probabilistic conditioning and combination by product [3].

In addition, ProMP also come with a model-based stochastic controller that reproduces the encoded trajectory distribution. In Fig. 2, we show the average and standard deviation of a ProMP and different sample trajectories from its distribution.

III. Dimensionality Reduction for ProMP (DR-ProMP)

Given a robot with d DoF, we will reduce the dimensionality of its motion representation to a latent space of dimension r, which is manually given. We can express the robot's state vector $\mathbf{y}_t$ with latent space variables $\mathbf{x}_t$ as

$$\mathbf{y}_t \simeq \begin{bmatrix} q_t \\ \dot{q}_t \end{bmatrix} = \mathbf{\Omega}_2 \mathbf{x}_t, \tag{4}$$

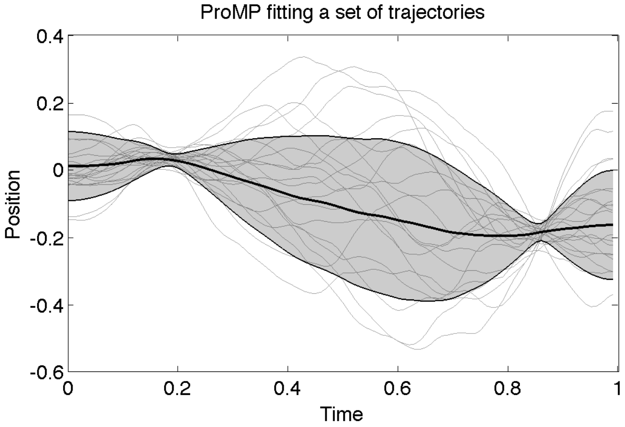

Fig. 2. ProMP fitting a set of trajectories, the mean and standard deviation for each timestep are shown.

where $\mathbf{\Omega}_s = I_s \otimes \mathbf{\Omega}$ $(sd \times sr)$, will be used throughout this paper as the Kronecker product of the s-dimensional identity matrix and what we define as coordination matrix $\mathbf{\Omega}$ $(d \times r)$, a linear mapping from an r-dimensional latent virtual joint space into the d-dimensional robot joint space. With this notation, we will also simplify $\mathbf{\Omega} = \mathbf{\Omega}_1$.

In order to represent the trajectory as a linear combination of some parameters $\boldsymbol{\omega}$ as in ProMP, we can write (4) as

$$\mathbf{y}_t = \mathbf{\Omega}_2 \mathbf{x}_t + \boldsymbol{\epsilon}_{fit} = \mathbf{\Omega}_2 \left(\mathbf{\Phi}_t^T \boldsymbol{\omega} + \boldsymbol{\epsilon}_x\right) + \boldsymbol{\epsilon}_{fit}, \tag{5}$$

with $\mathbf{\Phi}_t = [\phi_t, \dot{\phi}_t]$ being the $n \times 2$ matrix with the kernels used for the trajectory, and $\boldsymbol{\epsilon}_{fit}, \boldsymbol{\epsilon}_x$ the DR fitting error and the Gaussian noise for $\mathbf{x}$, respectively.

Thus, the probability of being in the latent state $\mathbf{x}_t$ given the weights $\boldsymbol{\omega} = [\boldsymbol{\omega}_1^T, ..., \boldsymbol{\omega}_r^T]^T$, $(rn \times 1)$ is

$$\begin{aligned} p(\mathbf{x}_t|\boldsymbol{\omega}) = & \left(\begin{bmatrix} \mathbf{x}_{1,t} \\ ... \\ \mathbf{x}_{d,t} \end{bmatrix} \middle| \begin{bmatrix} \mathbf{\Phi}_t^T & ... & \mathbf{0} \\ ... & ... & ... \\ \mathbf{0} & ... & \mathbf{\Phi}_t^T \end{bmatrix} \boldsymbol{\omega}, \mathbf{\Sigma}_x \right) \\ = & \; \mathcal{N}(\mathbf{x}_t|\mathbf{\Psi}_t^T\boldsymbol{\omega}, \mathbf{\Sigma}_x), \end{aligned} \tag{6}$$

with $\mathbf{\Psi}_t^T = I_r \otimes \mathbf{\Phi}_t^T$ $(2r \times rn)$. The probability of observing $\mathbf{y}_t$ given $\mathbf{x}_t$ is given by Eq.(5), i.e.,

$$p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t|\mathbf{\Omega}_2\mathbf{x}_t, \mathbf{\Sigma}_{fit}), \tag{7}$$

$\mathbf{\Sigma}_{fit}$ being the covariance of the reprojection error. Thus, the trajectory distribution in the full configuration space $\mathbf{y}_t$ is now

$$p(\mathbf{y}_t; \boldsymbol{\theta}) = \int \mathcal{N}(\mathbf{y}_t|\mathbf{\Omega}_2\mathbf{\Psi}_t^T\boldsymbol{\omega}, \mathbf{\Sigma}_y)\mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega)d\boldsymbol{\omega}, \tag{8}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega, \mathbf{\Omega}, \mathbf{\Sigma}_y\}$ is the set of parameters of the DR-ProMP representation, and $\mathbf{\Sigma}_y = \mathbf{\Sigma}_{fit} + \mathbf{\Omega}_2\mathbf{\Sigma}_x\mathbf{\Omega}_2^T$ the total system noise. We next define how Eq.(8) generalizes to the case of context variables.

