A Programming by Demonstration Model combining Statistics, Dynamical Systems and Optimal Control

Technical Report

Sylvain Calinon[1,2] and Danilo Bruno[2]

[1]Idiap Research Institute, Martigny, Switzerland.
[2]Department of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), Genova, Italy.
sylvain.calinon@idiap.ch, danilo.bruno@iit.it

October 2014

Abstract

This report comes along the *C++* and *Matlab/GNU Octave* source codes available at
`http://programming-by-demonstration.org/lib/`
It aims at presenting a set of learning from demonstration approaches in a step-by-step didactic style, with more detailed equations than in the publications in which the proposed approaches were originally presented. The report acts as a companion paper to interpret the implemented functionalities, with comments throughout the source codes that directly refer to the equations presented here. The report does not present experiments or comparisons, but directs the readers to publications showcasing or testing the approaches detailed in the document.

Contents

1 Introduction

Statistical machine learning, dynamical systems and optimal control can be combined to provide a robot with the capability to generalize an observed skill to new situations by generating movements that are natural, efficient and safe for the surrounding users.

The core idea of the proposed approach is to represent an observed movement (or skill) as the trajectory (or behavior) of a virtual spring-damper system pulling the system, where a Gaussian mixture model is used

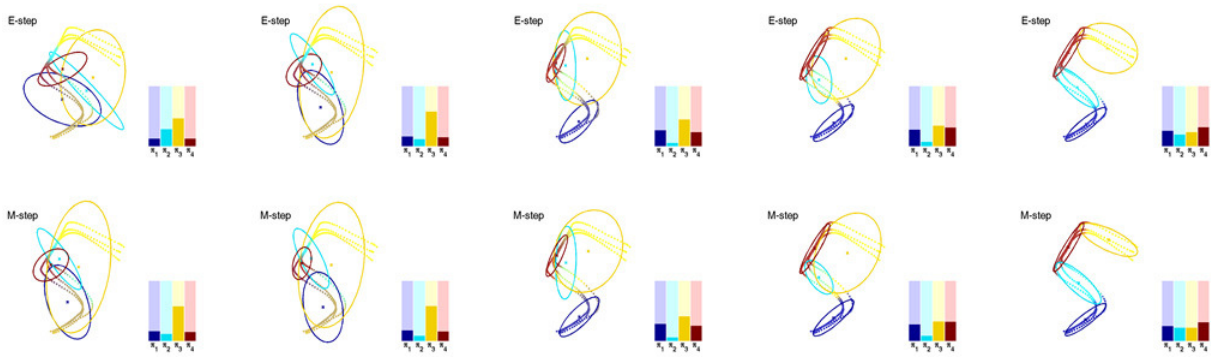


Figure 1: Illustration of the expectation-maximization (EM) process, in the case of random initialization of the GMM parameters. In practice, it is recommended to start EM from a coarse estimate of the parameters. For example, based on an equal split in time of motion segments, based on a geometric segmentation with k-means [MacQueen, 1967], based on moments or spectral approaches with circular covariances [Shi et al., 2009, Kulis and Jordan, 2012, Hsu and Kakade, 2013], or based on an iterative clustering algorithm [Scott and Szewczyk, 2001].

to encode the path, and where the variability is used to infer the impedance parameters of the system. These impedance parameters figuratively correspond to the stiffness of a spring and to the damping parameter of a viscous damper, with the difference that they can also be full stiffness and damping matrices in the cases we will consider. The model shares links with optimal feedback control strategies in which deviations from an average trajectory are corrected only when they interfere with task performance, such as in the *minimal intervention principle* [Todorov and Jordan, 2002, Wolpert et al., 2011].

The model will then be extended to its task-parameterized version, in which several frames of reference are interacting to describe the path of virtual springs in several coordinate systems, where the variations from each of these observers will be used to determine how strong these springs should be. In this model, the predicted task variations and couplings are thus exploited to regulate the impedance of virtual spring-damper systems acting in several frames of reference.

We will use the term *model parameters* to refer to the learned parameters of a model describing the movement or skill behaviors. When the model requires an adaptation to external parameters describing the current situation, such as position of objects or users, changes in the environment and changes of configurations of another robot parts, we will refer to these parameters as *task parameters*, which are used as input to the system and that will transform the learned *model parameters* accordingly to the current situation.

The different functionalities that compose the proposed approach can be used interdependently, and the order in which they are presented does not matter. Sections 2 and **??** first describe the core encoding and regression approaches being employed. Section 4 describes the use of these approaches within a dynamical systems strategy. Section 5 describes the optimal control strategy used to regulate the impedance behaviors during the reproduction of the learned skills.

2 Gaussian mixture model (GMM)

Model-based clustering is a widespread approach renowned for its probabilistic foundation and flexibility [McLachlan and Peel, 2000]. Clustering aims to group data into several homogeneous groups. Partitioning data can have various purpose, such as assigning a label to each cluster for classification purpose. In our case, we are mostly interested in clustering data to observe and encode their local trend, instead of trying to interpret the overall behavior of the data. We later want to exploit this partitioning to regenerate new data that can generalize and share the same characteristics as the training set.

Earliest approaches in model-based clustering were based on geometric procedures, such as the *k-means* algorithm [MacQueen, 1967]. Later, the use of statistical frameworks allowed researchers to formalize the notion of clusters to probability distributions, with the advantage of being able to interpret the obtained partitions from a statistical point of view.

The observations $\{\boldsymbol{\xi}_t\}_{t=1}^T$ are assumed to be independent realizations of a random process whereas the unobserved labels $\{z_t\}_{t=1}^T$ are assumed to be independent realizations of a random variable $z_t \in \{1, \dots, K\}$. The set of pairs $\{\boldsymbol{\xi}_t, z_t\}_{t=1}^T$ compose the complete data set. By denoting by $\mathcal{P}$ the probabilistic density function, a finite mixture model is described as

$$\mathcal{P}(\boldsymbol{\xi}_t) = \sum_{i=1}^{K} \pi_i \, f_i(\boldsymbol{\xi}_t), \tag{2.0.1}$$

where π_i and f_i represent respectively the mixture proportion and the conditional density function of the k-th mixture component, with the constraint that $\sum_{i=1}^K \pi_i = 1$. The clusters are in practice often modeled by the

same parametric density function. The conditional density function is then defined as $f_i(\boldsymbol{\xi}_t|\boldsymbol{\theta}_i)$, where $\boldsymbol{\theta}_i$ are parameters for the i-th mixture component.

Among the possible parametric density functions for the mixture components, the Gaussian distribution is often employed for theoretical and computational reasons that we will see next. In this case,

$$\mathcal{P}(\boldsymbol{\xi}_t) = \sum_{i=1}^{K} \pi_i \, \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2.0.2}$$

$$\text{with} \quad \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \exp\Big(-\frac{1}{2}(\boldsymbol{\xi}_t - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{\xi}_t - \boldsymbol{\mu}_i)\Big), \tag{2.0.3}$$

where $|\boldsymbol{\Sigma}_i|$ denotes the determinant of the matrix $\boldsymbol{\Sigma}_i$.

The parameters of a *Gaussian mixture model* (GMM) with K components are thus defined by $\{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$, where π_i is the prior (mixing coefficient), $\boldsymbol{\mu}_i$ is the center, and $\boldsymbol{\Sigma}_i$ is the covariance matrix of the i-th mixture component.

The GMM representation is very old, see e.g., [Pearson, 1894]. One of the first use of the representation might have been to measure populations of crabs in Naples, where it was observed that the skewness of the data could not be well approximated by a single symmetric Gaussian [McLachlan and Peel, 2000]. In [Pearson, 1894], Pearson fit two Gaussians with a moment-based approach that required to solve a polynomial of degree 9, revealing that there was not only one but two species of crabs that were being observed.

While the core representation did not change much since then, the learning algorithms changed through the years, exploiting the mathematical and computational solutions currently available. The moment approach was progressively replaced by algorithmic iterative solutions when computers became available (with a recent interesting regain of popularity with work such as [Hsu and Kakade, 2013]).

Despite its tendency to sometimes look old-fashioned, the estimation of mixture parameters is still today a rich research topic. The estimation can nowadays exploit a richer set of machine learning tools such as spectral analysis [Shi et al., 2009, Ng et al., 2001], quantum computing [Tanaka and Tsuda, 2008, Tsuda, 2009, Warmuth and Kuzmin, 2010], parallel GPU processing [Kumar et al., 2009, Srinivasan et al., 2010] or spanning trees with iterative pairwise replacement [Scott and Szewczyk, 2001].

At the present time, the approach that is the most often applied in practice is the *expectation-maximization* (EM) algorithm [Dempster et al., 1977]. Although it is not specifically dedicated to mixture models, EM is probably the most popular technique for inferring mixture models. EM is applied to problems with missing variables. In the case of GMM, the missing variables in the training set are the group memberships $\{z_t\}_{t=1}^T$.

For the set of observations $\boldsymbol{\xi} = \{\boldsymbol{\xi}_t\}_{t=1}^T$, the log-likelihood of the GMM is

$$\mathcal{L}(\boldsymbol{\theta}|\boldsymbol{\xi}) = \sum_{t=1}^{T} \log \left(\sum_{i=1}^{K} \pi_i \, \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \right). \tag{2.0.4}$$

The inference of this model cannot be directly done through the maximization of the likelihood since the group labels $\{z_t\}_{t=1}^T$ of the observations are unknown. However, trying to zero the derivative of the above log-likelihood results in an EM process iteratively refining the model parameters and converging to a local optimum of the likelihood. The two steps are described below.

E-step:

$$h_{t,i} = \frac{\pi_i \, \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{k=1}^{K} \pi_k \, \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}. \tag{2.0.5}$$

M-step:

$$\pi_i = \frac{\sum_{t=1}^{T} h_{t,i}}{T}, \tag{2.0.6}$$

$$\boldsymbol{\mu}_i = \frac{\sum_{t=1}^{T} h_{t,i} \boldsymbol{\xi}_t}{\sum_{t=1}^{T} h_{t,i}}, \tag{2.0.7}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_{t=1}^{T} h_{t,i} (\boldsymbol{\xi}_t - \boldsymbol{\mu}_i)(\boldsymbol{\xi}_t - \boldsymbol{\mu}_i)^\top}{\sum_{t=1}^{T} h_{t,i}}. \tag{2.0.8}$$

These two steps are iteratively computed until a stopping criterion is satisfied. The update step above is described for the case of full covariance matrices, which is the best representation for the approaches that we will describe next. For other applications, equivalent updates can be defined for diagonal or circular matrices, which reduce the complexity of the solution space.

Fig. 1 shows an example of the two steps of the iterative procedure. A relevant property of EM is that it guarantees to improve the likelihood function at each iteration, converging to a local optimum [Wu, 1983]. For applications requiring to learn the task while executing it, an online formulation of EM can also be derived [Neal and Hinton, 1999, Sato and Ishii, 2000, Mongillo and Deneve, 2008].

2.1 Regularization of the GMM parameters

In practice, it can be relevant to introduce regularization terms in some applications where an ill-posed problem or overfitting could potentially arise. Regularization has the effect of avoiding singularities and smoothing the solution space.

One way to do this is to define a minimal admissible eigenvalue $\lambda_{\min}$ and adjust each covariance matrix $\boldsymbol{\Sigma}$ so that

$$\boldsymbol{\Sigma} \leftarrow \boldsymbol{V}\tilde{\boldsymbol{D}}\boldsymbol{V}^{\top}, \quad \text{with} \quad \tilde{\boldsymbol{D}} = \begin{bmatrix} \tilde{\lambda}_1^2 & 0 & \cdots & 0 \\ 0 & \tilde{\lambda}_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \tilde{\lambda}_d^2 \end{bmatrix}, \quad \text{where} \quad \tilde{\lambda}_i = \max(\lambda_i, \lambda_{\min}) \quad \forall i \in \{1, \dots, d\}, \tag{2.1.1}$$

where $\boldsymbol{V}$ is matrix containing the stacked eigenvectors of $\boldsymbol{\Sigma}$, with λ_i the corresponding eigenvalues.

Another simpler approach is to set *a priori* uncertainties on the covariance parameters in the form of a diagonal isotropic covariance $\boldsymbol{I}\rho$ (Tikhonov regularization), so that

$$\boldsymbol{\Sigma} \leftarrow \boldsymbol{\Sigma} + \boldsymbol{I}\rho, \tag{2.1.2}$$

with $\boldsymbol{I}$ an identity matrix and ρ a small scalar factor that can be either set empirically or estimated from the data.

The difference with (2.1.1) is that a value ρ is added to each λ_i instead of truncating the eigenvalues. Note that the same development can be done with singular value decomposition, emphasizing the effect of the regularization on the condition number, by forcing it to be higher than a threshold as in (2.1.1) or by increasing the singular values as in (2.1.2).

Note that it is sometimes relevant to apply different forms of regularization at different steps of the procedure (e.g., at each iteration of the EM process and after).

2.2 Subspace clustering and parsimonious Gaussian mixture model

In some applications, classical model-based clustering tend to perform poorly in high-dimensional spaces if too few datapoints are available. This is the case for problems aiming at encoding multivariate and multimodal signals from a few number of demonstrations. Namely, if the training set is $\{\boldsymbol{\xi}_t\}_{t=1}^T$ with $\boldsymbol{\xi}_t \in \mathbb{R}^D$, the *curse of dimensionality* occurs if the dimension of the data D is too large compared to the size of the training set T. In particular, the problem can affect the full covariances $\boldsymbol{\Sigma}_i \in \mathbb{R}^{D \times D}$ because the number of parameters to be estimated quadratically grows with D.

Bouveyron and Brunet reviewed various ways of viewing the problem and coping with high-dimensional data in clustering problems [Bouveyron and Brunet, 2014]. Three principal viewpoints can be considered in practice:

1. Since D is too large compared to T, a global dimensionality reduction should be applied as a pre-processing step to reduce D.

