Appendix A
Performing DR in the Parameter Space Instead of the Task Space

In this appendix, we consider the cases where the dimensionality reduction of the MPs is performed not on the task space, but rather in the parameter space by basically switching the order in which the kernel function matrix and the linear projection matrix defined in Chap. 7 are used for the representation of the state vector. A principal component approach is introduced for DMPs, while an EM approach is used for ProMPs.

A.1 DR in the Parameter Space for DMPs

Dimensionality reduction (DR) of DMPs, which in Sect. 7.2 was performed in the robot joint space, can also be performed in the parameter space with the expression:

$$\mathbf{f}(x_t) = \boldsymbol{\Phi}_t^T \boldsymbol{\Omega} \boldsymbol{\nu}, \tag{A.1}$$

with $\boldsymbol{\Phi}_t^T$ being a $(d \times dN_f)$ matrix of Gaussian basis functions and $\boldsymbol{\Omega}$ being a $(dN_f \times M_f)$ matrix with the mappings from a parameter space of dimension M_f to the whole DMP parameter space. The DMP weight vector $\boldsymbol{\nu}$ now has dimension M_f. In order to initialize the projection matrix $\boldsymbol{\Omega}$, we have to take the data matrix in Eq. (7.26), $\overline{\mathbf{F}}$, knowing that:

$$\overline{\mathbf{F}} = [\mathbf{f}_1, ..., \mathbf{f}_{N_t}] = [\boldsymbol{\Phi}_1^T \boldsymbol{\Omega} \boldsymbol{\nu}, ..., \boldsymbol{\Phi}_{N_t}^T \boldsymbol{\Omega} \boldsymbol{\nu}], \tag{A.2}$$

which can be expressed as a least-squares minimization problem as:

$$[\boldsymbol{\Phi}_1^{\dagger,T} \mathbf{f}_1, ..., \boldsymbol{\Phi}_{N_t}^{\dagger,T} \mathbf{f}_{N_t}] \simeq \boldsymbol{\Omega} \boldsymbol{\nu}, \tag{A.3}$$

A. Colomé and C. Torras, *Reinforcement Learning of Bimanual Robot Skills*, Springer Tracts in Advanced Robotics 134,
https://doi.org/10.1007/978-3-030-26326-3

where † represents the pseudoinverse matrix. We can then obtain the projection matrix $\boldsymbol{\Omega}$ by performing PCA in Eq. (A.3). Note that, in order to fit the matrix $\boldsymbol{\Omega}$ $(dN_f \times M_f)$, we need at least a number of timesteps $N_t > M_f$.

A.2 DR in the Parameter Space for ProMPs

In Chap. 7, we devised an Expectation-Maximization based method for finding a linear Dimensionality Reduction (DR) technique to represent ProMPs in a latent space of the DoF of the robot, using the expression $\mathbf{y}_t = \boldsymbol{\Omega}\boldsymbol{\Phi}_t^T\boldsymbol{\nu}$ for the state space, where $\boldsymbol{\Omega}$ was a $(d \times r)$ matrix, d being the DoF of the trajectory and r a latent space dimension. In this paper, we used the reverse order between the two terms $\boldsymbol{\Omega}$ and $\boldsymbol{\Phi}_t$, resulting in:

$$\mathbf{y}_t = \boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\nu}, \tag{A.4}$$

where $\boldsymbol{\Omega}$ is now a $(dN_f \times M_f)$ matrix and we use $\boldsymbol{\nu} \sim \mathcal{N}(\boldsymbol{\mu}_\nu, \boldsymbol{\Sigma}_\nu)$ to represent the latent space parameters. In this case, the linear DR is performed directly in the Gaussian weights space of the ProMP, instead of the DoF' space, resulting in the probability distribution for the state space:

$$\begin{aligned} p(\mathbf{y}_t) &= \int_{\boldsymbol{\nu}} \mathcal{N}\left(\mathbf{y}_t|\boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\nu}, \boldsymbol{\Sigma}_y\right)\mathcal{N}\left(\boldsymbol{\mu}_\nu, \boldsymbol{\Sigma}_\nu\right)d\boldsymbol{\nu} \\ &= \mathcal{N}\left(\mathbf{y}_t|\boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\mu}_\nu, \boldsymbol{\Sigma}_y + \boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\Sigma}_\nu\boldsymbol{\Omega}\boldsymbol{\Phi}_t^T\right) \end{aligned} \tag{A.5}$$

We then describe the Expectation-Maximization step for obtaining the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_\nu, \boldsymbol{\Sigma}_\nu, \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y\}$ by using the complete-data log-likelihood. $\boldsymbol{\Phi}_t^T$ will be a $(d \times M_f)$ matrix, and $\mathbf{y}_t^j$ will be a d-dimensional vector. We use the vector $\mathbf{Y}^j$ to denote the concatenated position vectors of a single trajectory j,

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

Now, we want to use our model estimation algorithm for policy search algorithms that are based on data re-weighting. These algorithms introduce a weighting d_k for each trajectory. Hence, we also have to consider such a weighting in our EM algorithm. We will maximize the weighted marginal log-likelihood $\sum_k d_k \log p(\mathbf{y}_t, \boldsymbol{\theta})$ of the data and the latent space representation, thus we have to derive the equations with the marginalized $\boldsymbol{\nu}$ with the difficulties it entails. The following sections explain how to obtain the log-likelihood function and differentiate it.