Given a context variable $\mathrm{s} \in \mathbb{R}^M$, where M is the number of context variables, we can extend the ProMP framework to contextual ProMP, $p(\boldsymbol{\omega}|\mathbf{s}) = \mathcal{N}\left(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega + \mathbf{K}_\omega \mathbf{s}, \boldsymbol{\Sigma}_\omega\right)$, thus (8) becomes:

$$p(\mathbf{y}_t|\mathbf{s};\boldsymbol{\theta}) = \int \mathcal{N}(\mathbf{y}_t|\boldsymbol{\Omega}_2\boldsymbol{\Psi}_t^T\boldsymbol{\omega}, \boldsymbol{\Sigma}_y)\mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_\omega{+}\mathbf{K}_\omega\mathbf{s}, \boldsymbol{\Sigma}_\omega)d\boldsymbol{\omega},$$

where $\mathbf{K}_\omega$ is a linear mapping between the context variables and the ProMP mean.

A. DR-ProMP for robot control

In order to fully exploit the DR-ProMP, we must also characterize a stochastic controller that reproduces the motion variance. Assuming a known discrete-time linearized dynamics of the system with a time step of dt, the robot's dynamics equation is

$$\mathbf{y}_{t+dt} = (\mathbf{I} + dt\mathbf{A}_t)\mathbf{y}_t + \mathbf{B}_t dt\mathbf{u} + \mathbf{c}_t dt, \tag{9}$$

where $\mathbf{A_t}$, $\mathbf{B_t}$, and $\mathbf{c_t}$ are the system, input and drift terms of the first-order Taylor expansion of the dynamical system of the robot. This translates to a reduced dimensionality dynamic equation as

$$\boldsymbol{\Omega}_2\mathbf{x}_{t+dt} = (\mathbf{I} + dt\mathbf{A}_t)\boldsymbol{\Omega}_2\mathbf{x}_t + \mathbf{B}_t dt\mathbf{u} + \mathbf{c}_t dt. \tag{10}$$

Hence,

$$\mathbf{x}_{t+dt} = \boldsymbol{\Omega}_2^\dagger(\mathbf{I} + dt\mathbf{A}_t)\boldsymbol{\Omega}_2\mathbf{x}_t + \boldsymbol{\Omega}_2^\dagger\mathbf{B}_t dt\mathbf{u} + \boldsymbol{\Omega}_2^\dagger\mathbf{c}_t dt, \tag{11}$$

where † is used for the Moore-Penrose pseudoinverse. However, in Eq.(11), we still have a control action defined in the full $d-$dimensional space. Then, we can act on the reduced dimension $r-$space by using the control signal $\mathbf{u} \in \mathbb{R}^d$

$$\mathbf{u} = \boldsymbol{\Omega}\boldsymbol{\nu} + \boldsymbol{\epsilon}_u,$$

where $\boldsymbol{\epsilon}_u \sim \mathcal{N}(0, \boldsymbol{\Sigma}_u/dt)$ is defined as in [3]. The latent space controller $\boldsymbol{\nu} \in \mathbb{R}^r$ is defined with a linear gain $\mathbf{K}_t$ and drift $\boldsymbol{\kappa}_t$

$$\boldsymbol{\nu} = \mathbf{K}_t\mathbf{x}_t + \boldsymbol{\kappa}_t. \tag{12}$$

We will keep the controller noise in the high-dimensional space, allowing for exploration also outside the latent space.

Thus, inserting (12) into (11) results in

$$\begin{aligned} \mathbf{x}_{t+dt} = \; & \boldsymbol{\Omega}_2^\dagger\left[(\mathbf{I} + dt\mathbf{A}_t)\boldsymbol{\Omega}_2 + dt\mathbf{B}_t\boldsymbol{\Omega}\mathbf{K}_t\right]\mathbf{x}_t + \\ & \boldsymbol{\Omega}_2^\dagger\mathbf{B}_t(\boldsymbol{\Omega}\boldsymbol{\nu} + \boldsymbol{\epsilon}_u)dt + \boldsymbol{\Omega}_2^\dagger\mathbf{c}_t dt \\ = \; & \mathbf{F}\mathbf{x}_t + \mathbf{f} + \boldsymbol{\Omega}_2^\dagger\mathbf{B}_t dt\boldsymbol{\epsilon}_u, \end{aligned} \tag{13}$$

where $\mathbf{F} = \boldsymbol{\Omega}_2^\dagger\left[(\mathbf{I} + dt\mathbf{A}_t)\boldsymbol{\Omega}_2 + dt\mathbf{B}_t\boldsymbol{\Omega}\mathbf{K}_t\right]$ and $\mathbf{f} = dt\boldsymbol{\Omega}_2^\dagger\left(\mathbf{B}_t\boldsymbol{\Omega}\boldsymbol{\nu}dt + \boldsymbol{\Omega}_2^\dagger\mathbf{c}_t\right)$. Given the dynamics Equation (13), we can extract the probability of being in state $\mathbf{x}_{t+dt}$ at the next time step, i.e.,

$$\begin{gathered} p(\mathbf{x}_{t+dt}) = \textstyle\int \mathcal{N}(\mathbf{x}_{t+dt}|\mathbf{F}\mathbf{x}_t + \mathbf{f}, \boldsymbol{\Sigma}_s dt)\mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)d\mathbf{x}_t = \\ \mathcal{N}(\mathbf{x}_{t+dt}|\mathbf{F}\boldsymbol{\mu}_t + \mathbf{f}, \mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^T + \boldsymbol{\Sigma}_s dt), \end{gathered} \tag{14}$$

with $\boldsymbol{\mu}_t = \boldsymbol{\Psi}_t^T\boldsymbol{\mu}_\omega$ and $\boldsymbol{\Sigma}_t = \boldsymbol{\Psi}_t^T\boldsymbol{\Sigma}_\omega\boldsymbol{\Psi}_t$. We can match the control noise matrix $\boldsymbol{\Sigma}_s$ for each timestep by using the cross-correlation between consecutive steps of the trajectory distribution [3] and obtain

$$dt\boldsymbol{\Sigma}_s = \boldsymbol{\Sigma}_{t+dt} - \mathbf{C}_t^T\boldsymbol{\Sigma}_t\mathbf{C}_t,$$