2. Since D is too large compared to T, the solution space contains many poor local optima; the solution space can be smoothed and simplified by introducing ridge or lasso regularization in the estimation of the covariance, avoiding numerical problem and singular solutions when inverting the covariances. For example, a simple form of regularization can be achieved by setting $\boldsymbol{\Sigma}_i \leftarrow \boldsymbol{\Sigma}_i + \boldsymbol{I}\sigma$ after the maximization step of each EM loop, with σ a regularization term and $\boldsymbol{I}$ the identity matrix.

3. Since D is too large compared to T, the model is probably over-parametrized, and a more parsimonious model should be used, that is, estimating a fewer number of parameters.

An example in the last category would be to consider spherical or diagonal covariances instead of full matrices. for the proposed approach, such constraint would be too strong and would lose important synergistic information. There are, however, a wide range of alternatives in-between the encoding of diagonal and full covariances. These alternatives can be rephrased as a subspace clustering problem, that aims at grouping the data such that they can be locally projected in a subspace of reduced dimensionality, thus helping the analysis of the local trend while reducing the number of parameters to be estimated.

Many possible constraints can be considered, grouped in families such as parsimonious GMM [Bouveyron and Brunet, 2014,McNicholas and Murphy, 2008,Baek et al., 2010], *mixtures of factor analyzers* (MFA) [McLachlan et al., 2003] or *mixtures of probabilistic principal component analyzers* [Tipping and Bishop, 1999].

Parsimonious Gaussian mixture model

A parsimonious GMM considers the spectral decomposition of the covariances

$$\boldsymbol{\Sigma}_i = \lambda_i \boldsymbol{V}_i \boldsymbol{D}_i \boldsymbol{V}_i^{\top}, \tag{2.2.1}$$

with $\boldsymbol{V}_i$ a matrix of ordered eigenvectors determining the orientation of the cluster, $\boldsymbol{D}_i$ a diagonal matrix proportional to the ordered eigenvalues determining the shape of the cluster, and λ_i a scalar determining its volume. Constraints can then be set by setting some of these elements as shared among the clusters, and by keeping only the first d eigenvectors and eigenvalues in the parameterization.

Mixture of factor analyzers

If $\{\boldsymbol{\xi}_t\}_{t=1}^T$ are considered as random and independent realizations of $\boldsymbol{Y} \in \mathbb{R}^D$, MFA considers that $\boldsymbol{Y}$ can be expressed from an unobserved factor $\boldsymbol{X} \in \mathbb{R}^d$, with $d<D$.

The unobserved labels $\{z_t\}_{t=1}^T$ are assumed to be independent realizations of a random variable Z, where $z_t=i$ indicates that $\boldsymbol{\xi}_t$ is generated by the i-th factor analyzer. The relationship between these two spaces is assumed to be linear, conditionally to Z

$$\boldsymbol{Y}|_{Z=i} \quad = \quad \boldsymbol{\Lambda}_i \boldsymbol{X} + \boldsymbol{\mu}_i + \boldsymbol{\epsilon}, \tag{2.2.2}$$

where $\boldsymbol{\Lambda}_i$ is a $D \times d$ matrix and $\boldsymbol{\mu}_i \in \mathbb{R}^D$ is the mean vector of the i-th factor analyzer. $\boldsymbol{\epsilon} \in \mathbb{R}^D \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Psi}_i)$ is assumed to be a centered Gaussian noise with diagonal covariance $\boldsymbol{\Psi}_i$.

Besides, it is assumed that $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, implying that the conditional distribution of $\boldsymbol{Y}$ is also Gaussian, namely,

$$\boldsymbol{Y}|\boldsymbol{X}, Z=i \quad \sim \quad \mathcal{N}(\boldsymbol{\Lambda}_i \boldsymbol{X}+\boldsymbol{\mu}_i, \boldsymbol{\Psi}_i). \tag{2.2.3}$$

The marginal density of $\boldsymbol{Y}$ is thus a GMM with

$$\boldsymbol{\Sigma}_i = \boldsymbol{\Lambda}_i \boldsymbol{\Lambda}_i^\top + \boldsymbol{\Psi}_i. \tag{2.2.4}$$

Similarly to parsimonious GMM, the covariances can be constrained by fixing d or by sharing elements among the mixture components (to re-use previously discovered synergies in other parts of the skill). For each approach, a dedicated EM update can be derived corresponding to the considered type of constraints. This subspace clustering method can reconstruct estimates of the full covariances, which is an important requirement for the proposed approach.

2.3 Wrapped GMM to encode periodic signals

Several distributions were proposed to handle the periodicity of a movement, where the problem is often referred to as circular statistics [Mardia and Jupp, 1999, Agiomyrgiannakis and Stylianou, 2009]. We will focus here on the *wrapped GMM* (WGMM) that is directly compatible with the proposed approach.

The first step is to define a tiling (or lattice) that will be used to compute probabilities in regions adjacent to the range of periodic variables. This tiling is represented by a series of offset vectors $\{\boldsymbol{v}_m\}_{m=1}^M$ determining the neighboring regions where likelihood estimation should take place.

A variable $u_j \in \mathbb{N}$ is used to define the tiling for each dimension, taking a null value for non-periodic dimensions, and a positive integer value for periodic dimensions. This integer determines the adjacent regions where the likelihood will be evaluated. The tiling for dimension $j \in \{1, \dots, D\}$ is represented as a vector of offsets $\boldsymbol{o}_j \in \mathbb{R}^{O_j}$ of dimension $O_j=1+2u_j$ defined by

$$\boldsymbol{o}_j = \big[-u_j,\ -u_j+1,\ \dots,\ 0,\ \dots,\ u_j-1,\ u_j \big]\ r_j, \tag{2.3.1}$$

where r_j represents the period for dimension j (e.g., 2π). For mixture modeling problems, the eigenvalues of the covariance of each component will typically be lower than the range 2π, and u_j can be set to 1 or 2 without losing precision on the estimate. For other problems with large variance compared to the periodicity, u_j can still be set to a larger integer if a very precise estimate is required, at the expense of a slower processing time. With the above notation, the complete tiling is defined by $M = \prod_{j=1}^D O_j$ vectors $\boldsymbol{v}_m$ denoting a point on the lattice formed by offset vectors $\boldsymbol{o}_j$.

For example, for a dataset of 3 dimensions with $\boldsymbol{u}=[1\ 0\ 0]^\top$ (the first dimension denotes a phase variable and one adjacent region is considered in the calculation of probabilities), we have $\boldsymbol{o}_1=[-1,\ 0,\ 1]r_1$, $\boldsymbol{o}_2=[0]r_2$, $\boldsymbol{o}_3=[0]r_3$ (with dimensions $O_1=3$, $O_2=1$, $O_3=1$ and $M=O_1O_2O_3=3$). The vectors $\{\boldsymbol{v}_m\}_{m=1}^3$ will then be

$$\boldsymbol{v}_1 = \begin{bmatrix} -r_1 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{v}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{v}_3 = \begin{bmatrix} r_1 \\ 0 \\ 0 \end{bmatrix}. \tag{2.3.2}$$

We can observe that the rows of the lattice are null for the dimensions that are not periodic.

A WGMM of parameters $\boldsymbol{\Theta} = \{\pi_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$ with K Gaussian components is defined as

$$\mathcal{P}(\boldsymbol{\xi}_t|\boldsymbol{\Theta}) \quad = \quad \sum_{i=1}^K \pi_i\, \mathcal{N}^W(\boldsymbol{\xi}_t|\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2.3.3}$$

$$\text{with} \quad \mathcal{N}^W(\boldsymbol{\xi}_t|\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad = \quad \sum_{m=1}^M \mathcal{N}(\boldsymbol{\xi}_t - \boldsymbol{v}_m|\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i). \tag{2.3.4}$$

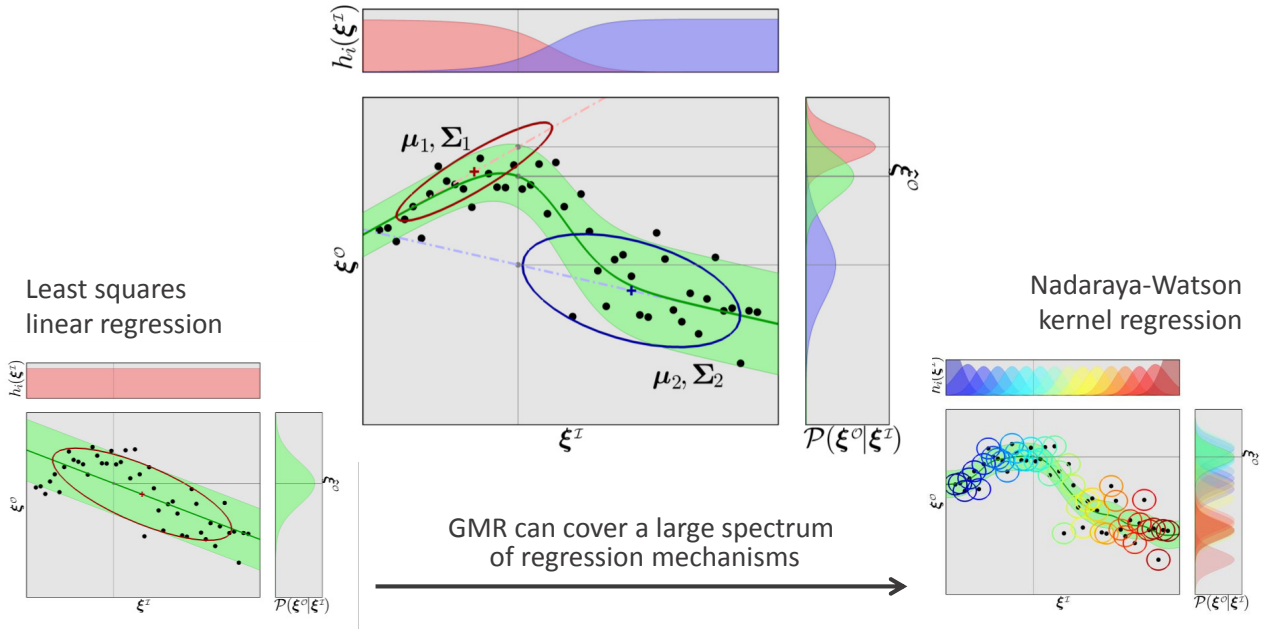


Figure 2: Illustration of the encoding of $\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}})$ as a Gaussian mixture model (GMM) with two components, and estimation of $\mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}})$ with Gaussian mixture regression (GMR), where both $\boldsymbol{\xi}^{\mathcal{I}}$ and $\boldsymbol{\xi}^{\mathcal{O}}$ can be multidimensional. The model can emulate a large spectrum of regression mechanisms, from standard linear regression (when a single component $K=1$ is used), to non-linear kernel regression with a Gaussian centered on each datapoint ($K=N$). Model selection (estimating the number of Gaussians in the GMM) shares the same core challenge as that of sparse kernel regression techniques requiring to find the right bandwidth parameters to select a subset of existing/new datapoints that are the most representatives of the dataset. Several approaches have been proposed for model selection in GMM, for example, based on *Bayesian information criterion* [Schwarz, 1978], *Dirichlet process* [Rasmussen, 2000], *iterative pairwise replacement* [Scott and Szewczyk, 2001] or *spectral clustering* [Ng et al., 2001, Shi et al., 2009, Kulis and Jordan, 2012].

Eq. (2.3.4) consists of replicating the Gaussian function on the lattice. We can observe that it collapses to a standard GMM by setting $\boldsymbol{u}=\mathbf{0}$ (dataset with only non-periodic variables), yielding $M=1$ and $\boldsymbol{v}_1=\mathbf{0}$.

The expectation-maximization process to estimate $\boldsymbol{\Theta}$ is computed as

E-step:

$$h_{t,m,i} = \frac{\pi_i \, \mathcal{N}(\boldsymbol{\xi}_t - \boldsymbol{v}_m | \, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_k^K \pi_k \, \mathcal{N}(\boldsymbol{\xi}_t - \boldsymbol{v}_m | \, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}. \tag{2.3.5}$$

M-step:

$$\pi_i = \frac{\sum_{t=1}^{T} \sum_{m=1}^{M} h_{t,m,i}}{T}, \tag{2.3.6}$$

$$\boldsymbol{\mu}_i = \frac{\sum_{t=1}^{T} \sum_{m=1}^{M} h_{t,m,i} \, (\boldsymbol{\xi}_t - \boldsymbol{v}_m)}{\sum_{t=1}^{T} \sum_{m=1}^{M} h_{t,m,i}}, \tag{2.3.7}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_{t=1}^{T} \sum_{m=1}^{M} h_{t,m,i} \, (\boldsymbol{\xi}_t - \boldsymbol{\mu}_i - \boldsymbol{v}_m)(\boldsymbol{\xi}_t - \boldsymbol{\mu}_i - \boldsymbol{v}_m)^\top}{\sum_{t=1}^{T} \sum_{m=1}^{M} h_{t,m,i}}. \tag{2.3.8}$$

In practice, the tiling process used in the above can be computed very efficiently by replacing the loops in the equations above by matrices operations, which can speed up the computation by several orders of magnitude by relying on efficient linear algebra routines such as LAPACK/BLAS (used for example by Matlab or GNU Octave).

3 Gaussian mixture regression (GMR)

With a GMM representation, the reproduction of a reference movement or an average skill behavior can be formalized as a regression problem [Ghahramani and Jordan, 1994]. We showed that in robotics, *Gaussian mixture regression* (GMR) offers a simple solution to handle encoding, recognition, prediction and reproduction in robot learning [Calinon et al., 2007, Calinon, 2009, Calinon et al., 2010a]. GMR relies on basic properties of normal distributions such as linear transformation and Gaussian conditioning. It provides a probabilistic

representation of movements or policies, in which the model can compute the next actions on-the-fly, with a computation time that is independent of the number of datapoints used to train the system.