Expectation Step

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

$$p(\boldsymbol{\nu}|\mathbf{Y}^j) = \frac{p(\mathbf{Y}^j|\boldsymbol{\nu})p(\boldsymbol{\nu})}{p(\mathbf{Y}^j)} \propto p(\mathbf{Y}^j|\boldsymbol{\nu})p(\boldsymbol{\nu}), \tag{A.6}$$

where:

$$p(\mathbf{Y}^j|\boldsymbol{v}) = \mathcal{N}(\mathbf{Y}^j|\boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{v}, \mathbf{I}_{N_t} \otimes \boldsymbol{\Sigma}_y). \tag{A.7}$$

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, we obtain $p(\boldsymbol{v}|\mathbf{Y}^j) = \mathcal{N}(\boldsymbol{v}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where

$$\boldsymbol{\mu}_j = \boldsymbol{\mu}_v + \boldsymbol{\Sigma}_v\boldsymbol{\Omega}_{N_t}^T\boldsymbol{\Phi}\boldsymbol{\Gamma}^{-1}(\mathbf{Y}^j - \boldsymbol{\Phi}^T\boldsymbol{\Omega}_{N_t}\boldsymbol{\mu}_v)$$

$$\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_v - \boldsymbol{\Sigma}_v\boldsymbol{\Omega}_{N_t}^T\boldsymbol{\Phi}\boldsymbol{\Gamma}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Omega}_{N_t}\boldsymbol{\Sigma}_v,$$

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

Maximization Step

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

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

After some calculations, analogously to Sect. 7.1.1, we obtain:

$$\begin{aligned} L = &-\tfrac{1}{2}\left[\left(\sum_{j=1}^{N_d} d_j\right)\log|2\pi\,\boldsymbol{\Sigma}_v| + N_t\log|2\pi\,\boldsymbol{\Sigma}_y|\right] \\ &-\tfrac{1}{2}\sum_{j=1}^{N_d} d_j(\boldsymbol{\mu}_v - \boldsymbol{\mu}_j)\boldsymbol{\Sigma}_v^{-1}(\boldsymbol{\mu}_v - \boldsymbol{\mu}_j) \\ &-\tfrac{1}{2}\sum_{j=1}^{N_d} d_j \sum_{t=1}^{N_t}\left[(y_t^j - \boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\mu}_j)^T\boldsymbol{\Sigma}_y^{-1}(y_t^j - \boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\mu}_j)\right. \\ &\left.+\mathrm{tr}(\boldsymbol{\Omega}^T\boldsymbol{\Phi}_t\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Phi}_t^T\boldsymbol{\Omega}\boldsymbol{\Sigma}_j)\right] - \tfrac{1}{2}\sum_{j=1}^{N_d} d_j\mathrm{tr}(\boldsymbol{\Sigma}_v^{-1}\boldsymbol{\Sigma}_j). \end{aligned}$$

Then, we derivate L w.r.t. $\boldsymbol{\theta} = \{\boldsymbol{\mu}_v, \boldsymbol{\Sigma}_v, \boldsymbol{\Omega}, \boldsymbol{\Sigma}_y^{-1}\}$:

$$\boldsymbol{\mu}_v = \left(\sum_{j=1}^{N_d} d_j\right)^{-1}\sum_{j=1}^{N_d} d_j(\boldsymbol{\mu}_j), \tag{A.8}$$

$$\boldsymbol{\Sigma}_v = \left(\sum_{j=1}^{N_d} d_j\right)^{-1}\sum_{j=1}^{N_d} d_j\left[\boldsymbol{\Sigma}_j + (\boldsymbol{\mu}_v - \boldsymbol{\mu}_j)(\boldsymbol{\mu}_v - \boldsymbol{\mu}_j)^T\right], \tag{A.9}$$

$$\boldsymbol{\Omega} = \left[\sum_{t=1}^{N_t} d_j \boldsymbol{\Phi}_t^T \boldsymbol{\Sigma}_y^{-1} \boldsymbol{\Phi}_t\right]^{\dagger} \left[\sum_{t=1}^{N_t} \boldsymbol{\Phi}_t \boldsymbol{\Sigma}_y^{-1} \sum_{j=1}^{N_d} d_j y_t^j \boldsymbol{\mu}_j^T\right] \cdot \left[\sum_{j=1}^{N_d} d_j \left(\boldsymbol{\Sigma}_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^T\right)\right]^{\dagger} \tag{A.10}$$

$$\begin{aligned} \boldsymbol{\Sigma}_y = \left(\sum_{j=1}^{N_d} d_j\right)^{-1} \frac{1}{N_t} \sum_{j=1}^{N_d} \sum_{t=1}^{N_t} d_j \big[\mathbf{y}_t^j (\mathbf{y}_t^j - \boldsymbol{\Phi}_t^T \boldsymbol{\Omega} \boldsymbol{\mu}_j)^T + \boldsymbol{\Phi}_t^T \\ + \boldsymbol{\Omega} \boldsymbol{\mu}_j (\boldsymbol{\Phi}_t^T \boldsymbol{\Omega} \boldsymbol{\mu}_j - \mathbf{y}_t^j)^T + \boldsymbol{\Phi}_t^T \boldsymbol{\Omega} \boldsymbol{\Sigma}_j \boldsymbol{\Omega} \boldsymbol{\Phi}_t^T \big] \end{aligned} \tag{A.11}$$