with $\mathbf{C}_t = \boldsymbol{\Psi}_t^T\boldsymbol{\Sigma}_\omega\boldsymbol{\Psi}_{t+dt}$. The controller terms $\mathbf{K}_t$, $\boldsymbol{\kappa}_t$ can be obtained by matching $\{\boldsymbol{\mu}_{t+dt}, \boldsymbol{\Sigma}_{t+dt}\}$ from the system dynamics in Eq.(14) and the ProMP model. See [3] for more details of a similar deduction,

$$\mathbf{K}_t = \boldsymbol{\Omega}^T\mathbf{B}^\dagger\left[\boldsymbol{\Omega}_2\left(\dot{\boldsymbol{\psi}}_t^T\boldsymbol{\Sigma}_\omega\dot{\boldsymbol{\psi}}_t - \boldsymbol{\Sigma}_s/2\right) - \mathbf{A}_t\boldsymbol{\Omega}_2\boldsymbol{\Sigma}_t\right]\boldsymbol{\Sigma}_t^{-1},$$

and

$$\boldsymbol{\kappa}_t = \boldsymbol{\Omega}^\dagger\mathbf{B}^\dagger\left[\boldsymbol{\Omega}_2\dot{\boldsymbol{\psi}}_t^T\boldsymbol{\mu}_\omega - (\mathbf{A}_t\boldsymbol{\Omega}_2 + \mathbf{B}_t\boldsymbol{\Omega}\mathbf{K}_t)\,\boldsymbol{\psi}_t^T\boldsymbol{\mu}_\omega - \mathbf{c}_t\right],$$

and the controller noise covariance estimation is given by

$$\boldsymbol{\Sigma}_u = \mathbf{B}^\dagger\boldsymbol{\Omega}_2\boldsymbol{\Sigma}_s\boldsymbol{\Omega}_2^T\mathbf{B}^{T\dagger} + \alpha^2\mathbf{I}. \tag{15}$$

Note that, defined as in (15), $\boldsymbol{\Sigma}_u$ would be a $d{\times}d$ symmetric, semipositive definite matrix with rank r at most, corresponding to that of the latent subspace defined by $\boldsymbol{\Omega}$. Adding the term $\alpha\mathbf{I}$, for a small value of α will ensure we can explore the neighbourhood of the latent space when executing the ProMP.

B. Fitting DR-ProMP parameters with EM

We can estimate the set of parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega, \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y, \mathbf{K}_\omega\}$ in closed form using Expectation-Maximization (EM). Without loss of generality, we will use only joint position data, i.e., $\boldsymbol{\Psi}_t^T$ will now be a $(r \times rn)$ matrix, and $\mathbf{y}_t^k$ will be a d-dimensional vector. We will use the vector $\mathbf{Y}^k$ to denote the concatenated position vectors of a single trajectory k,

$$\mathbf{Y}^k = \left[\mathbf{y}_1^{k\,T}, ..., \mathbf{y}_{N_t}^{k\;\;T}\right]^T.$$

For our EM-algorithm, we will consider the most general case of a contextual ProMP in the latent space $\mathbf{x}$. Hence, we need to estimate the following parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega, \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y, \mathbf{K}_\omega\}$. Additionally, we want to use our model estimation algorithm for policy search algorithms that are based on data reweighting. These algorithms introduce a weighting d_k for each trajectory. Hence, we also have to consider such a weighting in our EM algorithm. In this section, the most general case will be considered for obtaining the ProMP parameters. That is, a lower-dimensional estimation with context variables and weights for each demonstration.

For a context-free ProMP, $\mathbf{K}_\omega$ can be set to zero, while the weights d_k can be set to 1 when all demonstrations have the same importance.

We will maximize the weighted marginal log-likelihood $\sum_k d_k \log p(\mathbf{y}_t|s, \boldsymbol{\theta})$ of the data and the latent space representation, thus we have to derive the equations with the marginalized $\boldsymbol{\omega}$ with the difficulties it entails. The following subsections explain how to obtain the log-likelihood function and differentiate it.

1) Expectation step

In the expectation step, we must find the prior probabilities for each demonstration k with the old parameters $\boldsymbol{\theta}^{old}$:

$$p(\boldsymbol{\omega}|\mathbf{Y}^k) = \frac{p(\mathbf{Y}^k|\boldsymbol{\omega})p(\boldsymbol{\omega})}{p(\mathbf{Y}^k)} \propto p(\mathbf{Y}^k|\boldsymbol{\omega})p(\boldsymbol{\omega}), \tag{16}$$

where, using $\mathbf{\Omega}_{N_t} = \mathbf{I}_{N_t} \otimes \mathbf{\Omega}$,

$$p(\mathbf{Y}^k|\boldsymbol{\omega}) = \mathcal{N}(\mathbf{Y}^k|\mathbf{\Omega}_{N_t}\mathbf{\Psi}^T\boldsymbol{\omega}, \mathbf{I}_{N_t} \otimes \mathbf{\Sigma}_y). \tag{17}$$

Note that in the contextual case, we have omitted the conditioning on the context variables for simplicity of the equations. Using the Bayes rule for Gaussian distributions, see Equations (39) and (40) in [11], we obtain:

$$p(\boldsymbol{\omega}|\mathbf{Y}^k) = \mathcal{N}(\boldsymbol{\omega}|\boldsymbol{\mu}_k, \mathbf{\Sigma}_k),$$

where

$$\boldsymbol{\mu}_k = \boldsymbol{\mu}_\omega + \mathbf{K}_\omega\mathbf{s}_k + \mathbf{\Sigma}_\omega\mathbf{\Psi}\mathbf{\Omega}_{N_t}^T\mathbf{\Gamma}^{-1}(\mathbf{Y}^k - \mathbf{\Omega}_{N_t}\mathbf{\Psi}^T(\boldsymbol{\mu}_\omega + \mathbf{K}_\omega\mathbf{s}_k))$$

$$\mathbf{\Sigma}_k = \mathbf{\Sigma}_\omega - \mathbf{\Sigma}_\omega\mathbf{\Psi}\mathbf{\Omega}_{N_t}^T\mathbf{\Gamma}^{-1}\mathbf{\Omega}_{N_t}\mathbf{\Psi}^T\mathbf{\Sigma}_\omega,$$

and $\mathbf{\Gamma}$ is given by $\mathbf{\Gamma} = \mathbf{I}_{N_t} \otimes \mathbf{\Sigma}_y + \mathbf{\Omega}_{N_t}\mathbf{\Psi}^T\mathbf{\Sigma}_\omega\mathbf{\Psi}\mathbf{\Omega}_{N_t}^T$.