In contrast to other regression methods such as *Locally Weighted Regression* (LWR) [Schaal and Atkeson, 1998], *Locally Weighted Projection Regression* (LWPR) [Vijayakumar et al., 2005], or *Gaussian Process Regression* (GPR) [Nguyen-Tuong and Peters, 2008, Grimes et al., 2006], GMR does not model the regression function directly, but models the joint probability density function of the data. It then derives the regression function from the joint density model. This can be an advantage in robot applications where the input and output components are only specified at the very last step of the process, if input information are missing at some iterations, or if it is not relevant to retrieve the whole set of outputs (e.g. to predict or react as rapidly as possible). Density estimation can thus be learned in an off-line phase (with the GMM estimation machinery presented in the previous section), thus allowing the regression process to be computed very rapidly.

The superscripts $\mathcal{I}$ and $\mathcal{O}$ will be further used to describe the sets of dimensions that span for input and output variables (that will be used as exponents in vectors and matrices).

At each iteration step t, the datapoint $\boldsymbol{\xi}_t$ can be decomposed as two subvectors $\boldsymbol{\xi}_t^{\mathcal{I}}$ and $\boldsymbol{\xi}_t^{\mathcal{O}}$ spanning for the input and output variables. For trajectory encoding in task space, $\mathcal{I}$ corresponds to the time input dimension, and $\mathcal{O}$ corresponds to the output dimensions describing a path in task space (e.g., position).

With this notation, a block decomposition of the datapoint $\boldsymbol{\xi}_t$, vectors $\boldsymbol{\mu}_i$ and matrices $\boldsymbol{\Sigma}_i$ can be written as

$$\boldsymbol{\xi}_t = \begin{bmatrix} \boldsymbol{\xi}_t^{\mathcal{I}} \\ \boldsymbol{\xi}_t^{\mathcal{O}} \end{bmatrix}, \quad \boldsymbol{\mu}_i = \begin{bmatrix} \boldsymbol{\mu}_i^{\mathcal{I}} \\ \boldsymbol{\mu}_i^{\mathcal{O}} \end{bmatrix}, \quad \boldsymbol{\Sigma}_i = \begin{bmatrix} \boldsymbol{\Sigma}_i^{\mathcal{I}} & \boldsymbol{\Sigma}_i^{\mathcal{I}\mathcal{O}} \\ \boldsymbol{\Sigma}_i^{\mathcal{O}\mathcal{I}} & \boldsymbol{\Sigma}_i^{\mathcal{O}} \end{bmatrix}. \tag{3.0.1}$$

The GMM estimated in the previous section encodes the joint distribution $\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}}) \sim \sum_{i=1}^{K} \pi_i \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ of the dataset $\boldsymbol{\xi}$. At each reproduction step t, $\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{O}} | \boldsymbol{\xi}_t^{\mathcal{I}})$ is computed as the conditional distribution

$$\mathcal{P}(\boldsymbol{\xi}_t^{\mathcal{O}} | \boldsymbol{\xi}_t^{\mathcal{I}}) \sim \sum_{i=1}^{K} h_i(\boldsymbol{\xi}_t^{\mathcal{I}}) \, \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}_t^{\mathcal{I}}), \hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}}\Big), \tag{3.0.2}$$

$$\text{with} \quad \hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}_t^{\mathcal{I}}) = \boldsymbol{\mu}_i^{\mathcal{O}} + \boldsymbol{\Sigma}_i^{\mathcal{O}\mathcal{I}} \boldsymbol{\Sigma}_i^{\mathcal{I}-1} (\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}_i^{\mathcal{I}}), \tag{3.0.3}$$

$$\hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}} = \boldsymbol{\Sigma}_i^{\mathcal{O}} - \boldsymbol{\Sigma}_i^{\mathcal{O}\mathcal{I}} \boldsymbol{\Sigma}_i^{\mathcal{I}-1} \boldsymbol{\Sigma}_i^{\mathcal{I}\mathcal{O}}, \tag{3.0.4}$$

$$\text{and} \quad h_i(\boldsymbol{\xi}_t^{\mathcal{I}}) = \frac{\pi_i \mathcal{N}(\boldsymbol{\xi}_t^{\mathcal{I}} | \, \boldsymbol{\mu}_i^{\mathcal{I}}, \boldsymbol{\Sigma}_i^{\mathcal{I}})}{\sum_k^K \pi_k \mathcal{N}(\boldsymbol{\xi}_t^{\mathcal{I}} | \, \boldsymbol{\mu}_k^{\mathcal{I}}, \boldsymbol{\Sigma}_k^{\mathcal{I}})}. \tag{3.0.5}$$

Eq. (3.0.2) represents a multimodal distribution. There are problems in which a single peaked output distribution is preferred. In this case, Eq. (3.0.2) can also be approximated by a normal distribution.

Let us consider a datapoint $\boldsymbol{\xi}$ distributed as in Eq. (2.0.2), with $\mathcal{P}(\boldsymbol{\xi}) = \mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}})$ the joint distribution describing the data. The conditional probability of an output given an input is

$$\mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) = \frac{\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}})}{\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}})} = \frac{\sum_{i=1}^{K} \mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}} | z_i) \mathcal{P}(z_i)}{\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}})}, \tag{3.0.6}$$

where z_i represents the i-th component of the GMM. Namely,

$$\mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) = \sum_{i=1}^{K} \mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}, z_i) \frac{\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}} | z_i) \mathcal{P}(z_i)}{\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}})} = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}}) \, \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}^{\mathcal{I}}), \hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}}\Big). \tag{3.0.7}$$

The conditional mean can be computed as

$$\hat{\boldsymbol{\mu}}^{\mathcal{O}} = E(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) = \int \boldsymbol{\xi}^{\mathcal{O}} \mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) \, d\boldsymbol{\xi}^{\mathcal{O}} = \int \boldsymbol{\xi}^{\mathcal{O}} \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}}) \, \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}^{\mathcal{I}}), \hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}}\Big) = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}}) \hat{\boldsymbol{\mu}}_i^{\mathcal{O}}. \tag{3.0.8}$$

In order to evaluate the covariance, we calculate

$$\text{cov}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) = E(\boldsymbol{\xi}^{\mathcal{O}} \boldsymbol{\xi}^{\mathcal{O}\top} | \boldsymbol{\xi}^{\mathcal{I}}) - E(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) E(\boldsymbol{\xi}^{\mathcal{O}\top} | \boldsymbol{\xi}^{\mathcal{I}}). \tag{3.0.9}$$

We have that

$$E(\boldsymbol{\xi}^{\mathcal{O}} \boldsymbol{\xi}^{\mathcal{O}\top} | \boldsymbol{\xi}^{\mathcal{I}}) = \int \boldsymbol{\xi}^{\mathcal{O}} \boldsymbol{\xi}^{\mathcal{O}\top} \mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}} | \boldsymbol{\xi}^{\mathcal{I}}) d\boldsymbol{\xi}^{\mathcal{O}} \tag{3.0.10}$$

$$= \int \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}}) \boldsymbol{\xi}^{\mathcal{O}} \boldsymbol{\xi}^{\mathcal{O}\top} \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}^{\mathcal{I}}), \hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}}\Big) d\boldsymbol{\xi}^{\mathcal{O}} \tag{3.0.11}$$

$$= \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}}) \int \boldsymbol{\xi}^{\mathcal{O}} \boldsymbol{\xi}^{\mathcal{O}\top} \mathcal{N}\Big(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\boldsymbol{\xi}^{\mathcal{I}}), \hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}}\Big) d\boldsymbol{\xi}^{\mathcal{O}}. \tag{3.0.12}$$

By using Eq. (3.0.9) with a Gaussian distribution, we obtain

$$E(\boldsymbol{\xi}^{\mathcal{O}}\boldsymbol{\xi}^{\mathcal{O}\top}|\boldsymbol{\xi}^{\mathcal{I}}) = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}})\hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}} + \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}})\ \hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}})^{\top}. \tag{3.0.13}$$

Combining (3.0.9) with (3.0.13) we finally have that (see also [Sung, 2004])

$$\hat{\boldsymbol{\Sigma}}^{\mathcal{O}} = \text{cov}(\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}}) = \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}})\hat{\boldsymbol{\Sigma}}_i^{\mathcal{O}} + \sum_{i=1}^{K} h_i(\boldsymbol{\xi}^{\mathcal{I}})\hat{\boldsymbol{\mu}}_i^{\mathcal{O}}(\hat{\boldsymbol{\mu}}_i^{\mathcal{O}})^{\top} - \hat{\boldsymbol{\mu}}^{\mathcal{O}}\hat{\boldsymbol{\mu}}^{\mathcal{O}\top}. \tag{3.0.14}$$

The output distribution can then be approximated by the Gaussian

$$\mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}}) = \mathcal{N}\Big(\boldsymbol{\xi}^{\mathcal{O}}|\ \hat{\boldsymbol{\mu}}^{\mathcal{O}}, \hat{\boldsymbol{\Sigma}}^{\mathcal{O}}\Big). \tag{3.0.15}$$

Eq. (3.0.15) can be computed in a very rapid manner from the model parameters, allowing online retrieval of data. The retrieved signal encapsulates variation and correlation information in the form of full covariance matrices. GMR has so far mostly been used in three manners:

1. as an autonomous system with $\boldsymbol{\xi}=[\boldsymbol{x}^{\top}, \dot{\boldsymbol{x}}^{\top}]^{\top}$, by learning $\mathcal{P}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ with a GMM, with $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ representing position and velocity of the system (either in task space or joint space), and by retrieving iteratively during reproduction a series of velocity commands by estimating $\mathcal{P}(\dot{\boldsymbol{x}}|\boldsymbol{x})$ with GMR [Calinon et al., 2010a, Khansari-Zadeh and Billard, 2011];

2. as time-indexed trajectories with $\boldsymbol{\xi}=[t, \boldsymbol{x}^{\top}]^{\top}$, by learning $\mathcal{P}(t, \boldsymbol{x})$ with a GMM, and retrieving $\mathcal{P}(\boldsymbol{x}|t)$ with GMR for each time step to reproduce a trajectory [Calinon et al., 2007]. The retrieved trajectory presents nice smoothness properties (infinitely differentiable).

3. as a probabilistic formulation of dynamic movement primitives (DMP) [Calinon et al., 2012].

Fig. 2 illustrates the process in the case of time-indexed trajectories. For regression, any subset of input-output dimensions can be selected, which can change if required at each iteration during reproduction. Expectations on the remaining dimensions can be computed within the control loop of the robot (in a very fast manner), corresponding to a convex sum of linear approximations (with weights varying non-linearly). It can for example handle different sources of missing data, as the system is able to consider any combination of input/output mappings during the retrieval phase. In terms of computation, the estimation of the model depends linearly on the number of datapoints, while the prediction is independent of this number, which makes the approach an interesting alternative to regression methods such as *Gaussian process regression* (GPR) [Rasmussen and Williams, 2006] whose processing grows with the size of the dataset. The other potential benefit is that both input and output variables can be multidimensional without modification of the model.

GMR can be viewed as a trade-off between a global and local approach in the sense that the placement and spread of the basis functions are learned, together with their response, as a soft partitioning problem through *expectation-maximization* (EM),[1] while the prediction is a weighted superposition of locally linear systems. The prediction provides information about the local variations allowed by the task, and about the correlations among the different output terms, thus enabling the extraction of local coordination patterns.

As shown in the previous section, if the application requires the encoding of high-dimension data from few observations, subspace learning techniques such as *mixtures of factor analyzers* (MFA) [McLachlan et al., 2003] can be used to locally reduce the dimensionality without modifying the representation (where full covariances are reconstructed from the MFA parameters). Common synergy information can be shared among the Gaussians with *parsimonious GMM* [Bouveyron and Brunet, 2014] (e.g., to re-use and adapt previously discovered coordination patterns). It can be combined with *Dirichlet processes* [Rasmussen, 2000] to automatically select the number of basis functions to be used in the model.

3.1 Correspondence of GMR with weighted least squares

For signals encoding, GMR provides a smooth transition between locally linear trends, together with an adaptive placement of the basis functions determining the regions of the local trends. After the estimation of a GMM, for a given set of basis functions, GMR corresponds to a weighted least squares estimate of degree 1. To show this, we first rewrite Eq. (3.0.3) as a linear system, and note that the conditional probability $\mathcal{P}(\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}})$ of a joint distribution $\mathcal{P}(\boldsymbol{\xi}^{\mathcal{I}}, \boldsymbol{\xi}^{\mathcal{O}})$ encoded as a Gaussian distribution yields another Gaussian with parameters

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}}} &= \boldsymbol{\mu}^{\mathcal{O}} + \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}(\boldsymbol{\xi}^{\mathcal{I}} - \boldsymbol{\mu}_t^{\mathcal{I}}) && (3.1.1)\\
&= \overbrace{\boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}}^{\boldsymbol{A}}\ \boldsymbol{\xi}^{\mathcal{I}} + \overbrace{\boldsymbol{\mu}^{\mathcal{O}} - \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\mu}_t^{\mathcal{I}}}^{\boldsymbol{b}}, && (3.1.2)\\
\boldsymbol{\Sigma}_{\boldsymbol{\xi}^{\mathcal{O}}|\boldsymbol{\xi}^{\mathcal{I}}} &= \boldsymbol{\Sigma}^{\mathcal{O}} - \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\Sigma}^{\mathcal{I}\mathcal{O}}. && (3.1.3)
\end{aligned}$$

[1] Competition/collaboration arises due to the weighting term $h_{t,i}$ in Eq. (2.0.5) summing over the influence of the other Gaussian components.