Appendix B
External Resources

1- Closed-loop inverse kinematics for redundant robots
http://www.iri.upc.edu/groups/perception/#inverseKinematics

2- Friction model applications
http://www.iri.upc.edu/groups/perception/#ScarfTask

3- Realtime tracking and grasping of a moving object from range video
http://www.iri.upc.edu/groups/perception/#trackGrasp

4- Human-guided compliant control
http://www.iri.upc.edu/groups/perception/#adapt
http://www.iri.upc.edu/groups/perception/#adapt2

5- Dual Relative Entropy Policy Search
http://www.iri.upc.edu/groups/perception/#dualReps

6- Dimensionality Reduction for Dynamic Movement Primitives
http://www.iri.upc.edu/groups/perception/#drdmp

7- Contextualized ProMPs with obstacle avoidance
http://www.iri.upc.edu/groups/perception/#ProMPobstacles

8- Mixture models of adaptable ProMPs
http://www.iri.upc.edu/groups/perception/#DRGMM

A. Colomé and C. Torras, *Reinforcement Learning of Bimanual Robot Skills*, Springer Tracts in Advanced Robotics 134,
https://doi.org/10.1007/978-3-030-26326-3

Glossary

$\mathbf{J}$	Geometric Jacobian
$\mathbf{J}^{\dagger}$	Jacobian pseudoinverse
$\mathbf{q} = [q_1, \ldots, q_m]$,	Joint state
$\Delta\mathbf{q}$	joint state variation
$\mathbf{x}$	Task space variable
$\mathbf{e}$	Positioning error of the robot
$\kappa(\cdot)$	Condition number of a matrix
$FK(\cdot)$	Forward kinematics function
α	Gain in a CLIK algorithm
$\sigma_1, \ldots, \sigma_n$	Jacobian singular values
m	Number of joints
n	Task space dimension
$\mathbf{u}_c$	Robot control torque
$\mathbf{u}_e$	External torque acting on the robot
$\mathbf{u}_{PD}$	Proportional-derivative torque
$\mathbf{K}_P$	Proportional gain matrix
$\mathbf{K}_D$	Derivative gain matrix
$\mathbf{M}(\mathbf{q})$	Robot joint inertia matrix
$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$	Coriolis and centripetal forces
$\mathbf{F}_f$	Robot joint friction
$\mathbf{G}(\mathbf{q})$	Robot's gravity forces
$\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})$	Coriolis, centripetal, friction and gravity forces
π	Policy
$\boldsymbol{\Theta}$	Policy parameter space
$\boldsymbol{\theta} \in \boldsymbol{\Theta}$	Policy parameters
$\boldsymbol{\tau}$	Trajectory
$\mathbf{y}$	Robot state variable
$\mathbf{x}$	Robot's latent space state variable
$\mathbf{s}$	Contextual variable

A. Colomé and C. Torras, *Reinforcement Learning of Bimanual Robot Skills*, Springer Tracts in Advanced Robotics 134,
https://doi.org/10.1007/978-3-030-26326-3

$\mathbf{u}$	Action taken by the policy π
$\mathbf{R}$	Reward function
$\mathbf{J}_{\pi_\theta}$	Expected reward under policy π
$\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega)$	Sample from a normally distributed policy with mean $\boldsymbol{\mu}_\omega$ and covariance $\boldsymbol{\Sigma}_\omega$
λ, η	Weighting temperatures for PI2 and REPS, respectively
N_t	Number of timesteps of a trajectory
N_f	Number of Gaussian kernels used per DoF
N_k	Number of trajectories (rollouts) used per policy update
$\mathbf{y}_g$	DMP goal position
$\tau, \alpha_z, \beta_z, \alpha_x$	DMP fixed parameters
x	DMP phase variable
$\mathbf{f}(x) = \mathbf{f}_t$	DMP excitation function
$\mathbf{g}(x)$	N_f-dimensional vector of DMP Gaussian kernels
$\boldsymbol{\psi} = \mathbf{I}_d \otimes \mathbf{g}(x)$	$d \times dN_f$ matrix of Gaussian kernels
$\boldsymbol{\Omega}$	Robot's DoF coupling matrix
$\boldsymbol{\Sigma}_y$	ProMP fitting and system noise
$\boldsymbol{\Sigma}_f$	DMP fitting and system noise