2) Maximization step

Given the posterior probabilities $p(\boldsymbol{\omega}|\mathbf{Y}^k)$ for each demonstration, we now maximize the weighted expectation of the log-likelihood function, where d_k is used as weight for each trajectory, i.e.,

$$L = \sum_{k=1}^{N_d} d_k \mathbb{E}_{\boldsymbol{\omega}|\mathbf{Y}^K;\boldsymbol{\theta}^{old}}\left[\log\left(p(\boldsymbol{\omega}, \mathbf{Y}^k; \boldsymbol{\theta})\right)\right]$$

$$= \sum_{k=1}^{N_d} d_k \int_{\boldsymbol{\omega}} p(\boldsymbol{\omega}|\mathbf{Y}^k; \boldsymbol{\theta}^{old}) \log\left(p(\mathbf{Y}^k|\boldsymbol{\omega})p(\boldsymbol{\omega})\right) d\boldsymbol{\omega},$$

where

$$\log p(\mathbf{Y}^k|\boldsymbol{\omega})p(\boldsymbol{\omega}) = \log p(\boldsymbol{\omega}) + \sum_{t=1}^{N_t} \log p(\mathbf{y}_t^k|\boldsymbol{\omega}),$$

with $p(\boldsymbol{\omega})$ defined as in (2) and $p(\mathbf{y}_t^k|\boldsymbol{\omega}) = \mathcal{N}(\mathbf{y}_t^k|\mathbf{\Omega}\mathbf{\Psi}^T\boldsymbol{\omega}, \mathbf{\Sigma}_y)$.

Then, using the expectation identities for linear and quadratic transformations, (31) and (33) in [11], we obtain the value of the likelihood function to maximize in the M-step:

$$\begin{aligned} L = &-\tfrac{1}{2}\left[\left(\sum_{k=1}^{N_d} d_k\right)\log|2\pi\mathbf{\Sigma}_\omega| + N_t \log|2\pi\mathbf{\Sigma}_y|\right] \\ &-\tfrac{1}{2}\sum_{k=1}^{N_d} d_k(\boldsymbol{\mu}_\omega + \mathbf{K}_\omega\mathbf{s} - \boldsymbol{\mu}_k)\mathbf{\Sigma}_\omega^{-1}(\boldsymbol{\mu}_\omega + \mathbf{K}_\omega\mathbf{s} - \boldsymbol{\mu}_k) \\ &-\tfrac{1}{2}\sum_{k=1}^{N_d} d_k \sum_{t=1}^{N_t}\left[(y_t^k - \mathbf{\Omega}\mathbf{\Psi}_t^T\boldsymbol{\mu}_k)^T\mathbf{\Sigma}_y^{-1}(y_t^k - \mathbf{\Omega}\mathbf{\Psi}_t^T\boldsymbol{\mu}_k)\right. \\ &\left.+\mathrm{tr}(\mathbf{\Psi}_t\mathbf{\Omega}^T\mathbf{\Sigma}_y^{-1}\mathbf{\Omega}\mathbf{\Psi}_t^T\mathbf{\Sigma}_k)\right] - \tfrac{1}{2}\sum_{k=1}^{N_d} d_k \mathrm{tr}(\mathbf{\Sigma}_\omega^{-1}\mathbf{\Sigma}_k). \end{aligned}$$

Now, we can derivate L w.r.t. each of the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega^{-1}, \mathbf{\Omega}, \mathbf{\Sigma}_y^{-1}, \mathbf{K}_\omega\}$. Using the *funky trace derivative* [12], that $\lambda = tr(\lambda)$ for scalar values, the invariance of the trace w.r.t. cyclic permutations, the derivative of the log of a determinant, the derivative of a product in a trace [12] and the derivative w.r.t. the transpose of a matrix, we can obtain a closed-form solution of the parameters by setting each derivative to zero

$$\boldsymbol{\mu}_\omega = \left(\sum_{k=1}^{N_d} d_k\right)^{-1} \sum_{k=1}^{N_d} d_k(\boldsymbol{\mu}_k - \mathbf{K}_\omega\mathbf{s}), \tag{18}$$

$$\mathbf{\Sigma}_\omega = \left(\sum_{k=1}^{N_d} d_k\right)^{-1} \sum_{k=1}^{N_d} d_k\left[\mathbf{\Sigma}_k + (\boldsymbol{\mu}_\omega - \boldsymbol{\mu}_k)(\boldsymbol{\mu}_\omega - \boldsymbol{\mu}_k)^T\right], \tag{19}$$

$$\begin{aligned} \mathbf{\Omega} = &\left[\sum_{k=1}^{N_d} d_k \sum_{t=1}^{N_t} y_t^k(\boldsymbol{\mu}_k^T\mathbf{\Psi}_t)\right] \\ &\cdot\left[\sum_{t=1}^{N_t}\mathbf{\Psi}_t^T\sum_{k=1}^{N_d} d_k\left(\mathbf{\Sigma}_k + \boldsymbol{\mu}_k\boldsymbol{\mu}_k^T\right)\mathbf{\Psi}_t\right]^\dagger, \end{aligned} \tag{20}$$