This result corresponds to a least-squares estimate of first degree. With the training set $\{\boldsymbol{\xi}_t^{\mathcal{I}}, \boldsymbol{\xi}_t^{\mathcal{O}}\}_{t=1}^T$, by defining the quadratic error $e = \sum_{t=1}^T (\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b})^\top (\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b})$, the estimates of $\boldsymbol{A}$ and $\boldsymbol{b}$ minimizing e are given by

$$\frac{\partial e}{\partial \boldsymbol{A}} = -2\sum_{t=1}^T (\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b})\boldsymbol{\xi}_t^{\mathcal{I}\top} = 0 \quad \Leftrightarrow \quad \boldsymbol{A}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\boldsymbol{\xi}_t^{\mathcal{I}\top} + \boldsymbol{b}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top} = \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top}, \tag{3.1.4}$$

$$\frac{\partial e}{\partial \boldsymbol{b}} = -2\sum_{t=1}^T (\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b}) = 0 \quad \Leftrightarrow \quad \boldsymbol{A}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}} + T\boldsymbol{b} = \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}, \tag{3.1.5}$$

represented in a matrix form as

$$\begin{bmatrix}\boldsymbol{A} & \boldsymbol{b}\end{bmatrix} \overbrace{\begin{bmatrix}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\boldsymbol{\xi}_t^{\mathcal{I}\top} & \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}} \\ \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top} & T\end{bmatrix}}^{\boldsymbol{M}} = \begin{bmatrix}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} & \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\end{bmatrix}. \tag{3.1.6}$$

By using the block matrices composition rule

$$\begin{aligned}
\boldsymbol{M} &= \begin{bmatrix}\boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22}\end{bmatrix}, \quad \text{with} \begin{cases}\boldsymbol{M}_{11}\in\mathbb{R}^{d\times d}, & \boldsymbol{M}_{12}\in\mathbb{R}^{d\times D} \\ \boldsymbol{M}_{21}\in\mathbb{R}^{D\times d}, & \boldsymbol{M}_{22}\in\mathbb{R}^{D\times D}\end{cases} \\
\Leftrightarrow \quad \boldsymbol{M}^{-1} &= \begin{bmatrix}\boldsymbol{I} & \boldsymbol{0} \\ -\boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21} & \boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{S}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M}_{22}^{-1}\end{bmatrix}\begin{bmatrix}\boldsymbol{I} & -\boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1} \\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix} \\
&= \begin{bmatrix}\boldsymbol{S}^{-1} & -\boldsymbol{S}^{-1}\boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1} \\ -\boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21}\boldsymbol{S}^{-1} & \boldsymbol{M}_{22}^{-1} + \boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21}\boldsymbol{S}^{-1}\boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}\end{bmatrix},
\end{aligned}$$

where $\boldsymbol{S} = \boldsymbol{M}_{11} - \boldsymbol{M}_{12}\boldsymbol{M}_{22}^{-1}\boldsymbol{M}_{21}$ is the Schur complement of matrix $\boldsymbol{M}$, Eq. (3.1.6) yields

$$\begin{bmatrix}\boldsymbol{A} & \boldsymbol{b}\end{bmatrix} = \begin{bmatrix}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} & \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\end{bmatrix} \overbrace{\begin{bmatrix}\boldsymbol{I} & \boldsymbol{0} \\ -\frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top} & \boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{C} & \boldsymbol{0} \\ \boldsymbol{0} & \frac{1}{T}\end{bmatrix}\begin{bmatrix}\boldsymbol{I} & -\frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}} \\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}}^{\boldsymbol{M}^{-1}}. \tag{3.1.7}$$

Since $\boldsymbol{\mu}^{\mathcal{I}} = \frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}$ and $\boldsymbol{\mu}^{\mathcal{O}} = \frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}$, we obtain

$$\begin{aligned}
\boldsymbol{A} &= \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top}\boldsymbol{C} - \frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top}\boldsymbol{C} = \Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big)\boldsymbol{C}, && (3.1.8) \\
\text{and} \quad \boldsymbol{b} &= -\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top}\boldsymbol{C}\boldsymbol{\mu}^{\mathcal{I}} + \boldsymbol{\mu}^{\mathcal{O}} + T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\boldsymbol{C}\boldsymbol{\mu}^{\mathcal{I}} && (3.1.9) \\
&= -\Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big)\boldsymbol{C}\boldsymbol{\mu}^{\mathcal{I}} + \boldsymbol{\mu}^{\mathcal{O}} = -\boldsymbol{A}\boldsymbol{\mu}^{\mathcal{I}} + \boldsymbol{\mu}^{\mathcal{O}}, && (3.1.10)
\end{aligned}$$

$$\text{with} \quad \boldsymbol{C} = \Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - \frac{1}{T}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top}\Big)^{-1} = \Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{I}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big)^{-1}. \tag{3.1.11}$$

Knowing that

$$\begin{aligned}
\boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}} &= \frac{1}{T}\sum_{t=1}^T (\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}})(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})^\top && (3.1.12) \\
&= \frac{1}{T}\Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - \sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top} - \boldsymbol{\mu}^{\mathcal{I}}\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}\top} + \sum_{t=1}^T \boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big) && (3.1.13) \\
&= \frac{1}{T}\Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top} + T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big) && (3.1.14) \\
&= \frac{1}{T}\Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{O}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{O}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big), && (3.1.15) \\
\boldsymbol{\Sigma}^{\mathcal{I}} &= \frac{1}{T}\Big(\sum_{t=1}^T \boldsymbol{\xi}_t^{\mathcal{I}}\boldsymbol{\xi}_t^{\mathcal{I}\top} - T\boldsymbol{\mu}^{\mathcal{I}}\boldsymbol{\mu}^{\mathcal{I}\top}\Big), && (3.1.16)
\end{aligned}$$

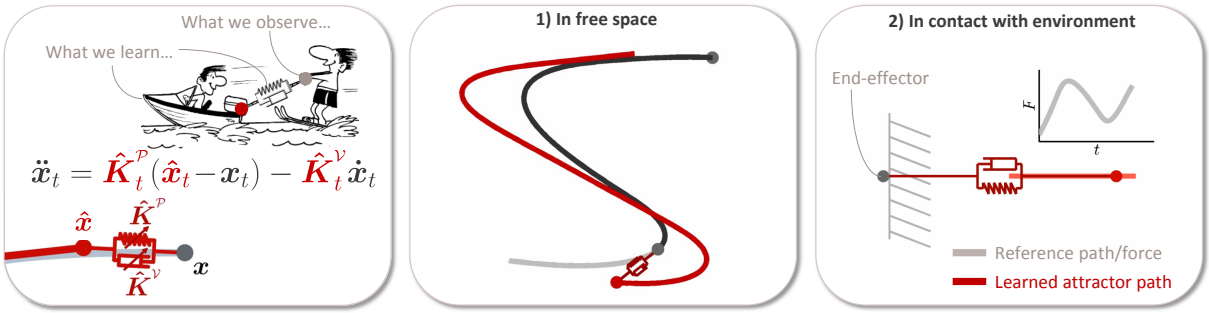


Figure 3: Illustration of the DS-GMR approach (see main text for details).

we obtain

$$\boldsymbol{A} = \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}, \tag{3.1.17}$$

$$\text{and} \quad \boldsymbol{b} = \boldsymbol{\mu}^{\mathcal{O}} - \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\mu}_t^{\boldsymbol{\xi}^{\mathcal{I}}}. \tag{3.1.18}$$

Since $\boldsymbol{\Sigma}^{\mathcal{I}\top} = \boldsymbol{\Sigma}^{\mathcal{I}}$ and $\boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}\top} = \boldsymbol{\Sigma}^{\mathcal{I}\mathcal{O}}$, the sum of squared residual covariance matrix can be formulated as

$$\boldsymbol{\Sigma}^{r} = \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b})(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{b})^{\top} \tag{3.1.19}$$

$$= \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{O}} + \boldsymbol{A}\boldsymbol{\mu}^{\mathcal{I}})(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{A}\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{O}} + \boldsymbol{A}\boldsymbol{\mu}^{\mathcal{I}})^{\top} \tag{3.1.20}$$

$$= \frac{1}{T}\sum_{t=1}^{T}\Big((\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}}) - \boldsymbol{A}(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})\Big)\Big((\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}}) - \boldsymbol{A}(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})\Big)^{\top} \tag{3.1.21}$$

$$= \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}})(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}})^{\top} + \boldsymbol{A}\Big(\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})^{\top}\Big)\boldsymbol{A}^{\top} \tag{3.1.22}$$

$$-\boldsymbol{A}\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}})^{\top} - \frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{\xi}_t^{\mathcal{O}} - \boldsymbol{\mu}^{\mathcal{O}})(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{\mu}^{\mathcal{I}})^{\top}\boldsymbol{A}^{\top} \tag{3.1.23}$$

$$= \boldsymbol{\Sigma}^{\mathcal{O}} + \Big(\boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\Big)\boldsymbol{\Sigma}^{\mathcal{I}}\Big((\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\Sigma}^{\mathcal{I}\mathcal{O}}\Big) - \Big(\boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\Big)\boldsymbol{\Sigma}^{\mathcal{I}\mathcal{O}} - \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}\Big((\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\Sigma}^{\mathcal{I}\mathcal{O}}\Big) \tag{3.1.24}$$

$$= \boldsymbol{\Sigma}^{\mathcal{O}} - \boldsymbol{\Sigma}^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}^{\mathcal{I}})^{-1}\boldsymbol{\Sigma}^{\mathcal{I}}. \tag{3.1.25}$$

We can see that (3.1.18) and (3.1.25) correspond to the conditional probability distribution in (3.0.15). A similar development can be followed for the mixture case (namely, for Gaussian mixture regression and weighted least squares).

3.2 Wrapped Gaussian mixture regression for periodic signals

In the case of periodic signals, the corresponding *wrapped GMR* can be defined as

$$\hat{\boldsymbol{\xi}}_t^{\mathcal{O}} = \sum_{i=1}^{K}\sum_{m=1}^{M} h_{t,m,i}(\boldsymbol{\xi}_t^{\mathcal{I}})\Big[\boldsymbol{\mu}_i^{\mathcal{O}} + \boldsymbol{\Sigma}_i^{\mathcal{O}\mathcal{I}}(\boldsymbol{\Sigma}_i^{\mathcal{I}})^{-1}\big(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{v}_m(\boldsymbol{\xi}_t^{\mathcal{I}}) - \boldsymbol{\mu}_i^{\mathcal{I}}\big)\Big], \tag{3.2.1}$$

where $h_{t,m,i}(\boldsymbol{\xi}_t^{\mathcal{I}})$ and $\boldsymbol{v}_m(\boldsymbol{\xi}_t^{\mathcal{I}})$ are based on the input variables dimensions, namely

$$h_{n,m,i}(\boldsymbol{\xi}_t^{\mathcal{I}}) = \frac{\pi_i\,\mathcal{N}\Big(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{v}_m(\boldsymbol{\xi}_t^{\mathcal{I}})|\,\boldsymbol{\mu}_i^{\mathcal{I}}, \boldsymbol{\Sigma}_i^{\mathcal{I}}\Big)}{\sum_k^K \pi_k\,\mathcal{N}\Big(\boldsymbol{\xi}_t^{\mathcal{I}} - \boldsymbol{v}_m(\boldsymbol{\xi}_t^{\mathcal{I}})|\,\boldsymbol{\mu}_k^{\mathcal{I}}, \boldsymbol{\Sigma}_k^{\mathcal{I}}\Big)}.$$

4 Extension to dynamical systems (DS-GMR)

The encoding of rich motor skills requires representations robust to various sources of perturbation, continuously arising from the environment, from the user, and from the robot. To tackle such challenge, the combination of dynamical systems and statistical models appear to be a suitable candidate. Several routes of research have been pursued in this direction, by either relying on simple point attractor and complex dynamics, see e.g., [Khansari-Zadeh and Billard, 2011], or by relying on more complex attractor manifolds and simpler dynamics. We rely

here on the second approach, with trajectory attractors and by constraining the dynamics to be of a form similar to spring-damper systems but with full stiffness and damping matrices.

Hogan and Sternad show in [Hogan and Sternad, 2013] that various forms of attractors can be considered to represent human motor skills. They provide manifold attractor examples such as trajectory attractor, synergy attractor, or impedance attractor, which might be thought as some forms of primitives that the central nervous system will momentarily try to come back to if a perturbation occurs. They also mention the potential existence of more complex forms of attractors such as fractal geometry attractors in chaotic dynamics.

With simple point attractor, a popular approach in robotics is the *dynamic movement primitives* (DMP) model [Ijspeert et al., 2013]. In DMP, a forcing term for each dimension of the movement modulates a spring-damper system centered on a target point, where the different forcing terms are synchronized by another dynamical system (usually acting as a decay term). After converting an observed movement into forcing term trajectories, and after setting a set of basis functions sequentially activated through the decay term, the learning task consists of individually approximating the forcing term profiles for each dimension.

Although the DMP formulation does not restrict the way in which forcing terms are learned [Ijspeert et al., 2013], most work relied in practice on *locally weighted regression* to train the model parameters with predefined basis functions (e.g., equal bandwidth and equal interval spacing). The forcing terms in such dynamical systems can however be learned by other learning strategies.

We introduced in [Calinon et al., 2012] a probabilistic formulation of dynamic movement primitives based on the encoding the joint evolution of the input (decay term) and the output (forcing terms) within a multivariate GMM, where GMR is then used to retrieve at each iteration the forcing terms corresponding to the current input (either time-dependent or time-invariant). The advantage is that the movement is not considered as a set of univariate outputs as in standard DMP. Instead, it also models local correlations among outputs, but also among inputs, and in-between inputs-outputs (e.g., to discover and re-use local sensorimotor patterns or synergies). It is also proposed in [Calinon et al., 2012] to replace the open-loop forcing terms formulation with a moving spring-damper system producing the equivalent force profile, thus requiring the system to learn the path of a virtual attractor instead of a force profile. The evolution of a decay term can for example be used as input of the system by encoding the evolution of this decay term together with the path of the virtual spring-damper system in a GMM, which can then be used with GMR to retrieve a generalized path from a decay term profile used as input. Alternatively, time stamps starting at the onset of the motion can be used as input to the system.