$$\mathbf{K}_\omega = \left[\sum_{k=1}^{N_d} d_k(\boldsymbol{\mu}_k - \boldsymbol{\mu}_\omega)\mathbf{s}_k^T\right]\left[\sum_{k=1}^{N_d} d_k\mathbf{s}_k\mathbf{s}_k^T\right]^\dagger, \tag{21}$$

$$\begin{aligned} \mathbf{\Sigma}_y = &\left(\sum_{k=1}^{N_d} d_k\right)^{-1} \tfrac{1}{N_t}\sum_{k=1}^{N_d}\sum_{t=1}^{N_t} d_k[\mathbf{\Omega}\mathbf{\Psi}_t^T\mathbf{\Sigma}_k\mathbf{\Psi}_t\mathbf{\Omega}^T \\ &+(\mathbf{y}_t^k - \mathbf{\Omega}\mathbf{\Psi}_t^T\boldsymbol{\mu}_k)(\mathbf{y}_t^k - \mathbf{\Omega}\mathbf{\Psi}_t^T\boldsymbol{\mu}_k)^T]. \end{aligned} \tag{22}$$

In order to update all the parameters in the M-step, we will first obtain the new mean for the weights from Eq.(18), and subsequently use it to obtain its covariance with Eq.(19). As the next step, we compute the new coordination matrix $\mathbf{\Omega}^{new}$ with (20) and $\mathbf{K}_\omega$ in the case of contextual ProMP with (21). Finally, we use all the newly computed parameters to obtain the new noise covariance $\mathbf{\Sigma}_y$ with (22).

3) Initialization

Given a fixed value $r \leq d$ of the latent space dimension, a good initialization of the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega, \mathbf{\Omega}, \mathbf{\Sigma}_y, \mathbf{K}_\omega\}$ can increase the convergence speed of the EM algorithm. To that purpose, we will perform PCA on the trajectories $\mathbf{Y}^k$ of the robot for all demonstrations $k = 1..N_d$. After obtaining an initial guess for $\mathbf{\Omega}$ and use it in Eq.(5), we obtain the fitting weights $\boldsymbol{\omega}_k$ for each demonstration k, which we will use to initialize $\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega$

$$\boldsymbol{\mu}_\omega = \frac{1}{N_d}\sum_{k=1}^{N_d}\boldsymbol{\omega}_k,$$

$$\mathbf{\Sigma}_\omega = \frac{1}{N_d}\sum_{k=1}^{N_d}(\boldsymbol{\omega}_k - \boldsymbol{\mu}_\omega - \mathbf{K}_\omega\mathbf{s})(\boldsymbol{\omega}_k - \boldsymbol{\mu}_\omega - \mathbf{K}_\omega\mathbf{s})^T.$$

Finally, we can initialize $\mathbf{\Sigma}_y$ as

$$\mathbf{\Sigma}_y = \frac{1}{N_d N_t}\sum_{k=1}^{N_d}\sum_{t=1}^{N_t}(\mathbf{y}_t^k - \overline{\mathbf{y}})(\mathbf{y}_t^k - \overline{\mathbf{y}})^T,$$

where $\overline{y}$ is the average over all timesteps and demonstrations of the joint state vector.

In the case of a contextual ProMP, we will initialize $[\boldsymbol{\mu}_\omega, \mathbf{K}_\omega]$ together using weighted least squares with the weight vector $\mathbf{d} = [d_1, ..., d_{N_d}]$

$$[\boldsymbol{\mu}_\omega, \mathbf{K}_\omega]^T = \left(\mathbf{S} \cdot \text{diag}(\mathbf{d}) \cdot \mathbf{S}^T + \lambda \mathbf{I}\right)^{-1} \mathbf{S}^T \text{diag}(\mathbf{d}) \mathbf{W}_d^T, \tag{23}$$

where $\mathbf{W}_d = [\boldsymbol{\omega}_1, ..., \boldsymbol{\omega}_{N_d}]$ are the weights obtained from fitting the demonstrations, $\lambda \mathbf{I}$ a regularization term, $\mathbf{d}$ the weights vector for each demonstration and $\mathbf{S}$ is a matrix containing all the context vectors for the demonstrations

$$\mathbf{S} = \begin{bmatrix} 1 & ... & 1 \\ \mathbf{s}_1 & ... & \mathbf{s}_{N_d} \end{bmatrix},$$

where the 1s in the first row are added to be able to simultaneously compute $\boldsymbol{\mu}_\omega$ and $\mathbf{K}_\omega$.

4) Comparison of EM versus PCA

To illustrate the benefits of the EM-based algorithm presented in this section in comparison to PCA, we created d-dimensional probabilistic trajectories, tracked with a stochastic optimal controller as that in Fig. 2. We fitted these trajectories using both a PCA matrix projection and our EM-based approach and used the *Kullbach-Leibler divergence* [13]. We computed the KL divergence between the data distribution and the fitted models using EM and PCA. In Fig. 3, we can see the color plot of the ratio ρ, defining the relative KL-divergence gain w.r.t. the PCA approach for a set of simulated ProMP and their DR fitting. We can observe that, despite not being significantly better when the fitting dimension is equal or almost equal to the original dimension, the KL-divergence of the original trajectory distribution fitted with our approach is reduced by around 20% over the PCA approach with ρ defined as

$$\rho = \frac{\sum_{t=1}^{N_t} \text{KL}(p(y_t; \text{data}) | p(y_t; \boldsymbol{\theta}^{em})) - \text{KL}(p(y_t; \text{data}) | p(y_t; \boldsymbol{\theta}^{pca}))}{\sum_{t=1}^{N_t} \text{KL}(p(y_t; \text{data}) | p(y_t; \boldsymbol{\theta}^{pca}))}, \tag{24}$$

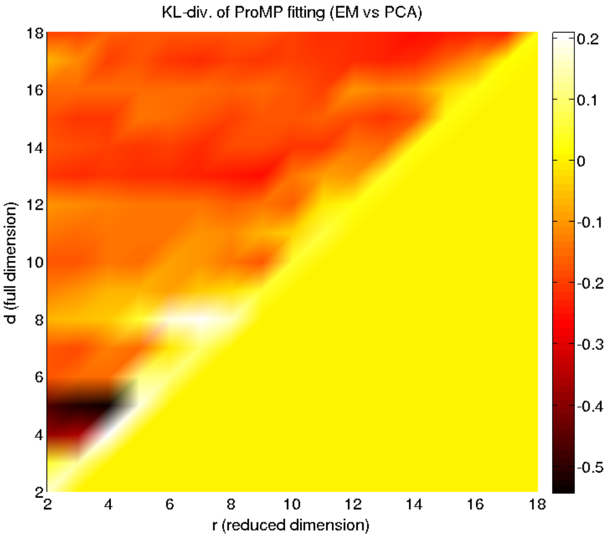

Fig. 3. Plot of the ratio ρ defined in (24), showing that the EM approach reduces the KL-divergence by around 20% w.r.t. PCA.

C. Reinforcement Learnig with DR-ProMP

The proposed EM algorithm can be straightforwardly used for RL algorithms that are based on data re-weighting [14]. These algorithms are used to choose the weighting d_k of each data trajectory. We will use the REPS algorithm [15], which can also be applied in the contextual case and has been shown to perform well in practice. REPS can be used for arbitrary policy representations and learns the parameters that maximize the expected reward. In order to obtain the policy update, it uses a bound on the KL-divergence between the old trajectory distribution $q_\theta(\boldsymbol{\tau})$ and the new trajectory distribution $p_\theta(\boldsymbol{\tau})$. In practice, REPS provides weights d_k for the set of N_k trajectories, which are then used to infer a new policy by using weighted maximum likelihood estimation. In our case, we will use our EM algorithm for learning synergetic ProMP to obtain a new policy. In the case of contextual variables, a contextual version of REPS can also be found in [14].

Algorithm 1 EM-REPS learning with DR-ProMP

```
Input:
Previous DR-ProMP parameters $\boldsymbol{\theta}^{old} = \{\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega, \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y, \mathbf{K}_\omega, \}$.
Kullback-Leibler divergence bound $\epsilon_{kl}$.
Other ProMP parameters $dt, N_t, N_f$.
for $k = 1...N_k$ do
    Obtain the context variable $\mathbf{s}_k$.
    Reproduce ProMP sampling from the trajectory distribution, store joint data $\mathbf{Y}^k$ and evaluate reward function $R_k$.
end for
Compute weights using contextual REPS: $d_k$ = reps.$(\mathbf{R}, \epsilon_{kl})$
while no convergence do
    Perform weighted EM in Sec.III-B to obtain new parameters $\boldsymbol{\theta}^{\text{new}} = \{\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y, \mathbf{K}_\omega\}$.
end while
```

After observing a set of a trajectories, and given their relative importance (weights) provided by the REPS algorithm, we can use the EM estimation in Section III-B to update the parameters of the ProMP using Equations (18)-(22). In Alg.1, we show the iterative procedure for learning with the EM-REPS approach. This algorithm updates all the ProMP parameters given the reward-based weights of the executed trajectories. If no context variables are considered, $\mathbf{K}_\omega$ and s can be ignored from Equations (18)-(22), and use the non-contextual REPS in [15].

IV. Experiments

We performed two experiments to evaluate the proposed approach, a first one fitting a lower-dimensionality walking policy for the NAO robot, and a comparison of methods for RL with DMP and REPS on a planar manipulator.

A. Walking Couplings of a NAO robot

We used our DR-ProMP approach to encapsulate similar walking behaviours of the robot *NAO* in Fig. 1. The trajectories for its leg joints, which are 6-DoF for each leg, were obtained by controlling the Zero Moment Point of the robot [16]. In Fig. 5, we show the observed distribution (in red) obtained from 13 different walking experiments and its fit with the proposed method (in blue) for a reduced dimensionality of $r = 4 \leq 12 = d$. We can clearly see that 4 synergies are enough to represent the whole walking behaviour. We can extract its couplings and relations from the matrix $\mathbf{\Omega}$, which relates the joints according to the correlation matrix plotted in Fig. 4. Note that the intuitive couplings between joints are those arising from a symmetry of the legs position and thus, joints $< 1, 7 >$, $< 3, 9 >$, $< 4, 10 >$, $< 5, 11 >$ will usually have opposite values, while $< 2, 8 >$, $< 6, 12 >$ would have the same sign. In fact, joints $< 1, 7 >$ are mechanically coupled, so the effective dimensionality is $d = 11$.

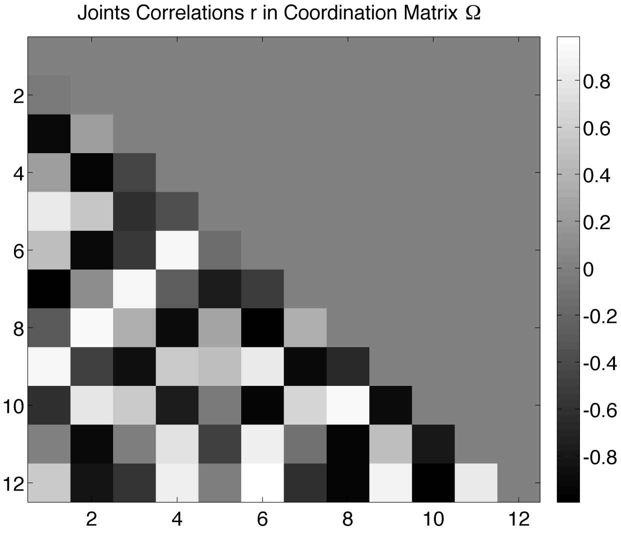

Fig. 4. Correlation between joints provided by the coordination matrix $\mathbf{\Omega}$. Black color indicates $r \sim -1$ while white indicates $r \sim 1$.