If $\boldsymbol{K}_t^{\mathcal{P}}$ and $\boldsymbol{K}_t^{\mathcal{V}}$ are stiffness and damping matrices, the evolution of the system is described by

$$\ddot{\boldsymbol{x}}_t = \boldsymbol{K}_t^{\mathcal{P}}(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) - \boldsymbol{K}_t^{\mathcal{V}}\dot{\boldsymbol{x}}_t, \tag{4.0.1}$$

where $\hat{\boldsymbol{x}}_t$ denotes the path of the virtual attractor.

By observing the evolution of the robot with position $\boldsymbol{x}_t$, velocity $\dot{\boldsymbol{x}}_t$ and acceleration $\ddot{\boldsymbol{x}}_t$, the evolution of the attractor $\hat{\boldsymbol{x}}_t$ can reversely be computed as

$$\hat{\boldsymbol{x}}_t = (\boldsymbol{K}_t^{\mathcal{P}})^{-1}\ddot{\boldsymbol{x}}_t + (\boldsymbol{K}_t^{\mathcal{P}})^{-1}\boldsymbol{K}_t^{\mathcal{V}}\dot{\boldsymbol{x}}_t + \boldsymbol{x}_t, \tag{4.0.2}$$

$$= \begin{bmatrix}\boldsymbol{I}, & (\boldsymbol{K}_t^{\mathcal{P}})^{-1}\boldsymbol{K}_t^{\mathcal{V}}, & (\boldsymbol{K}_t^{\mathcal{P}})^{-1}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t\\ \dot{\boldsymbol{x}}_t\\ \ddot{\boldsymbol{x}}_t\end{bmatrix}. \tag{4.0.3}$$

corresponding to a simple linear transformation of the observed data.

Fig. 3 illustrates the approach. It shows that learning the motion of the attractor can be used in free space to encode a movement, but that it can also be used to encode skills requiring specific force profiles to be applied. These two examples are illustrative of tasks in robotics traditionally decomposed into motion phases (in which robots freely move in the air), and contact phases (in which robots apply desired force patterns to objects/environment with hard contacts). However, the approach can be applied to a broader range of situations, such as applications requiring contacts with soft objects, tools or surfaces, and within environments that would be better characterized by local levels of pressures rather than binary contact states (e.g., surgical robots moving in-between organs, robots for ocean seabed/space exploration, humanoids walking on carpets and sitting on sofas, etc.).

5 Minimal intervention controller based on a linear quadratic regulator (LQR)

The approach presented in the previous section assumes that stiffness and damping parameters are provided to the system. We will here show that these parameters can also be estimated from the observation by following a reasoning similar to the development of linear quadratic regulators.

The approach exploits the regression approach presented in Section 3 to retrieve a reference trajectory in the form of a full distribution $\mathcal{N}(\hat{\boldsymbol{\xi}}_t^{\mathcal{O}}, \hat{\boldsymbol{\Sigma}}_t^{\mathcal{O}})$ varying at each time step t, given by Eq. (3.0.15), see [Calinon et al., 2014] for an experiment with a 7 DOFs manipulator.

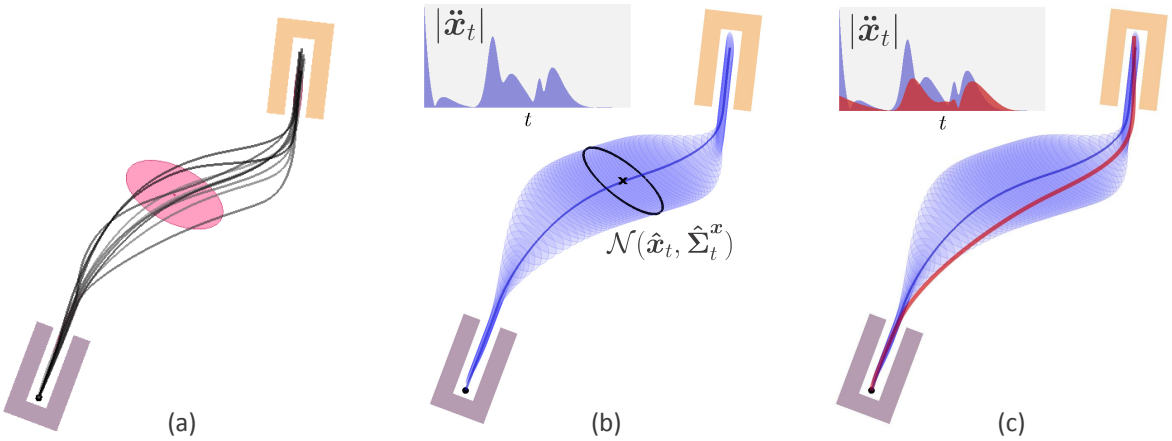


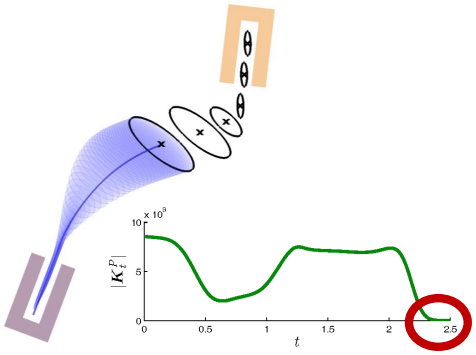
Figure 4: Minimal intervention controller based on a linear quadratic regulator (LQR). *(a)* Multiple demonstrations. *(b)* Reproduction of the reference path defined as the centers of the Gaussians retrieved by GMR, with the corresponding acceleration profile. *(c)* Reproduction with a linear quadratic regulator, reducing the accelerations and jerks while keeping the movement within a boundary defined by the demonstrations.

Finite horizon LQR

Cost function:

$$c^{(1)} = \sum_{t=1}^{T} (\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t)^\top \boldsymbol{Q}_t (\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) \;+\; \ddot{\boldsymbol{x}}_t^\top \boldsymbol{R}\, \ddot{\boldsymbol{x}}_t$$

- Riccati Ordinary Differential Equation
- Requires us to set desired final impedance



Infinite horizon LQR

Cost function: $\forall n \in \{1, \ldots, T\}$

$$c_n^{(2)} = \sum_{t=1}^{\infty} (\hat{\boldsymbol{x}}_n - \boldsymbol{x}_t)^\top \boldsymbol{Q}_n (\hat{\boldsymbol{x}}_n - \boldsymbol{x}_t) \;+\; \ddot{\boldsymbol{x}}_t^\top \boldsymbol{R}\, \ddot{\boldsymbol{x}}_t$$

- Algebraic Riccati Equation
- Estimation of covariance at current time step

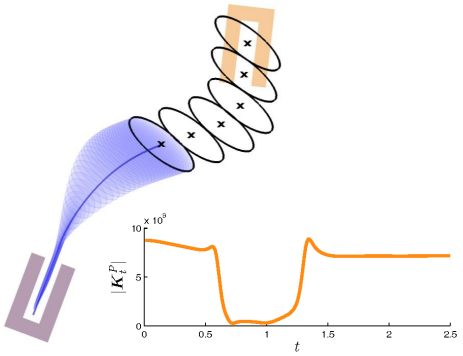


Figure 5: Difference between finite and infinite time horizon for minimal intervention control.

Similarly as the solution proposed by Medina *et al.* in the context of risk-sensitive control for haptic assistance [Medina et al., 2012], the predicted variability can be exploited to form a minimal intervention controller (in task space or in joint space). An acceleration command

$$\boldsymbol{u}_t = \hat{\boldsymbol{K}}_t^{\mathcal{P}}(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) - \hat{\boldsymbol{K}}_t^{\mathcal{V}}\dot{\boldsymbol{x}}_t \tag{5.0.1}$$

is used to control the robot (in task space or in joint space), with $\hat{\boldsymbol{x}}_t$ estimated by GMR in Eq. (3.0.15).

$\hat{\boldsymbol{K}}_t^{\mathcal{P}}$ and $\hat{\boldsymbol{K}}_t^{\mathcal{V}}$ are full stiffness and damping matrices estimated by a *linear quadratic regulator* (LQR) with time-varying weights. For a *finite horizon LQR*, this is achieved by minimizing the cost function

$$c^{(1)} = \sum_{t=1}^{T}(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t)^\top \boldsymbol{Q}_t(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) \ + \ \boldsymbol{u}_t^\top \boldsymbol{R}_t \, \boldsymbol{u}_t, \tag{5.0.2}$$

subject to the constraints of a double integrator system. This cost function aims at finding an optimal feedback controller minimizing simultaneously the tracking errors and the control commands in proportion defined by weighting matrices $\boldsymbol{Q}_t$ and $\boldsymbol{R}_t$.

The solution can be computed by backward integration of a *Riccati ordinary differential equation* with varying full weighting matrix $\boldsymbol{Q}_t = (\hat{\boldsymbol{\Sigma}}_t^{\boldsymbol{x}})^{-1}$ estimated with Eq. (3.0.15), and by setting $\boldsymbol{R}_t$ as a constant diagonal matrix. It provides a time-varying feedback control law in the form of Eq. (5.0.1) with full stiffness and damping matrices $\hat{\boldsymbol{K}}_t^{\mathcal{P}}$ and $\hat{\boldsymbol{K}}_t^{\mathcal{V}}$.

To solve the above minimization problem, a boundary condition needs to be set on the final feedback term, which can for example be set to zero. It is in this case assumed that the robot comes back to a compliant state after the task is fulfilled.

At iteration t, the backward recursion to minimize Eq. (5.0.2) requires the estimation of $\hat{\boldsymbol{\Sigma}}_n^{\mathcal{O}} = \hat{\boldsymbol{\Sigma}}_n^{\boldsymbol{x}}$ for $n \in \{t, t+1, \dots, T\}$. If the position and orientation of external objects are changing over time, the predicted trajectory and associated covariances need to be recomputed during the movement, providing a new recursion path for the Riccati equation.

In some situations, it might be computationally expensive to recompute at each iteration t a prediction on the remaining movement. An approximation that we proposed in [Calinon et al., 2014] can in this case be locally computed by considering an *infinite horizon LQR* formulation to estimate a feedback term at iteration t by considering only the current estimate $\hat{\boldsymbol{\Sigma}}_t^{\boldsymbol{x}}$. This corresponds to the estimation of a feedback controller that does not know in advance whether the precision at which it should track a target will vary. The corresponding cost function at iteration t corresponds to

$$c_t^{(2)} = \sum_{n=t}^{\infty}(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_n)^\top \boldsymbol{Q}_t(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_n) \ + \ \boldsymbol{u}_n^\top \boldsymbol{R}_t \, \boldsymbol{u}_n, \quad \forall t \in \{1, \dots, T\} \tag{5.0.3}$$

which can be solved iteratively through the *algebraic Riccati equation*, providing an optimal feedback controller in the form of Eq. (5.0.1) with full stiffness and damping matrices $\hat{\boldsymbol{K}}_t^{\mathcal{P}}$ and $\hat{\boldsymbol{K}}_t^{\mathcal{V}}$.

In this setting, and for minimization problems defined in task space, $\boldsymbol{R}_t$ can alternatively be set in function of the endpoint stiffness ellipsoid evaluated through the Jacobian at the current pose. For problems defined i joint space, $\boldsymbol{R}_t$ can optionally be set to apply local minimal intervention selectively to the joints (e.g., to cope with temporary damaged, unpowered or weak motors, or to preserve some degrees of freedom for additional task constraints).

Figures 4 and 5 illustrate the finite and infinite horizons approaches. These procedures can be considered as a form of inverse optimal control in the sense that the parameters of an objective function are learned from demonstrations, and that a controller for the robot is then found to satisfy the learned cost function.

In [Calinon et al., 2010b], a feedback controller was heuristically estimated by computing a stiffness matrix at each iteration t as proportional to the estimated precision matrix $(\hat{\boldsymbol{\Sigma}}_t^{\boldsymbol{x}})^{-1}$ of the current point to be tracked. The LQR approaches minimizing Equations (5.0.2) and (5.0.3) result in a controller sharing similar characteristics, but they provide a systematic way of adapting the impedance parameters, see also [Calinon et al., 2014].

5.1 Riccati ordinary differential equation (ODE)

In the next two sections, we will omit the index t when it is not relevant to the comprehension of the approach.

To emulate a virtual spring-damper system with an equilibrium point at the origin and a control variable $\boldsymbol{u}$ used as an acceleration command, we first define a double integrator system

$$\overbrace{\begin{bmatrix}\dot{\boldsymbol{x}}\\ \ddot{\boldsymbol{x}}\end{bmatrix}}^{\dot{\boldsymbol{\xi}}} = \overbrace{\begin{bmatrix}\boldsymbol{0} & \boldsymbol{I}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}}^{\boldsymbol{A}}\overbrace{\begin{bmatrix}\boldsymbol{x}\\ \dot{\boldsymbol{x}}\end{bmatrix}}^{\boldsymbol{\xi}} + \overbrace{\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{I}\end{bmatrix}}^{\boldsymbol{B}}\boldsymbol{u}. \tag{5.1.1}$$

For a fixed target at the origin, the total cost in Eq. (5.0.2) can be rewritten as

$$C = \int_0^T c\, dt, \quad \text{with} \quad c = \frac{1}{2}\boldsymbol{\xi}^\top \boldsymbol{Q}\boldsymbol{\xi} + \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u}. \tag{5.1.2}$$

Analogously to the optimization of a function subject to constraints, we construct the Hamiltonian H to find the optimal control, namely

$$H = c + \boldsymbol{\lambda}^\top(\boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u}), \tag{5.1.3}$$

with $\boldsymbol{\lambda}$ a vector of Lagrange multipliers (costate variables). $\boldsymbol{A}$ and $\boldsymbol{B}$ describe a linerar system such as in Eq. (5.2.8).