B. Planar Trajectory Tracking

We simulated a $d = 15$ DoF planar arm, with the all joints having a length of 1 m. The task was to follow a cartesian trajectory. Using the same initial and desired conditions for all experiments, we compared the following learning strategies:

- *DMP+REPS*. We obtained the weights for each demonstration with least squares, and fitted a distribution $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega)$ over them, which was used for sampling and exploring with REPS.
- *ProMP+REPS*.
- *EM-ProMP+REPS* as in Alg. 1 with the full dimension.
- *EM-ProMP+REPS* as in Alg. 1 with $r = 8$ and no perturbation ($\alpha = 0$) on $\mathbf{\Omega}$.
- *EM-ProMP+REPS* as in Alg. 1 with $r = 8$ and a perturbation of $\alpha = 0.01$ on $\mathbf{\Omega}$.

To keep the possibility of exploring outside the restricted joint subspace provided by the projection matrix $\mathbf{\Omega}$, we will add a perturbation $\epsilon_\alpha \sim \mathcal{N}(0, \alpha^2)$ to each element of the coordination matrix before every rollout, which will provide a similar effect to the α value given in Eq.(15). We used 100 trajectory samples, and updated the policy every 20 samples once we had the first 100. For the REPS algorithm, we took a Kullback-Leibler bound of $\epsilon_{KL} = 0.5$. We used 8 Gaussian kernel functions for each DoF, up to a total of $8d$ or $8r$ for the reduced dimension cases. We performed 100 policy updates and the reward function was given by

$$R = \sum_{t=1}^{N_t} -\left(\mathbf{e}_t^T \mathbf{D} \mathbf{e}_t + \ddot{\mathbf{q}}_t^T \mathbf{H} \ddot{\mathbf{q}}_t\right), \qquad (25)$$

where $\mathbf{D}$ and $\mathbf{H}$ are diagonal matrices. To generate samples with ProMP, the desired framework would be to use the stochastic controller defined in Sec.III-A. However, the controller is computationally more expensive and can be sensitive to numerical conditioning of the covariances involved for a small timestep. For these reasons, we sampled the trajectories by directly evaluating $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\mu}_\omega, \mathbf{\Sigma}_\omega)$.

In Fig. 6, we display the average and standard deviation over 10 experiments for each algorithm. The results show that, for $r = d$, the EM-based algorithm does not improve over the standard REPS update. This behaviour was expected as, for the full-rank case of the coordination matrix, the parameter update mostly becomes equivalent to the standard REPS case, with the addition of possible numerical error. However, the DR on the ProMP (black line in the plot) slightly improves performance over the previous algorithm. This improvement becoming more substantial when a perturbation α is added to the coordination matrix, yielding also a faster improvement on the earlier updates, due to the reduced parameter dimensionality. For the DMP case, we observe a more unstable behaviour in the earlier steps but better average (with more variance) on reward than with the standard ProMP approach.

V. Conclusions

In this paper, we provided a novel, EM-based approach to fit trajectory distributions with ProMP that computes a linear latent space for the DoF of the robot. Working in this latent space with a small perturbation on the coordination matrix or the controller provides a faster algorithm for RL with REPS, thanks to the drastical reduction of the parameters associated to those eliminated DoF. Our future aim is to automatically estimate the optimal space dimension r and improve the computational cost of evaluating the posterior of the expectation step in Sec.III-B.1.

Acknowledgement

We would also like to thank A. Abdolameki, from the Inst. of Electronics and Telematics Eng. of Aveiro (IEETA), for the help in obtaining experimental data for the NAO robot.

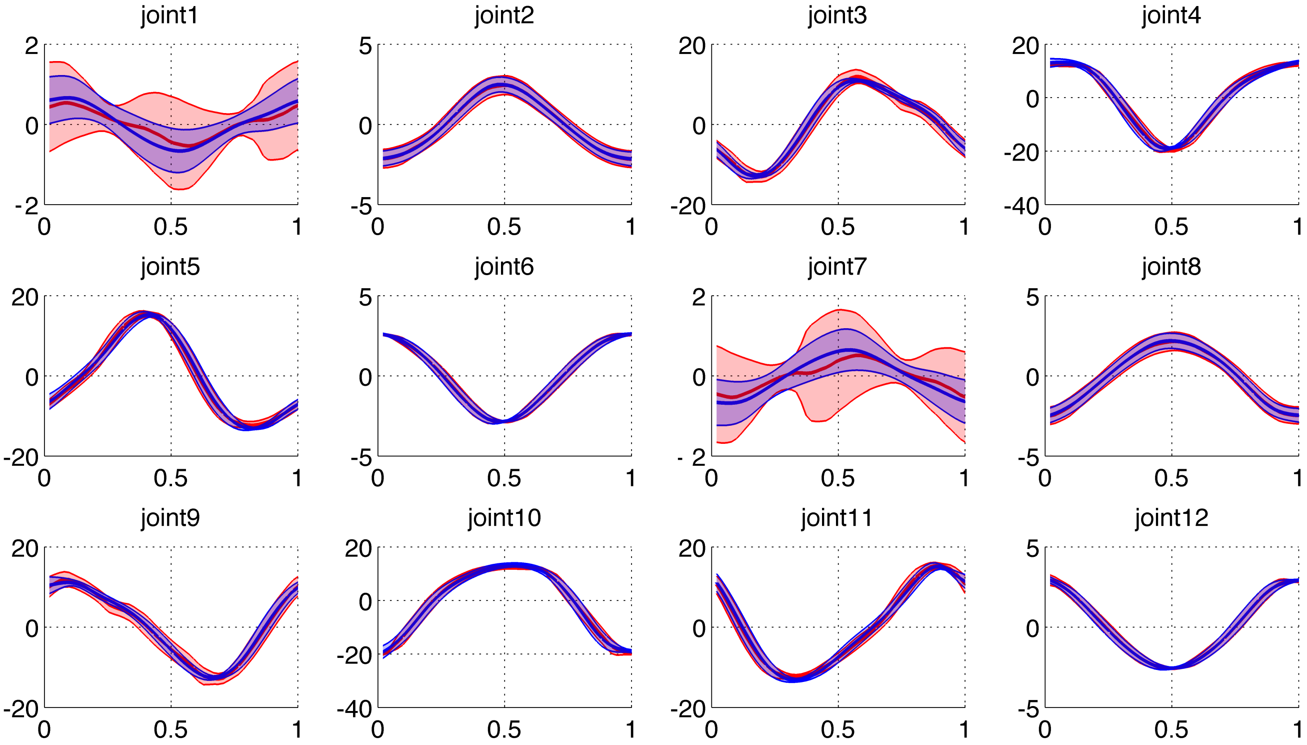

Fig. 5. Original walking trajectory distribution, in degrees, for each one of the 12 NAO leg joints (red) and the fitting obtained with a reduced dimension of 4 (blue) in a normalized time-scale. We can see that, despite the dimensionality reduction, the trajectory distribution could be reproduced accurately.

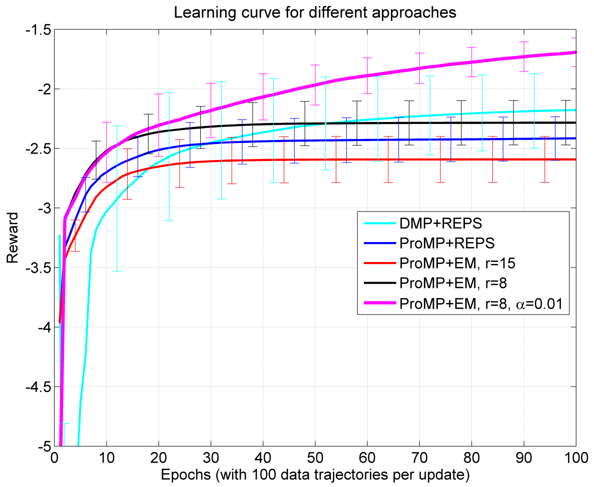

Fig. 6. Comparing different learning approaches on a 15-DoF planar manipulator simulated task.

References

[1] A. J. Ijspeert, J. Nakanishi and S. Schaal, “Movement Imitation with Nonlinear Dyamical Systems in Humanoid Robots”. *IEEE ICRA*, pp 1398-1403, 2002.

[2] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor and S. Schaal, “Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviours". *Neural Computation*, vol 25, no 2, pp 328-373 2013.

[3] A. Paraschos, G Neumann, C. Daniel, and J. Peters, “Probabilistic movement primitives". *In Advances in NIPS*, Cambridge, MA: MIT Press., 2013.

[4] N. Bernstein, “The Coordination and Regulation of Movements". Oxford: Pergamon Press. 1967.

[5] A. d’Avella and E. Bizzi,"Shared and specific muscle synergies in natural motor behaviours". *Proc Natl Acad Sci USA*. vol 22, no 102(8), pp 3076-3081, 2005.

[6] R. Vinjamuri, M. Sun, Ch.-Ch. Chang, H.-N. Lee, R. J. Sclabassi and Z.-H. Mao, “Dimensionality Reduction in Control and Coordination of the Human Hand". *IEEE Trans. on Biomedical Eng.*, vol 57, no. 2, pp 284-294, 2010.

[7] K.-S. Luck, G. Neumann, E. Berger, J. Peters and H. B. Amor, “Latent Space Policy Search for Robotics", *IEEE/RSJ IROS*, 2014.

[8] A. Colomé and C. Torras, “Dimensionality Reduction and Motion Coordination in Learning Trajectories with Dynamic Movement Primitives", *IEEE/RSJ IROS*, pp 1414-1420 2014.

[9] H. B. Amor, O. Kroemer, U. Hillenbrand, G. Neumann and J. Peters, “Generalization of Human Grasping for Multi-Fingered Robot Hands", *IEEE/RSJ IROS*, pp 2043 - 2050, 2012.

[10] A. Lazaric and M. Ghavamzadeh, “Bayesian Multi-Task- Reinforcement Learning". *27th International Conference on Machine Learning (ICML)*, pp 599-606, 2010.

[11] M. Toussaint, “Lecture Notes: Gaussian identities". [online] Available: `http://ipvs.informatik.uni-stuttgart.de/mlr/marc/notes/gaussians.pdf`

[12] J. Duchi, “Properties of the Trace and Matrix Derivatives". [online] Available: `http://www.eecs.berkeley.edu/~jduchi/projects/matrix_prop.pdf`

[13] S. Kullback and R.A. Leibler, “On Information and Sufficiency". *Annals of Mathematical Statistics 22*, vol 1, pp 79–86, 1951.

[14] M. P. Deisenroth, G. Neumann and J. Peters, “A survey on Policy Search for Robotics". *Foundations and Trends in Robotics*, vol 2, pp 1-142, 2013.

[15] J. Peters, K. Mülling and Y. Altün, “Relative Entropy Policy Search”.*Twenty-Fourth National Conf. on Artificial Intelligence*, pp 182-189, 2011.

[16] N. Shafii, A. Abdolmaleki, R. Ferreira, N. Lau and L.P. Reis, "Omnidirectional Walking and Active Balance for Soccer Humanoid Robot". *Progress in Artificial Intelligence*, vol 8154, pp 283-294, 2013.

x