A set of necessary conditions for the solution to be optimal can be derived by the *Pontryagin maximum principle*, namely

$$\frac{\partial H}{\partial \boldsymbol{u}} = 0 \quad \Leftrightarrow \quad -\boldsymbol{\lambda}^\top \boldsymbol{B} = \boldsymbol{u}^\top \boldsymbol{R} \quad \Leftrightarrow \quad \boldsymbol{u} = -\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{\lambda}, \tag{5.1.4}$$

$$\dot{\boldsymbol{\lambda}}^\top = -\frac{\partial H}{\partial \boldsymbol{\xi}} = -\boldsymbol{\lambda}^\top \boldsymbol{A} - \boldsymbol{\xi}^\top \boldsymbol{Q} \quad \Leftrightarrow \quad \dot{\boldsymbol{\lambda}} = -\boldsymbol{A}^\top \boldsymbol{\lambda} - \boldsymbol{Q}^\top \boldsymbol{\xi}, \tag{5.1.5}$$

$$\dot{\boldsymbol{\xi}} = \frac{\partial H}{\partial \boldsymbol{\lambda}^\top} = \boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u}. \tag{5.1.6}$$

The form of the optimal solution is given by the solution of a differential equation with boundary conditions. In this case, one must solve a two point boundary value problem using the initial condition $\boldsymbol{\xi}_0$ and the final condition $\boldsymbol{\lambda}_T$. We can attempt to find a solution to such problem by setting $\boldsymbol{\lambda} = \boldsymbol{P}\boldsymbol{\xi}$ (a hint is that we try to get a linear feedback between $\boldsymbol{u}$ and $\boldsymbol{\xi}$), which is used in (5.1.4) to obtain

$$\boldsymbol{u} = -\overbrace{\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P}}^{\boldsymbol{K}}\, \boldsymbol{\xi}. \tag{5.1.7}$$

Deriving $\boldsymbol{\lambda} = \boldsymbol{P}\boldsymbol{\xi}$ results in $\dot{\boldsymbol{\lambda}} = \dot{\boldsymbol{P}}\boldsymbol{\xi} + \boldsymbol{P}\dot{\boldsymbol{\xi}}$, which is used with (5.1.5) and (5.1.6) to obtain

$$-\boldsymbol{A}^\top \boldsymbol{\lambda} - \boldsymbol{Q}^\top \boldsymbol{\xi} = \dot{\boldsymbol{P}}\boldsymbol{\xi} + \boldsymbol{P}(\boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u}), \tag{5.1.8}$$

$$\Leftrightarrow \quad -\boldsymbol{A}^\top \boldsymbol{P}\,\boldsymbol{\xi} - \boldsymbol{Q}^\top \boldsymbol{\xi} = \dot{\boldsymbol{P}}\,\boldsymbol{\xi} + \boldsymbol{P}\boldsymbol{A}\,\boldsymbol{\xi} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P}\,\boldsymbol{\xi}. \tag{5.1.9}$$

The equation is satisfied if we can find $\boldsymbol{P}$ such that

$$-\dot{\boldsymbol{P}} = \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P} + \boldsymbol{Q}. \tag{5.1.10}$$

The above matrix equation is the *Riccati ordinary differential equation*. By setting $\dot{\boldsymbol{P}}_t = \frac{1}{\Delta t}(\boldsymbol{P}_t - \boldsymbol{P}_{t-1})$, we get the backward recursion

$$\boldsymbol{P}_{t-1} = \boldsymbol{P}_t + \Delta t \left(\boldsymbol{A}^\top \boldsymbol{P}_t + \boldsymbol{P}_t \boldsymbol{A} - \boldsymbol{P}_t \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P}_t + \boldsymbol{Q}_t\right). \tag{5.1.11}$$

We can notice that the above equation can be solved by recursion without knowing $\boldsymbol{\xi}_t$ and $\boldsymbol{u}_t$. At each time step t, the estimation of $\boldsymbol{P}_t$ then provides the feedback control law in Eq. (5.1.7). The system $\dot{\boldsymbol{\xi}}_t = \boldsymbol{A}\boldsymbol{\xi}_t + \boldsymbol{B}\boldsymbol{u}_t$ then becomes

$$\dot{\boldsymbol{\xi}}_t = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}_t)\,\boldsymbol{\xi}_t \tag{5.1.12}$$

$$= (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P}_t)\,\boldsymbol{\xi}_t. \tag{5.1.13}$$

corresponding to

$$\ddot{\boldsymbol{x}}_t = -\hat{\boldsymbol{K}}_t^{\mathcal{P}}\boldsymbol{x}_t - \hat{\boldsymbol{K}}_t^{\mathcal{V}}\dot{\boldsymbol{x}}_t, \tag{5.1.14}$$

with full stiffness and damping matrices $\hat{\boldsymbol{K}}_t^{\mathcal{P}}$ and $\hat{\boldsymbol{K}}_t^{\mathcal{V}}$.

5.1.1 Note on stability at the equilibrium point

To verify the asymptotic stability of the system at its equilibrium point, we can define the Lyapunov function

$$v = \boldsymbol{\xi}^\top \boldsymbol{P}\,\boldsymbol{\xi}. \tag{5.1.15}$$

v is positive definite. By using Eq. (5.1.13), it has the following time derivative along the solutions of the LQR problem

$$\dot{v} = \dot{\boldsymbol{\xi}}^\top \boldsymbol{P}\,\boldsymbol{\xi} + \boldsymbol{\xi}^\top \dot{\boldsymbol{P}}\,\boldsymbol{\xi} + \boldsymbol{\xi}^\top \boldsymbol{P}\,\dot{\boldsymbol{\xi}} \tag{5.1.16}$$

$$= \boldsymbol{\xi}^\top \left[\boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{K}^\top \boldsymbol{B}^\top \boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{K} + \dot{\boldsymbol{P}}\right]\boldsymbol{\xi} \tag{5.1.17}$$

$$= \boldsymbol{\xi}^\top \left[\boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}^\top \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P} + \dot{\boldsymbol{P}}\right]\boldsymbol{\xi}. \tag{5.1.18}$$

Substituting $\dot{\boldsymbol{P}}$ from Eq. (5.1.10), we have

$$\dot{v} = \boldsymbol{\xi}^\top \left[-\boldsymbol{P}^\top \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P} - \boldsymbol{Q}\right] \boldsymbol{\xi}, \tag{5.1.19}$$

Since $\boldsymbol{Q}$ and $\boldsymbol{R}$ are positive semi-definite, $\dot{v}$ is negative for all $\boldsymbol{\xi} \neq \boldsymbol{0}$, and zero if $\boldsymbol{\xi} = \boldsymbol{0}$. By the Lyapunov theorem, the system is thus asymptotically stable around its equilibrium point.

Note that the above development provides relevant information about the optimization results, by knowing that the controller in (5.1.14) will evaluate appropriate ratios between stiffness and damping components and that the full matrices will be of appropriate forms. However, it only constitutes a partial proof, because a tracking system with a moving attractor point is considered in the proposed approach, which means that the attractor path should also be taken into account, as we will describe in next section.

5.1.2 Riccati ODE for a tracking problem

For a general tracking problem, and by defining a weighting term $\boldsymbol{Q}$ on the state variable $\boldsymbol{\xi}$, the total cost in Eq. (5.0.2) can be rewritten as

$$c^{(1)} = \int_0^T c\, dt, \quad \text{with} \quad c = \frac{1}{2}(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi})^\top \boldsymbol{Q}(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}) + \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u}. \tag{5.1.20}$$

Similarly to (5.1.4)-(5.1.6), the equations for the optimal trajectory can be obtained from the Hamiltonian $H = c + \boldsymbol{\lambda}^\top(\boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u})$ as

$$\frac{\partial H}{\partial \boldsymbol{u}} = 0 \quad \Leftrightarrow \quad -\boldsymbol{\lambda}^\top \boldsymbol{B} = \boldsymbol{u}^\top \boldsymbol{R} \quad \Leftrightarrow \quad \boldsymbol{u} = -\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{\lambda}, \tag{5.1.21}$$

$$\dot{\boldsymbol{\lambda}}^\top = -\frac{\partial H}{\partial \boldsymbol{\xi}} = -\boldsymbol{\lambda}^\top \boldsymbol{A} + (\hat{\boldsymbol{\xi}} - \boldsymbol{\xi})^\top \boldsymbol{Q} \quad \Leftrightarrow \quad \dot{\boldsymbol{\lambda}} = -\boldsymbol{A}^\top \boldsymbol{\lambda} + \boldsymbol{Q}^\top(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}), \tag{5.1.22}$$

$$\dot{\boldsymbol{\xi}} = \frac{\partial H}{\partial \boldsymbol{\lambda}^\top} = \boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u}. \tag{5.1.23}$$

If we make the hypothesis that the controller is obtained as a feedback term on the state variables and a feedforward term, we get

$$\boldsymbol{\lambda} = -\boldsymbol{P}(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}) - \boldsymbol{d}. \tag{5.1.24}$$

Taking the time derivatives in both sides of (5.1.24), we obtain

$$\dot{\boldsymbol{\lambda}} = -\dot{\boldsymbol{P}}(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}) - \boldsymbol{P}\dot{\hat{\boldsymbol{\xi}}} + \boldsymbol{P}\dot{\boldsymbol{\xi}} - \dot{\boldsymbol{d}}. \tag{5.1.25}$$

Substituting (5.1.21)-(5.1.23) into (5.1.25), and by denoting $\boldsymbol{e} = \hat{\boldsymbol{\xi}} - \boldsymbol{\xi}$, we get

$$-\boldsymbol{A}^\top\big(-\boldsymbol{P}\boldsymbol{e} - \boldsymbol{d}\big) + \boldsymbol{Q}^\top \boldsymbol{e} = -\dot{\boldsymbol{P}}\boldsymbol{e} + \boldsymbol{P}\Big(\boldsymbol{A}\boldsymbol{\xi} - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top\big(-\boldsymbol{P}\boldsymbol{e} - \boldsymbol{d}\big)\Big) - \boldsymbol{P}\dot{\hat{\boldsymbol{\xi}}} - \dot{\boldsymbol{d}}$$

$$\Leftrightarrow \quad \boldsymbol{A}^\top\boldsymbol{P}\boldsymbol{e} + \boldsymbol{Q}^\top\boldsymbol{e} + \boldsymbol{A}^\top\boldsymbol{d} = -\dot{\boldsymbol{P}}\boldsymbol{e} - \boldsymbol{P}\boldsymbol{A}\boldsymbol{e} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top\boldsymbol{P}\boldsymbol{e} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}\boldsymbol{d} - \boldsymbol{P}\dot{\hat{\boldsymbol{\xi}}} - \dot{\boldsymbol{d}} + \boldsymbol{P}\boldsymbol{A}\hat{\boldsymbol{\xi}}. \tag{5.1.26}$$

We have in the above a relation of the form $\alpha\boldsymbol{e} + \beta = 0$ that we would like to be satisfied $\forall \boldsymbol{e}$, which requires $\alpha = 0$ and $\beta = 0$, namely

$$-\dot{\boldsymbol{P}} = \boldsymbol{A}^\top\boldsymbol{P} + \boldsymbol{Q} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top\boldsymbol{P}, \tag{5.1.27}$$

$$-\dot{\boldsymbol{d}} = \boldsymbol{A}^\top\boldsymbol{d} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}\boldsymbol{d} + \boldsymbol{P}\dot{\hat{\boldsymbol{\xi}}} - \boldsymbol{P}\boldsymbol{A}\hat{\boldsymbol{\xi}}. \tag{5.1.28}$$

The first equation is the classical Riccati equation for LQR systems as in (5.1.10), while the second is a linear differential equation for the feedforward term, depending on the solution of the Riccati equation. The two equations can be computed backward by following (5.1.11) and

$$\boldsymbol{d}_{t-1} = \boldsymbol{d}_t + \Delta t\Big(\boldsymbol{A}^\top\boldsymbol{d}_t - \boldsymbol{P}_t\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}\boldsymbol{d}_t + \boldsymbol{P}_t\dot{\hat{\boldsymbol{\xi}}}_t - \boldsymbol{P}_t\boldsymbol{A}\hat{\boldsymbol{\xi}}_t\Big), \tag{5.1.29}$$

with $\boldsymbol{d}_T = \boldsymbol{0}$.

The resulting controller is

$$\ddot{\boldsymbol{\xi}}_t = -\boldsymbol{R}^{-1}\boldsymbol{B}^\top\Big(\boldsymbol{P}_t(\hat{\boldsymbol{\xi}}_t - \boldsymbol{\xi}_t) + \boldsymbol{d}_t\Big), \tag{5.1.30}$$

which corresponds to (with appropriate weighting terms in the cost function)

$$\ddot{\boldsymbol{x}}_t = \hat{\boldsymbol{K}}_t^{\mathcal{P}}(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) - \hat{\boldsymbol{K}}_t^{\mathcal{V}}\dot{\boldsymbol{x}}_t + \hat{\boldsymbol{F}}_t. \tag{5.1.31}$$

In the above, the feedback terms $\hat{\boldsymbol{K}}_t^{\mathcal{P}}$, $\hat{\boldsymbol{K}}_t^{\mathcal{V}}$ are thus estimated by backward recursion of the Riccati equation (5.1.27), and the feedforward term $\hat{\boldsymbol{F}}_t$ is estimated as the linear first order equation (5.1.28).

In practice, for motion with low dynamics, the feedforward term in (5.1.31) is negligible compared to the feedback parts, and can in some applications be omitted.

5.2 Algebraic Riccati equation (ARE)

For the infinite horizon setting, the cost function in Eq. (5.0.3) can be minimized with the algebraic Riccati equation

$$\boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0} \tag{5.2.1}$$

$$\Leftrightarrow \begin{bmatrix}\boldsymbol{P} & -\boldsymbol{I}\end{bmatrix} \boldsymbol{H} \begin{bmatrix}\boldsymbol{I} \\ \boldsymbol{P}\end{bmatrix} = \boldsymbol{0}, \tag{5.2.2}$$

with the Hamiltonian matrix

$$\boldsymbol{H} = \begin{bmatrix}\boldsymbol{A} & -\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \\ -\boldsymbol{Q} & -\boldsymbol{A}^\top\end{bmatrix}. \tag{5.2.3}$$

Under suitable hypotheses about symmetry, stabilizability and detectability, Eq. (5.2.1) has a unique positive semidefinite solution $\boldsymbol{P}$, which can be obtained by several methods, see [Laub, 1979] for details.

The key is to convert the problem to a stable invariant subspace problem of the Hamiltonian matrix, i.e., finding the invariant subspace corresponding to eigenvalues of $\boldsymbol{H}$ with negative real parts. This subspace can be found in several ways. The eigendecomposition approach (with ordered eigencomponents) decomposes $\boldsymbol{H}$ as

$$\boldsymbol{H} = \boldsymbol{V}\begin{bmatrix}\boldsymbol{\lambda}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\lambda}_2\end{bmatrix}\boldsymbol{V}^\top, \quad \text{with} \quad \boldsymbol{V} = \begin{bmatrix}\boldsymbol{V}_1 & \boldsymbol{V}_{12} \\ \boldsymbol{V}_{21} & \boldsymbol{V}_2\end{bmatrix}, \tag{5.2.4}$$

where $\begin{bmatrix}\boldsymbol{V}_1 \\ \boldsymbol{V}_{21}\end{bmatrix}$ forms the stable eigenspace of $\boldsymbol{H}$. We have that $\boldsymbol{H} \in \mathbb{R}^{4D\times 4D}$ for a double integrator as in Eq. (5.2.8), which precisely has $2D$ eigenvalues with negative real parts. Together with (5.2.2), it provides the ARE solution

$$\boldsymbol{P} = \boldsymbol{V}_{21}\boldsymbol{V}_1^{-1}. \tag{5.2.5}$$

The eigenvector approach can be numerically unstable when the Hamiltonian matrix is close to defective, as can occur some cases in which H is close to a matrix whose Jordan form has ones above its main diagonal. In this case, the matrix $\boldsymbol{V}_1$ will be ill-conditioned. However, the ill-conditioning of $\boldsymbol{V}_1$ is independent of the conditioning of the Riccati equation [Laub, 1979, Paige and Van Loan, 1981].

To circumvent the ill-conditioning problems associated with a defective Hamiltonian matrix, the same stable eigenspace of Hamiltonian can be spanned by Schur vectors [Laub, 1979]. In this case, an orthogonal transformation $\boldsymbol{U}$ is found to reduce $\boldsymbol{H}$ to real Schur form

$$\boldsymbol{H} = \boldsymbol{U}\begin{bmatrix}\boldsymbol{T}_1 & \boldsymbol{T}_{12} \\ \boldsymbol{0} & \boldsymbol{T}_2\end{bmatrix}\boldsymbol{U}^\top, \quad \text{with} \quad \boldsymbol{U} = \begin{bmatrix}\boldsymbol{U}_1 & \boldsymbol{U}_{12} \\ \boldsymbol{U}_{21} & \boldsymbol{U}_2\end{bmatrix}, \tag{5.2.6}$$

ordered such that the eigenvalues of $\boldsymbol{T}_1$ are stable and those of $\boldsymbol{T}_2$ are unstable. This can be achieved with additional orthogonal transformations reordering the Schur form such that the negative eigenvalues precede the non-negative eigenvalues on the diagonal of the upper triangular matrices.

In the above, $\begin{bmatrix}\boldsymbol{U}_1 \\ \boldsymbol{U}_{21}\end{bmatrix}$ spans for the same stable eigenspace of $\boldsymbol{H}$ as in (5.2.4), providing with Eq. (5.2.2) the ARE solution

$$\boldsymbol{P} = \boldsymbol{U}_{21}\boldsymbol{U}_1^{-1}. \tag{5.2.7}$$

Both (5.2.5) and (5.2.7) solve (5.2.1) with $\boldsymbol{P} = \boldsymbol{P}^\top$ positive definite.

As an example of application, in the case of diagonal stiffness and damping matrices, it is easy to show that the stiffness-damping ratio estimated by ARE corresponds to an ideal damping ratio, see for example the supplementary notes from [Astrom and Murray, 2008]. By defining a double integrator system as

$$\begin{bmatrix}\dot{e} \\ \ddot{e}\end{bmatrix} = \overbrace{\begin{bmatrix}0 & 1 \\ 0 & 0\end{bmatrix}}^{\boldsymbol{A}}\begin{bmatrix}e \\ \dot{e}\end{bmatrix} + \overbrace{\begin{bmatrix}0 \\ 1\end{bmatrix}}^{\boldsymbol{B}} u, \tag{5.2.8}$$

and by setting the weighting terms

$$\boldsymbol{Q} = \begin{bmatrix}c_1 & 0 \\ 0 & 0\end{bmatrix}, \quad R = c_2, \tag{5.2.9}$$

with c_1 and c_2 constant scalars, the optimal feedback law found by the system is described by $u = -\kappa^{\mathcal{P}} e - \kappa^{\mathcal{V}} \dot{e}$, with $\kappa^{\mathcal{V}} = \sqrt{2\kappa^{\mathcal{P}}}$, which corresponds to the ideal underdamped damping ratio $\zeta = \frac{\kappa^{\mathcal{V}}}{2\sqrt{\kappa^{\mathcal{P}}}} = \frac{1}{\sqrt{2}} \approx 0.7071$ in control design, optimizing the reaction speed while still allowing the system to asymptotically come back to its rest position.

Similarly, for multivariate problems, the stiffness and damping matrices estimated by the system automatically satisfy the relation

$$\hat{\boldsymbol{K}}_t^{\mathcal{V}} = \boldsymbol{V}(2\boldsymbol{D})^{1/2}\boldsymbol{V}^\top, \tag{5.2.10}$$

with $\hat{\boldsymbol{K}}_t^{\mathcal{P}} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^\top$ the eigendecomposition of $\hat{\boldsymbol{K}}_t^{\mathcal{P}}$, with unit eigenvectors represented as a matrix $\boldsymbol{V}$, and the squared eigenvalues represented in a diagonal matrix $\boldsymbol{D}$.

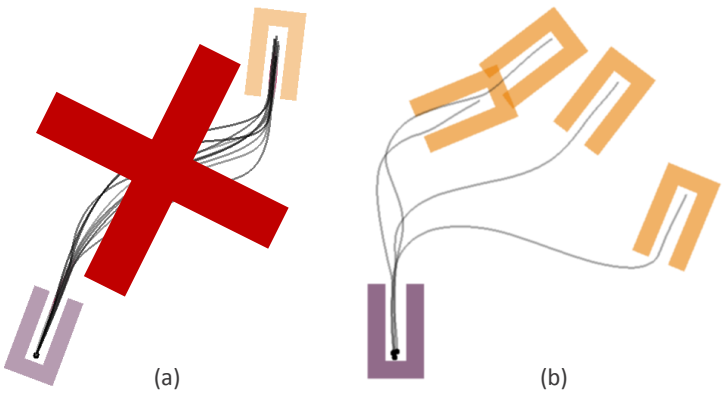


Figure 6: Dataset of movement collected in: *(a)* the same situation; and *(b)* different situations. In practice, it is easier to collect data and gather different databases if we can consider recordings collected in different situations such as in *(b)*.

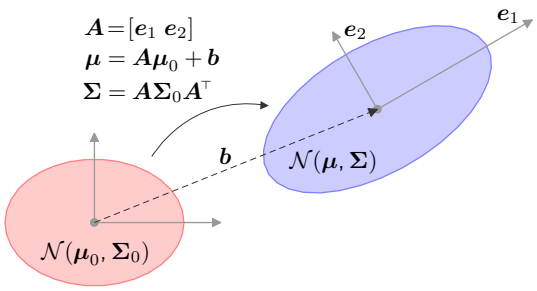


Figure 7: Representation of a frame of reference as a coordinate system described by position $\boldsymbol{b}$ and transformation matrix $\boldsymbol{A}$, with associated Gaussian projection properties.

6 Task-parameterized GMM (TP-GMM)

In human-robot teaching interactions, it is difficult to collect a set of demonstrations that are performed in the exact same conditions and situations. Fig. 6 illustrates this issue for the case of different position and orientation of a final landmark (e.g., prehension of an object).

The reproduction phase also faces a similar issue. Namely, after having observed a set of demonstrations in some situations, we would like to generalize the skill to new situations (e.g., characterized by new position and orientations of object).

We recall that the term *model parameters* will be used to refer to the learned parameters of a model describing the movement or skill behaviors. The external parameters describing the current situation (such as position of objects or users, changes in the environment and changes of configurations of another robot parts) will be described as *task parameters*. Task parameters are used as inputs to transform the learned *model parameters* in accordance to the situation.

Various approaches have been proposed to encode such task-parameterized skills (sometimes called parametric skills). We presented in [Calinon et al., 2013] a classification and comparisons of these approaches. Most of them rely on interpolation techniques using datasets concatenating the parameters of the task with the parameters describing the motion (sometimes in the form of a compact model).

We showed in [Calinon et al., 2013] that the centers and covariances of a GMM could be modulated by the location and orientation of multiple objects, virtual landmarks or coordinate systems, providing an efficient way of generalizing movements, for the special case in which the task parameters can be represented as frames of reference. It shares with [Wilson and Bobick, 1999] the strategy of adapting the Gaussian centers of a GMM to new situations, but provides a way to modulate the full covariance matrices.

The proposed *task-parameterized GMM* probabilistically encodes the relevance of candidate frames, which can change throughout the task. It thus allows autonomous transitions between different coordinate systems that are potentially relevant for the task (e.g., adaptation to multiple viapoints in the middle of the movement, with local position, orientation and shape modulation).

The task parameters are represented as P coordinate systems, defined at time step t by $\{\boldsymbol{b}_{t,j},\boldsymbol{A}_{t,j}\}_{j=1}^{P}$, representing respectively the origin of the observer and a set of basis vectors $\{\boldsymbol{e}_1,\boldsymbol{e}_2,...\}$ forming a transformation matrix $\boldsymbol{A}=[\boldsymbol{e}_1\boldsymbol{e}_2\cdots]$, see Fig. 7.

A movement $\boldsymbol{\xi}\in\mathbb{R}^{D\times T}$ is observed from these different viewpoints, forming a third order tensor dataset $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{D\times T\times P}$, composed of P trajectory samples $\boldsymbol{X}^{(j)}\in\mathbb{R}^{D\times T}$ observed in P candidate frames, corresponding

to matrices composed of D-dimensional observations at T time steps.

The parameters of the proposed *task-parameterized GMM* (TP-GMM) with K components are defined by $\{\pi_i, \{\boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\}_{j=1}^P\}_{i=1}^K$ (π_i are the mixing coefficients, $\boldsymbol{\mu}_i^{(j)}$ and $\boldsymbol{\Sigma}_i^{(j)}$ are the center and covariance matrix of the i-th Gaussian component in frame j).

Learning of the parameters is achieved with the constrained problem of maximizing the log-likelihood under the constraints that the data in the different frames are generated from the same source, resulting in an EM process to iteratively update the model parameters until convergence.

E-step:

$$h_{t,i} = \frac{\pi_i \prod_{j=1}^{P} \mathcal{N}\big(\boldsymbol{X}_t^{(j)} | \, \boldsymbol{\mu}_i^{(j)}, \boldsymbol{\Sigma}_i^{(j)}\big)}{\sum_{k=1}^{K} \pi_k \prod_{j=1}^{P} \mathcal{N}\big(\boldsymbol{X}_t^{(j)} | \, \boldsymbol{\mu}_k^{(j)}, \boldsymbol{\Sigma}_k^{(j)}\big)}. \tag{6.0.1}$$

M-step:

$$\pi_i = \frac{\sum_{t=1}^{T} h_{t,i}}{T}, \tag{6.0.2}$$

$$\boldsymbol{\mu}_i^{(j)} = \frac{\sum_{t=1}^{T} h_{t,i} \, \boldsymbol{X}_t^{(j)}}{\sum_{t=1}^{T} h_{t,i}}, \tag{6.0.3}$$

$$\boldsymbol{\Sigma}_i^{(j)} = \frac{\sum_{t=1}^{T} h_{t,i} \, (\boldsymbol{X}_t^{(j)} - \boldsymbol{\mu}_i^{(j)})(\boldsymbol{X}_t^{(j)} - \boldsymbol{\mu}_i^{(j)})^\top}{\sum_{t=1}^{T} h_{t,i}}. \tag{6.0.4}$$

The model parameters are initialized with a *k-means* procedure. Model selection is compatible with the techniques employed in standard GMM (*Bayesian information criterion* [Schwarz, 1978], *Dirichlet process* [Rasmussen, 2000], *iterative pairwise replacement* [Scott and Szewczyk, 2001], etc.). In a standard GMM as described in Section 2, the role of EM is to estimate constant Gaussian parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$. Here, EM is used to estimate task-parameterized model parameters $\boldsymbol{\mu}_i^{(j)}$ and $\boldsymbol{\Sigma}_i^{(j)}$ by incrementally modeling the local importance of the candidate frames. For a movement in Cartesian space with 10 demonstrations and 3 candidate frames, the overall learning process typically takes 1 to 4 *sec*. The reproduction is much faster and can be computed online (below 1 *msec*).

The version of TP-GMM presented in [Calinon et al., 2014] is equivalent to the version presented in [Calinon et al., 2013], but is computationally more efficient. The E-step formulated in [Calinon et al., 2014] and above involves a product of probabilities (multiplication of scalars), while the E-step in [Calinon et al., 2013] first computes the intersection of Gaussians (products of Gaussians) before evaluating the likelihoods. With the above formulation, there is no need to explicitly provide the parameters $\boldsymbol{A}_{t,j}$ and $\boldsymbol{b}_{t,j}$ in the learning phase (this information is already contained in the third order tensor dataset $\boldsymbol{\mathcal{X}}$, with the demonstrations observed from different perspectives).

The learned model can further be used to reproduce movements in other situations (for new positions and orientations of candidate frames). The model first retrieves at each time step t a GMM by computing a product of linearly transformed Gaussians

$$\mathcal{N}(\boldsymbol{\mu}_{t,i}, \boldsymbol{\Sigma}_{t,i}) \propto \prod_{j=1}^{P} \mathcal{N}\Big(\boldsymbol{A}_{t,j}\boldsymbol{\mu}_i^{(j)} + \boldsymbol{b}_{t,j}, \; \boldsymbol{A}_{t,j}\boldsymbol{\Sigma}_i^{(j)}\boldsymbol{A}_{t,j}^\top\Big), \tag{6.0.5}$$

$$\Leftrightarrow \; \boldsymbol{\Sigma}_{t,i} = \Big(\sum_{j=1}^{P} (\boldsymbol{A}_{t,j}\boldsymbol{\Sigma}_i^{(j)}\boldsymbol{A}_{t,j}^\top)^{-1}\Big)^{-1}, \tag{6.0.6}$$

$$\boldsymbol{\mu}_{t,i} = \boldsymbol{\Sigma}_{t,i} \sum_{j=1}^{P} (\boldsymbol{A}_{t,j}\boldsymbol{\Sigma}_i^{(j)}\boldsymbol{A}_{t,j}^\top)^{-1}(\boldsymbol{A}_{t,j}\boldsymbol{\mu}_i^{(j)} + \boldsymbol{b}_{t,j}). \tag{6.0.7}$$

7 Example of application

Fig. 8 presents an illustration of the overall approach combining statistical mixture models, dynamical systems and optimal control to learn and reproduce movement skills represented as trajectories indexed by time.

The complete procedure includes a *demonstration* phase, a *learning* phase and a *reproduction* phase. In the demonstration phase, a set of movements is recorded as position and orientation of the robot end-effector (output) with associated time stamp (input). The multiple demonstrations can optionally be aligned in time with *dynamic time warping*.

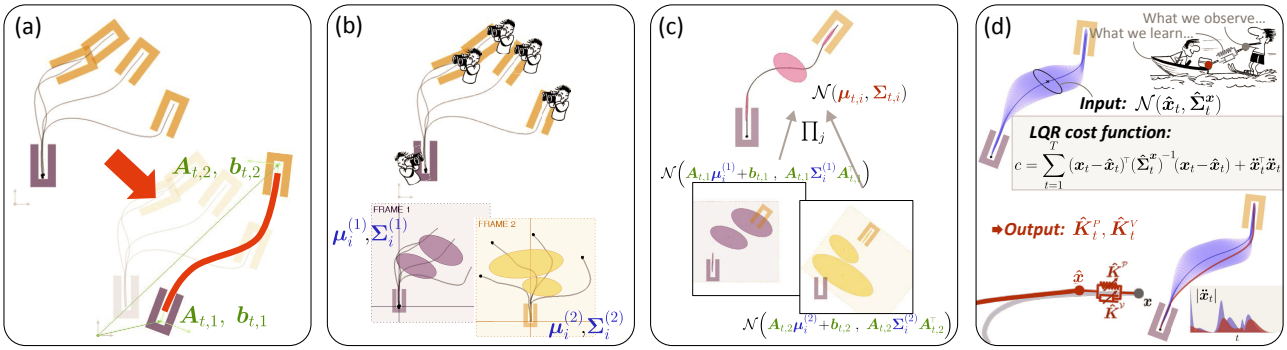


Figure 8: Illustration of the overall approach (see main text for details).

The position and orientation of a set of candidate frames (related to objects in the robot workspace) is also collected. The recorded movements are projected in these frames (observation of the same movement from multiple viewpoints). The input and output variables are concatenated for each frame, forming a 3rd order tensor dataset. In this example, time is used as input variable, but a decay term, the robot state or other external object position variables can similarly be employed [Calinon et al., 2012].

In the learning phase, a task-parameterized mixture model is fit to the tensor dataset by following an *expectation-maximization* (EM) procedure (Section 6). The training set can then be discarded.

In the reproduction phase, for a situation involving new position and orientation of objects, the learned model is first used to estimate a temporary Gaussian mixture model (GMM), that is automatically updated if there is a change in position/orientation of the objects. Depending on the application, this temporary GMM either needs to be updated at each time step (e.g., adapting movements to moving targets), or for each new reproduction attempt (planning approach).

Gaussian mixture regression (GMR) is then used to retrieve statistical information about the current reference to track, corresponding to the equilibrium point of a virtual spring-damper system (Section 3).

This information is finally used by a linear quadratic regulator to form a minimal intervention controller (Section 5).

8 Conclusion

An approach capable of adapting the centers and covariance matrices of a GMM to external task parameters represented as candidate frames of reference was presented. This *task-parameterized model* was applied in the context of *learning from demonstration* to encode and generalize a demonstrated task to new situations. It was shown that the approach could be combined with a virtual spring-damper system with variable impedance gains. For new position and orientation of candidate frames, the system generates a flow tube predicting the path of the virtual spring and its variations. It was then shown that the covariance information could be exploited in an optimal control strategy to locally reduce the control commands according to the precision required at each step of the task. Two different minimization strategies were considered to estimate varying full stiffness and damping matrices for the regulation of the movement, depending on the availability or non-availability of the predicted covariances over the whole movement.

References

[Agiomyrgiannakis and Stylianou, 2009] Agiomyrgiannakis, Y. and Stylianou, Y. (2009). Wrapped Gaussian mixture models for modeling and high-rate quantization of phase data of speech. *IEEE Trans. on Audio, Speech, and Language Processing*, 17(4):775–786.

[Astrom and Murray, 2008] Astrom, K. J. and Murray, R. M. (2008). *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, Princeton, NJ, USA.

[Baek et al., 2010] Baek, J., McLachlan, G. J., and Flack, L. K. (2010). Mixtures of factor analyzers with common factor loadings: Applications to the clustering and visualization of high-dimensional data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(7):1298–1309.

[Bouveyron and Brunet, 2014] Bouveyron, C. and Brunet, C. (2014). Model-based clustering of high-dimensional data: A review. *Computational Statistics and Data Analysis*, 71:52–78.

[Calinon, 2009] Calinon, S. (2009). *Robot Programming by Demonstration: A Probabilistic Approach*. EPFL/CRC Press. EPFL Press ISBN 978-2-940222-31-5, CRC Press ISBN 978-1-4398-0867-2.

[Calinon et al., 2013] Calinon, S., Alizadeh, T., and Caldwell, D. G. (2013). On improving the extrapolation capability of task-parameterized movement models. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 610–616, Tokyo, Japan.

[Calinon et al., 2014] Calinon, S., Bruno, D., and Caldwell, D. G. (2014). A task-parameterized probabilistic model with minimal intervention control. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 3339–3344, Hong Kong, China.

[Calinon et al., 2010a] Calinon, S., D'halluin, F., Sauser, E. L., Caldwell, D. G., and Billard, A. G. (2010a). Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression. *IEEE Robotics and Automation Magazine*, 17(2):44–54.

[Calinon et al., 2007] Calinon, S., Guenter, F., and Billard, A. G. (2007). On learning, representing and generalizing a task in a humanoid robot. *IEEE Trans. on Systems, Man and Cybernetics, Part B*, 37(2):286–298.

[Calinon et al., 2012] Calinon, S., Li, Z., Alizadeh, T., Tsagarakis, N. G., and Caldwell, D. G. (2012). Statistical dynamical systems for skills acquisition in humanoids. In *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, pages 323–329, Osaka, Japan.

[Calinon et al., 2010b] Calinon, S., Sardellitti, I., and Caldwell, D. G. (2010b). Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 249–254, Taipei, Taiwan.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38.

[Ghahramani and Jordan, 1994] Ghahramani, Z. and Jordan, M. I. (1994). Supervised learning from incomplete data via an EM approach. In Cowan, J. D., Tesauro, G., and Alspector, J., editors, *Advances in Neural Information Processing Systems*, volume 6, pages 120–127, San Francisco, CA, USA. Morgan Kaufmann Publishers, Inc.

[Grimes et al., 2006] Grimes, D. B., Chalodhorn, R., and Rao, R. P. N. (2006). Dynamic imitation in a humanoid robot through nonparametric probabilistic inference. In *Proc. Robotics: Science and Systems (RSS)*, pages 1–8.

[Hogan and Sternad, 2013] Hogan, N. and Sternad, D. (2013). Dynamic primitives in the control of locomotion. *Frontiers in Computational Neuroscience*, 7(71).

[Hsu and Kakade, 2013] Hsu, D. and Kakade, S. M. (2013). Learning mixtures of spherical Gaussians: Moment methods and spectral decompositions. In *Conf. on Innovations in Theoretical Computer Science*, pages 11–20.

[Ijspeert et al., 2013] Ijspeert, A., Nakanishi, J., Pastor, P., Hoffmann, H., and Schaal, S. (2013). Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25(2):328–373.

[Khansari-Zadeh and Billard, 2011] Khansari-Zadeh, S. M. and Billard, A. (2011). Learning stable non-linear dynamical systems with Gaussian mixture models. *IEEE Trans. on Robotics*, 27(5):943–957.

[Kulis and Jordan, 2012] Kulis, B. and Jordan, M. I. (2012). Revisiting k-means: New algorithms via Bayesian non-parametrics. In *Proc. Intl Conf. on Machine Learning (ICML)*.

[Kumar et al., 2009] Kumar, N. S. L. P., Satoor, S., and Buck, I. (2009). Fast parallel expectation maximization for Gaussian mixture models on GPUs using CUDA. In *IEEE Intl Conf. on High Performance Computing and Communications*, pages 103–109, Washington, DC, USA.

[Laub, 1979] Laub, A. J. (1979). A Schur method for solving algebraic Riccati equations. *IEEE Trans. on Automatic Control*, 24(6):913–921.

[MacQueen, 1967] MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symp. on mathematical statistics and probability*, pages 281–297.

[Mardia and Jupp, 1999] Mardia, K. V. and Jupp, P. E. (1999). *Directional Statistics*. Wiley.

[McLachlan and Peel, 2000] McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models*. Wiley-Interscience, New York, USA.

[McLachlan et al., 2003] McLachlan, G. J., Peel, D., and Bean, R. W. (2003). Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics and Data Analysis*, 41(3-4):379–388.

[McNicholas and Murphy, 2008] McNicholas, P. D. and Murphy, T. B. (2008). Parsimonious Gaussian mixture models. *Statistics and Computing*, 18(3):285–296.

[Medina et al., 2012] Medina, J. R., Lee, D., and Hirche, S. (2012). Risk-sensitive optimal feedback control for haptic assistance. In *IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 1025–1031.

[Mongillo and Deneve, 2008] Mongillo, G. and Deneve, S. (2008). Online learning with hidden Markov models. *Neural Computation*, 20(7):1706–1716.

[Neal and Hinton, 1999] Neal, R. M. and Hinton, G. E. (1999). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. MIT Press, Cambridge, MA, USA.

[Ng et al., 2001] Ng, A., Jordan, M., and Weiss, Y. (2001). On Spectral Clustering: Analysis and an algorithm. In Dietterich, T., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press.

[Nguyen-Tuong and Peters, 2008] Nguyen-Tuong, D. and Peters, J. (2008). Local Gaussian process regression for real-time model-based robot control. In *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 380–385.

[Paige and Van Loan, 1981] Paige, C. and Van Loan, C. (1981). A Schur decomposition for Hamiltonian matrices. *Linear Algebra and its Applications*, 41:11–32.

[Pearson, 1894] Pearson, K. (1894). Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110.

[Rasmussen, 2000] Rasmussen, C. E. (2000). The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, pages 554–560. MIT Press.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press, Cambridge, MA, USA.

[Sato and Ishii, 2000] Sato, M. A. and Ishii, S. (2000). On-line EM algorithm for the normalized Gaussian network. *Neural Computation*, 12(2):407–432.

[Schaal and Atkeson, 1998] Schaal, S. and Atkeson, C. G. (1998). Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084.

[Schwarz, 1978] Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464.

[Scott and Szewczyk, 2001] Scott, D. W. and Szewczyk, W. F. (2001). From kernels to mixtures. *Technometrics*, 43(3):323–335.

[Shi et al., 2009] Shi, T., Belkin, M., and Yu, B. (2009). Data spectroscopy: eigenspace of convolution operators and clustering. *The Annals of Statistics*, 37(6B):3960–3984.

[Srinivasan et al., 2010] Srinivasan, B. V., Qi, H., and Duraiswami, R. (2010). GPUML: Graphical processors for speeding up kernel machines. In *Siam Conf. on Data Mining. Workshop on High Performance Analytics - Algorithms, Implementations, and Applications*.

[Sung, 2004] Sung, H. G. (2004). *Gaussian Mixture Regression and Classification*. PhD thesis, Rice University, Houston, Texas.

[Tanaka and Tsuda, 2008] Tanaka, K. and Tsuda, K. (2008). A quantum-statistical-mechanical extension of Gaussian mixture model. *Journal of Physics Conf. Series*, 95.

[Tipping and Bishop, 1999] Tipping, M. E. and Bishop, C. M. (1999). Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482.

[Todorov and Jordan, 2002] Todorov, E. and Jordan, M. I. (2002). Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, 5:1226–1235.

[Tsuda, 2009] Tsuda, K. (2009). Machine learning with quantum relative entropy. In *Journal of Physics: Conf. Series*, volume 143, page 012021. IOP Publishing.

[Vijayakumar et al., 2005] Vijayakumar, S., D'souza, A., and Schaal, S. (2005). Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634.

[Warmuth and Kuzmin, 2010] Warmuth, M. and Kuzmin, D. (2010). Bayesian generalized probability calculus for density matrices. *Machine Learning*, 78(1):63–101.

[Wilson and Bobick, 1999] Wilson, A. D. and Bobick, A. F. (1999). Parametric hidden Markov models for gesture recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(9):884–900.

[Wolpert et al., 2011] Wolpert, D. M., Diedrichsen, J., and Flanagan, J. R. (2011). Principles of sensorimotor learning. *Nature Reviews*, 12:739–751.

[Wu, 1983] Wu, C. (1983). On the convergence properties of the EM algorithm. *Annals of Statistics*, 11:95–103.